И.Б. ШУБИНСКИЙ

Надежные отказоустойчивые информационные системы

Методы синтеза

ШУБИНСКИЙ И.Б.

НАДЕЖНЫЕ ОТКАЗОУСТОЙЧИВЫЕ ИНФОРМАЦИОННЫЕ СИСТЕМЫ

Методы синтеза

ООО «Журнал Надежность»
Москва, 2016 г.

ББК 39.973
Ш 93
УДК 681.31

Шубинский И.Б.
Надежные отказоустойчивые информационные системы. Методы синтеза / И.Б. Шубинский. – М.: «Журнал Надежность», 2016, – 546 с., ил.

ISBN……..

В книге приведены концептуальные положения обеспечения структурной и функциональной надежности информационных систем на всех этапах их жизненного цикла. Различные виды структурного резервирования рассмотрены с учетом ограничений в обнаружении отказов. При этих условиях оценена их эффективность и установлены предельные возможности структурного резервирования в предположении бесконечного количества резервных устройств. Рассмотрены способы обеспечения функциональной надежности программных средств, в том числе приведены рекомендации по разработке спецификации требований к программам, описана технология разработки архитектуры надежной программы, приведены хорошо апробированные правила и рекомендации по проектированию надежного программного обеспечения и его реализацию, а также интеграции с аппаратными средствами системы.

Представлены теоретические и практические положения адаптивной отказоустойчивости (активной защиты) информационных систем, в том числе методы и дисциплины активной защиты, способы ее практической реализации. Предложен метод синтеза активной защиты, а также результаты исследования надежности информационных систем с различными дисциплинами активной защиты. Оценена ее эффективность по отношению к традиционным методам структурного резервирования.

Рассмотрены принципы обеспечения функциональной безопасности информационных систем, обоснована возможность перезапуска независимых каналов в двухканальных безопасных системах, разработаны правила определения допустимого времени гарантированного обнаружения одиночных и двойных отказов, одиночных и двойных отказов, разработан метод синтеза комбинированной двухуровневой информационной системы, к которой предъявлены повышенные требования по функциональной безопасности.

Для подтверждения соответствия надежности и функциональной безопасности разработан метод ускоренных натурных испытаний информационной системы и приведен пример его практической реализации, развиты методики сертификационных испытаний по требованиям безопасности информации и декларирования соответствия программных средств.

В конце каждой главы содержатся контрольные вопросы по наиболее сложному и значимому материалу главы. Книга рассчитана, в первую очередь, на специалистов, занимающихся практической работой по разработке, производству, эксплуатации и модификации информационных систем. Она предназначена научным работникам в области структурной надежности различных дискретных систем, преподавательскому составу, аспирантам и студентам, специализирующимся в области информационных систем, а также в области автоматизированных систем управления.

УДК 681.31
ББК 32.972

ISBN ……

© ООО «Журнал Надежность», 2016

Предисловие

Информационные системы представляют собой сложные программно-аппаратные комплексы и являются неотъемлемой частью современных технических систем. Многие из них в составе автоматизированных систем управления осуществляют управление технологическими процессами в реальном масштабе времени. Структурная и функциональная надежность и безопасность информационных систем оказывают определяющее влияние на эффективность функционирования автоматизированных систем управления. Отсюда очевидно, что без обеспечения надежности информационных систем нельзя добиться бесперебойной и безошибочной работы систем управления.

Задачи построения надежных информационных систем имеют комплексный характер – они не могут ограничиться только структурной или только функциональной надежностью. Здесь под *структурной надежностью* подразумевается надежность продукции – объектов (элементов, систем). *Функциональная надежность* – это надежность оказания услуг (выполнения процессов сбора, обработки, передачи информации, управления подчиненными объектами). Только сочетание способов повышения и одной, и другой составляющих надежности дает ощутимый эффект. Этот эффект может выражаться в значительном повышении надежности системы и при этом в экономии объема дополнительных ресурсов. Например, контроль состояния структурной надежности объекта информационной системы может осуществляться с помощью методов функциональной надежности путем оценки безошибочности выдаваемых объек-

том результатов обработки информации. С другой стороны, для обнаружения сбоев аппаратуры чаще всего применяют методы, основанные на анализе промежуточных и выходных результатов. Известно, что реализация мажоритарного резервирования осуществляется комплексно методами структурной надежности (структурная избыточность) и методами функциональной надежности (программная реализация логических действий восстанавливающего органа). Можно привести множество подобных примеров. Главное, чтобы применяемые способы комплексного повышения надежности органично соответствовали структуре и процессам функционирования исследуемого объекта. Это условие не всегда выполнимо, особенно в отношении традиционных способов. Так, например, широко распространенные схемы структурного резервирования в информационных системах построены без учета реализуемых функций. Во многих схемах резервирования отсутствует разветвленный достоверный контроль состояния надежности составных элементов резервированных объектов. Одна из главных причин того, что разработчики информационных систем не уделяют должного внимания контролю надежности объектов систем, вызвана недостатками теоретических моделей надежности резервированных объектов. В большинстве известных моделей надежности резервированных объектов предполагается идеальный контроль (полный непрерывный с гарантированной достоверностью и к тому же с такими техническими средствами контроля, которые по предположению авторов никогда не отказывают). Подобные предпосылки часто приводят к заблуждениям относительно возможностей традиционных схем резервирования. Практика обычно развенчивает эти иллюзии.

Комплексное решение проблемы обеспечения надежности служит также методическим фундаментом, на котором созданы и развиваются способы обеспечения функциональной бе-

зопасности информационных систем. Эти задачи чрезвычайно актуальны в настоящее время для информационных систем реального времени, которые управляют критически важными, представляющими повышенную опасность для окружающей среды, объектами. К подобным системам предъявляются повышенные требования по надежности и функциональной безопасности. Подтверждение соответствия системы этим требованиям может быть практически выполнено на основе ускоренных натурных испытаний при доказательстве адекватности этих испытаний реальным условиям. Подтверждение соответствия заданным требованиям к системе тесно связано с подтверждением соответствия к качеству, функциональной и информационной безопасности программного обеспечения.

Предлагаемая читателю книга «*Надежные отказоустойчивые информационные системы. Методы синтеза*» является *третьей* заключительной частью проекта «Надежность информационных систем». В *первой книге* проекта [1] под названием «Структурная надежность информационных систем. Методы анализа» приведены основные понятия и показатели структурной надежности, подробно описаны Марковские модели надежности и графовые полумарковские методы расчета, рассмотрены инженерные методы расчета и приближенного прогнозирования структурной надежности информационных систем с оценками погрешностей расчетов и прогнозов. Во *второй книге* «Функциональная надежность информационных систем. Методы анализа» [2] впервые представлена теория функциональной надежности информационных систем как составная часть общей теории надежности. Она включает понятия и определения; основные угрозы нарушения функциональной надежности информационных систем; систему показателей и методы их расчета; методы оценки надежности выполнения цифровыми устройствами предусмотренных функций; методы и модели

оценки функциональной надежности программного обеспечения. В отдельном разделе книги рассмотрена функциональная надежность информационных систем, осуществляющих управление критически важными объектами, в том числе особенности оценки опасных отказов и рисков, сбойных ошибок, ошибок операторов. Приведены требования к спецификации и архитектуре функционально надежного программного обеспечения.

Третья книга проекта содержит методы синтеза надежности информационных систем.

Основные идеи построения надежных и безопасных информационных систем изложены в ***первой главе*** «Основы надежности и отказоустойчивости информационных систем». Эта глава предназначена для лиц, принимающих решения, которым достаточно понимать проблему на концептуальном уровне. В этой главе изложены постулаты обеспечения надежности информационных систем, ключевые понятия избыточности, отказоустойчивости, отказобезопасности и киберзащищенности систем.

Во ***второй главе*** книги существенно развиты традиционные модели надежности как невосстанавливаемых, так и восстанавливаемых объектов с общим и раздельным постоянным резервированием, а также резервированием замещением, на основе учета скрытых отказов и эффективности их обнаружения. Особый интерес представляют модели надежности объектов с мажоритарным резервированием, в которых впервые совмещены модель структурной надежности мажоритарного объекта и функциональной надежности его восстанавливающего органа.

В главе выполнена предельная оценка надежности резервированных объектов с бесконечным количеством резервных устройств, но ограниченной эффективностью системы обнаружения отказов, которая убедительно показывает, что не следует

питать иллюзии относительно достижения требуемого уровня надежности системы за счет увеличения количества резервных устройств. Скромные возможности структурного резервирования как с восстановлением, так и, тем более, без восстановления, вызывают необходимость в разработке нетривиальных методов обеспечения отказоустойчивости информационных систем. Это тем более важно, потому что рассчитывать на эффективное применение временного и/или функционального резервирования в информационно-управляющих системах, работающих в реальном времени, не представляется возможным. В главе также систематизированы методы информационного резервирования, основанные на различных требованиях к эффективности контроля достоверности хранимой информации.

В ***третьей главе*** представлены оригинальные методы построения модульных информационных систем с адаптивной отказоустойчивостью. Изложены идеи адаптивной отказоустойчивости (активной защиты), приведены способы организации активной защиты, способы автоматического обнаружения и устранения неисправностей, временные интервалы и дисциплины активной защиты. Оценена эффективность применения методов активной защиты. Изложен метод синтеза активной защиты. Показаны несомненные преимущества активной защиты перед традиционными методами структурного резервирования как в отношении надежности, так и в отношении технико-экономических показателей. При этом активная защита обеспечивает возможности адаптации системы не только к отказам, но и к сбойным и программным ошибкам.

В ***четвертой главе*** описаны методы построения надежных программных средств, в том числе обсуждены их характерные недостатки, приведены рекомендации по разработке спецификации требований к проектируемым программам, достаточно подробно раскрыта технология разработки архитектуры надеж-

ной программы. Большое внимание уделено вопросам проектирования надежного программного обеспечения и его реализации, в том числе верификации программ и их интеграции с аппаратными средствами, а также их аттестации, эксплуатации, сопровождению и конфигурации.

Пятая глава посвящена актуальной тематике функциональной безопасности информационно-управляющих систем критически важными и ответственными объектами. Рассмотрены ключевые понятия состояния безопасности, функции и полноты безопасности. Изложены основные принципы функциональной безопасности, в том числе принципы отказобезопасности, избыточности, разнообразия и локализации развития неблагоприятных событий. Произведена оценка допустимого времени обнаружения одиночного и двойного опасных отказов. Описаны и проанализированы модели функциональной безопасности двухканальной системы со встроенными средствами диагностики и с внешним контролем. Рассмотрена проблема перезапуска каналов. Предложена и изучена модель для оценки вероятности возникновения опасных отказов при перезапуске двухканальных систем. Совместное применение разных информационных технологий построения информационных систем управления ответственными и критически важными объектами создает естественные условия для построения двухуровневой системы управления. В двухуровневой системе возможно применение небезопасных систем. В главе приведены результаты математического моделирования различных стратегий построения двухуровневых информационных систем. Показано, что значительно эффективнее по сравнению с другими стратегиями является та, которая при наличии недостаточно безопасных составных систем позволяет рационально использовать естественную дополнительную информацию, имеющуюся по результатам предыдущих циклов управления.

В ***шестую главу*** включены принципы и методы подтверждения соответствия информационных систем требованиям Технических регламентов и нормативных документов. Приведены методы испытаний программных средств на соответствие по требованиям качества и функциональной безопасности, а также безопасности информации. Рассмотрены вопросы практического применения методов испытаний программ. Особое внимание уделено проблеме ускорения испытаний. Описаны основные пути ускорения испытаний на основе понижения дисперсий получаемых показателей качества, надежности и безопасности испытуемых объектов: метод Монте-Карло и метод значимой выборки. Изложена инженерная методика ускоренных натурных испытаний на функциональную надежность и функциональную безопасность информационных систем управления ответственными или критически важными объектами, в том числе базовые теоретические положения, порядок проведения испытаний и оценка их продолжительности, порядок обработки результатов и формы представления данных. Приведен пример практического применения ускоренных натурных испытаний информационной системы диспетчерского управления на железнодорожном транспорте. Приведены основные положения методик испытаний программных средств по требованиям качества и функциональной безопасности и отсутствия недекларированных возможностей в программах, а также порядок подтверждения комплексной безопасности программных средств.

В конце каждой главы книги содержатся контрольные вопросы по наиболее сложному и значимому материалу главы. Книга рассчитана, в первую очередь, на специалистов, занимающихся практической работой по разработке, производству, эксплуатации и модификации информационных систем. Она предназначена научным работникам в области надежности программно-аппаратных систем, профессорско-преподавательскому составу,

аспирантам и студентам, специализирующимся в области информационных технологий, в области информационных систем, а также в области автоматизированных систем управления.

Данная книга завершает проект из трех книг, предназначенный для представления широкой аудитории специалистов методов анализа структурной и функциональной надежности информационных систем и, особенно, методов синтеза надежности, отказоустойчивости и функциональной безопасности таких систем.

Автору доставляет удовольствие выразить признательность докт. физ.-мат. наук Х. Шебе (Германия) за многолетнее творческое содружество в области функциональной надежности и функциональной безопасности систем, главному инженеру Российских железных дорог канд. техн. наук В.А. Гапановичу за организацию и проведение масштабных научно-практических работ в области надежности больших систем и привлечение автора к этим работам, что способствовало усилению прикладной направленности данных книг. Автор благодарен своим ученикам проф. Е.Н. Розенбергу за участие в разработке пп. 5.4, 5.6, 5.8 книги III, проф. Л.В. Уткину за участие в разработке пп. 6.4 и 6.5 книги III, докт. техн. наук А.М. Замышляеву за активное участие в обсуждении ряда разделов данных книг. Особую благодарность хочу выразить моей жене – В.В. Дубровской, которая с большим пониманием, чуткостью и долготерпением относилась к моей работе над этими книгами в отпусках, в выходные дни и вечера.

ГЛАВА III.1. ОСНОВЫ НАДЕЖНОСТИ И ОТКАЗОУСТОЙЧИВОСТИ ИНФОРМАЦИОННЫХ СИСТЕМ

III.1.1. Введение

Для обеспечения надежности информационных систем следует применять весь известный арсенал методов, способов, технических решений. В этом направлении много идей можно почерпнуть из области биологических систем. Идея применения знаний о живой природе для решения инженерных задач принадлежит *Леонардо да Винчи*, который пытался построить летательный аппарат с машущими крыльями, как у птиц – орнитоптер. Появление *кибернетики*, рассматривающей общие принципы управления и связи в живых организмах и машинах, стало стимулом для более широкого изучения строения и функций живых систем с целью выяснения их общности с техническими системами, а также использования полученных сведений о живых организмах для создания новых приборов, механизмов, материалов и т.п.

Исследования возможностей человека показали, что они обладают рядом важных и ценных особенностей и преимуществ перед всеми самыми современными вычислительными устройствами. Эти особенности, изучение которых очень важно для дальнейшего совершенствования информационных систем, следующие:

- высокая надёжность, значительно превышающая надёжность технических систем (последние выходят из строя при

обрыве в цепи одной или нескольких деталей; при гибели же миллионов нервных клеток из миллиардов, составляющих головной мозг, работоспособность системы сохраняется);

▪ совершенное и гибкое восприятие внешней информации вне зависимости от формы, в которой она поступает (например, от почерка, шрифта, цвета текста, чертежей, тембра и других особенностей голоса и т.п.);

▪ миниатюрность элементов нервной системы: при количестве элементов 10^{10}-10^{11} объём мозга человека 1,5 $дм^3$. Цифровое устройство с таким же числом элементов заняло бы объём в несколько сотен, а то и тысяч $м^3$ [3];

▪ экономичность работы: потребление энергии мозгом человека не превышает нескольких десятков *вт*;

▪ высокая степень самоорганизации нервной системы, быстрое приспособление к новым ситуациям, к изменению программ деятельности.

III.1.2. Постулаты обеспечения надежности

Изучение механизмов, обеспечивающих надёжность человека, и в первую очередь его нервной системы, очень важно для техники, т.к. решение этой первоочередной технической проблемы даст ключ к обеспечению надёжности многих технических систем. При этом будем руководствоваться рядом утверждений, которые принимаются без доказательств и служат основой научной теории обеспечения надежности и отказоустойчивости информационных систем. Речь идет о следующих основных **постулатах**:

1. Не существует абсолютной надежности информационных систем.

2. Чем сложнее система, тем больше задач она выполняет и тем ниже ее надежность.

3. Введение различных видов контроля в состав информационной системы обеспечивает ее наблюдаемость, но вместе с тем приводит к снижению надежности системы.

4. Необходимым условием повышения надежности системы есть введение избыточности. Сочетание контроля с используемой избыточностью повышает надежность информационной системы.

5. Надежность информационной системы должна обеспечиваться на всех этапах жизненного цикла

6. Уровень надежности информационной системы ограничен экономическими рисками заказчика и эксплуатирующей организации.

Прокомментируем перечисленные постулаты.

Постулат 1.

Невозможность достижения абсолютной надежности технических систем, как правило, не вызывает сомнения у специалистов. Вместе с тем, некоторые заказчики информационных систем, особенно систем реального времени, требуют максимально высокого уровня гарантии их надежности, который сопоставим с абсолютной надежностью. Достичь этого невозможно по многим причинам. Прежде всего, надо учитывать тот непреложный факт, что частота сбоев цифровой техники на три и более порядков превышает частоту ее отказов [2]. Обнаружение сбоев возможно путем создания многоуровневой разветвленной системы контроля, что, однако, приводит к значительному удорожанию системы. В принципе возможно построение устойчивых к сбойным ошибкам алгоритмов системы. Однако нет универсальных рецептов построения таких алгоритмов для решения предусмотренных в такой системе задач и нет гарантии того, что будет обеспечена толерантность алгоритмов ко всем сбойным ошибкам. Можно ожидать, что на коротких отрезках времени функционирования информационной системы будет

обеспечен требуемый заказчиком уровень гарантии надежности, однако для этого потребуются значительные усилия разработчиков и, кроме того, большие дополнительные затраты.

Постулат 2

Стремление расширить функциональные возможности информационных систем вызвано активным развитием систем управления производственными и другими объектами и напрямую связано с увеличением объема, как программного обеспечения, так и аппаратуры. Это естественно вызывает рост сложности системы и снижение ее надежности. Специфика информационных систем такова, что невыполнение какой – либо прикладной задачи не приводит к отказу системы в целом. Однако это событие следует классифицировать как частичный отказ, что означает снижение надежности системы в допустимых пределах. Если система не выполнила хотя бы однократно больше допустимого количества прикладных задач, то такое событие приводит к полному отказу информационной системы. Следовательно, с позиций надежности информационная система не может рассматриваться как простая система с последовательно соединенными элементами. Для ее исследования нельзя применять принцип «слабого» звена. Это сложная многофункциональная система с частичными отказами. Однако очевидно, что, несмотря на отсутствие линейной зависимости надежности системы от надежности ее элементов, рост сложности системы приводит к снижению ее надежности.

Постулат 3

С помощью средств контроля обеспечивается наблюдаемость и определяется факт возникновения отказа составного средства информационной системы и/или устанавливается факт возникновения функционального отказа при реализации конкретной прикладной задачи. Эти функции контроля возлагаются в информационной системе главным образом на встроен-

ные программные, аппаратные и/или программно – аппаратные средства контроля. Многоуровневая система контроля функционирования элементов и информационной системы в целом обеспечивает оперативную ее наблюдаемость, что позволяет своевременно предпринимать необходимые действия по устранению или блокированию возникающих ошибок и особенно отказов. Вместе с тем, введение указанной системы контроля снижает надежность информационной системы. Действительно, средства контроля – это дополнительное оборудование, это дополнительные программные средства, которые так же как и основные программно – аппаратные средства имеют обыкновение допускать ошибки в работе и отказывать. Для того, чтобы определить, какие средства – основные или дополнительные отказали нужно проконтролировать ранее введенные средств контроля. Здесь решается задача «кто сторожит сторожей?». Объем аппаратуры средств контроля второго уровня конечно меньше, чем первого уровня, однако и эта аппаратура имеет обыкновение ошибаться и отказывать. И так далее.

Важно подчеркнуть, что *в условиях отсутствия средств оперативного устранения или блокирования обнаруженных средствами контроля ошибок или отказов введение средств контроля в состав информационной системы приводит к снижению ее надежности.*

Для пояснения этого положения рассмотрим объект информационной системы без какой – либо избыточности. Это предполагает, что любой отказ элемента объекта или любое нарушение его функции приводит к его структурному или функциональному отказу. Таким объектом может быть процессор обработки информации, устройство ввода/вывода информации, терминал автоматизированное рабочее место пользователя и т.д. Обозначим вероятность безотказной работы объекта за фиксированное время как P_0. Устройство охвачено средствами контроля. Под

средствами контроля понимается введение в состав устройства встроенных аппаратных, программных, программно – аппаратных средств контроля и диагностирования, а также применение функциональных контролей, которые в совокупности правильно обнаруживают отказы устройства. Вероятность безотказной работы средств контроля за фиксированное время, обозначим в виде $P_К$. Тогда вероятность безотказной работы объекта и средств контроля за фиксированное время равна произведению их вероятностей безотказной работы $P_{ОК} = P_О \cdot P_К$.

Эффективность контроля оценивается с помощью вероятности α правильного обнаружения отказов. Это комплексный показатель эффективности средств контроля, учитывающий полноту охвата устройства контролем и диагностикой, непрерывность, своевременность и достоверность результатов контроля и диагностирования, возможности применяемых функциональных контролей. Обозначим элементарное событие обнаружения отказа каким – либо одним *i-м* средством контроля (i=1 ...N) как A_i. Тогда сложное событие A, состоящее в том, что осуществляется хотя бы одно из событий $A_1,\ldots, A_N$, равно $A = \sum_{i=1}^{N} A_i$. Отсюда вероятность правильного обнаружения отказа устройства одноуровневым контролем α_1 есть вероятность суммы конечного числа событий:

$\alpha_1 = P\{A\} = P\{\sum_{i=1}^{N} A_i\} = 1 - P\{\prod_{i=1}^{N} \overline{A_i}\} = 1 - \prod_{i=1}^{N} \overline{\alpha}_i.$ В свою очередь, $\overline{\alpha}_1 = 1 - \alpha = \prod_{i=1}^{N} \overline{\alpha_i}$.

Теперь рассмотрим ситуацию контроля «сторожа», когда средства встроенного контроля и диагностирования в свою очередь контролируются вторым уровнем контроля (вероятность правильного обнаружения отказов средствами второго уровня контроля обозначим как α_2).

Вероятность безотказной работы средств первого уровня контроля за фиксированное время, такое же, как и у контролируемого объекта, обозначим как $P_{К1}$, а второго уровня контроля – $P_{К2}$.

Надежность объекта с контролем определяется как сумма вероятностей двух событий: *А* – в течение фиксированного времени отсутствуют отказы объекта и средств первого уровня контроля, который обеспечивает контроль объекта; *Б* – в течение фиксированного времени при условии отсутствия отказа объекта возник и обнаружен отказ средств первого уровня контроля. Это возможно при условии исправности средств второго уровня контроля, который обеспечивает контроль средств первого уровня. При этом предполагается что вероятность одновременных отказов объекта и средств контроля первого уровня ничтожно мала.

Таким образом, вероятность безотказной работы объекта с двухуровневым контролем в течение фиксированного времени определяется как

$$P_{О\text{К}} = P_О \cdot P_{\text{К1}} + P_О \cdot \overline{P}_{\text{К1}} \cdot P_{\text{К2}} \cdot \alpha_2 = P_О[1 - \overline{P}_{\text{К1}}(1 - P_{\text{К2}} \cdot \alpha_2)] \quad (1.1)$$

Логика формулы (1.1) проста – при одноуровневом контроле вероятность безотказной работы контролируемого объекта равна произведению вероятности безотказной работы объекта на вероятность безотказной работы средств контроля. Второй уровень контроля есть ни что иное, как введение избыточности, с помощью которой могут фильтроваться отказы средств первого уровня контроля. Так, в пределе при идеальной надежности ($P_{\text{К2}}$=1) и при идеальной эффективности (α_2=1) средств контроля второго уровня надежность контролируемого объекта будет стремиться к надежности самого объекта. Аналогичный результат можно ожидать и при идеальной надежности средств контроля первого уровня. При любых вариантах организации

контроля, эффективности и надежности средств контроля и диагностики в принципе невозможно превысить уровень надежности исходного объекта.

Постулат 4

Необходимым условием повышения надежности системы есть введение избыточности. Использование избыточности в сочетании с контролем повышает надежность информационной системы

Накопление опыта является в определенной степени прогнозированием будущего, позволяющего человеку успешно участвовать в борьбе за сохранение и расширение «жизненного пространства». «При этом важно учитывать, что наличная ситуация по своим пространственно-временным параметрам заведомо отличается от прежних ситуаций, с которыми сталкивался организм. Такую ситуацию можно спрогнозировать лишь приблизительно и с какой-то долей вероятности, поэтому успешное приспособление может быть обеспечено только путем выработки таких программ предстоящего поведения, которые носят заведомо избыточный характер по отношению к породившему их комплексу внешних факторов. Избыточность таких программ и обеспечивает возможность построения активного поведения в новой, отличной от прежних ситуации» [5].

Избыточность – свойство большинства существующих технических объектов (систем) выполнять больше функций, чем требуется, и иметь ресурсы выше, чем необходимо для выполнения только требуемых функций. ***Избыточность*** *– это наличие в техническом объекте возможностей сверх тех, которые минимально необходимы для обеспечения его нормального функционирования.* С диалектической точки зрения *избыточность* является одним из условий перехода количества в *новое качество.* Это одно из свойств *вещи* позволяющее ей относительно просто превращаться в другую *вещь*.

В технических системах в настоящее время применяются различные механизмы рационального использования избыточности для парирования структурных и функциональных отказов. Будем называть их средствами защиты от отказов.

Показателем эффективности защиты от отказов является вероятность β успешной адаптации объекта (информационной системы) со средствами защиты к отказам

$$\beta = P\{\Omega \le \Omega_д\},$$

где Ω – ресурс (структурный, временной и т.д.), который возможно использовать без ухудшения других показателей эффективности работы объекта или системы в целом для защиты от отказов; $\Omega_д$ – допустимый расход ресурса, при котором один или несколько показателей эффективности объекта (системы) достигают предельно допустимого минимального значения.

Оценим вероятность того, что в процессе выполнения задачи либо не возникнут отказы, либо возникшие частичные отказы будут успешно нейтрализованы указанными средствами обеспечения защиты на основе допустимых затрат избыточных ресурсов. Обозначим вероятность безотказного выполнения задачи как $P_з$, и вероятность того, что внутри средств защиты в процессе выполнения задачи не возникли отказы как P_1. Тогда вероятность безотказного выполнения задачи под прикрытием средств защиты оценивается выражением

$$P_{з1}=P_з P_1+(1-P_з)P_1\beta=1-g_з-g_1+g_з g_1+\beta\cdot(g_з-g_з g_1),$$

где $g_1=1-P_1$; $g_з=1-P_з$.

Поскольку $g_1 \ll 1$ и $g_з \ll 1$, то с погрешностью, не превышающей второго порядка малости, справедливо выражение

$$P_{з1}=1-g_1-g_з(1-\beta). \qquad (1.2)$$

Определим допустимые границы ненадежности средств защиты (а это означает также допустимые объемы средств защи-

ты) в зависимости от их эффективности и от ненадежности основных средств выполнения процесса.

Защита от отказов имеет смысл только в том случае, если выполняется условие

$$P_{з1} > P_{з}, \tag{1.3}$$

при котором вероятность безотказной работы системы с защитой от отказов выше вероятности безотказного выполнения задачи без средств защиты.

Подставив в неравенство (1.3) выражение (1.2), находим соотношение между вероятностями отказов при выполнении задачи ($g_з$) и средств защиты (g_1)

$$g_1 < g_з \cdot \beta. \tag{1.4}$$

Полученное неравенство (1.4) имеет фундаментальное значение в построении средств защиты от отказов. Прежде всего, оно устанавливает, что объем средств защиты от отказов не должен превышать объем программно – аппаратных средств информационной системы, реализующих данную задачу. Это действительно так, поскольку всегда $\beta<1$и поэтому всегда справедливо неравенство $g_з < g_1$. В свою очередь, вероятности отказов прямо пропорциональны объемам аппаратно – программных средств защиты ($g_1 \equiv W_1$) и выполнения задачи ($g_з \equiv W_з$), где W_1 и $W_з$ – объемы средств защиты и выполнения задачи соответственно. Следовательно, на основании неравенства (1.4) имеет место неравенство $W_1 < W_з \cdot \beta$, из которого, в частности, следует тривиальный вывод о том, что если защита от отказов не эффективна ($\beta \to 0$), то, нет смысла в применении средств защиты.

Постулат 5

Надежность информационной системы должна обеспечиваться на всех этапах жизненного цикла от начала разработ-

ки концепции системы, оценки рисков, разработки требований к системе до завершения ее эксплуатации и утилизации. Этот постулат означает, что усилия в обеспечении надежности не должны ограничиваться этапами создания высоконадежных объектов и систем в целом – не менее важное значение следует уделять этапам модификации программного обеспечения, модернизации технических средств системы, а также эффективному техническому сопровождению и объективному оцениванию соответствия надежности системы заданным или измененным требованиям. Этапы жизненного цикла надежности информационной системы приведены в п. III.1.3.

Постулат 6

Уровень надежности информационной системы ограничен приемлемыми остаточными экономическими рисками пользователя. Этот постулат основывается на широко распространенном в Европе принципе ALARP (As Low As Reasonably Practicable – **риск настолько низок, насколько это достижимо на практике**) [6, 8, 10]. Этот принцип может быть представлен диаграммой, приведенной на рис. III.1.1.

В диаграмме ALARP предусмотрено, что некоторые риски могут быть настолько велики, а последствия настолько неприемлемы, что они недопустимы и не могут быть ни в коем случае оправданы. Верхняя граница определяет уровни риска, которые являются недопустимыми. Если уровень риска не может быть опущен ниже этой границы, тогда риск должен быть исключен.

Нижняя граница диаграммы устанавливает широкую область применения, в которой уровень риска считается настолько низким, что все усилия по его еще большему снижению, скорее всего, не будут оправданы.

Зона между верхней и нижней границей называется областью ALARP. Следует подчеркнуть, что недостаточно опреде-

лить факт нахождения риска какого-либо вида в области ALARP Его следует сделать настолько низким, насколько это достижимо на практике. Существуют различные способы демонстрации ALARP. В ряде случаев достаточно указать, что использованы самые лучшие из имеющихся современных стандартов и практических наработок. В случае новых видов деятельности или когда адекватность современных стандартов и практических наработок находится под сомнением, применяют концепцию анализа затрат и выгод.

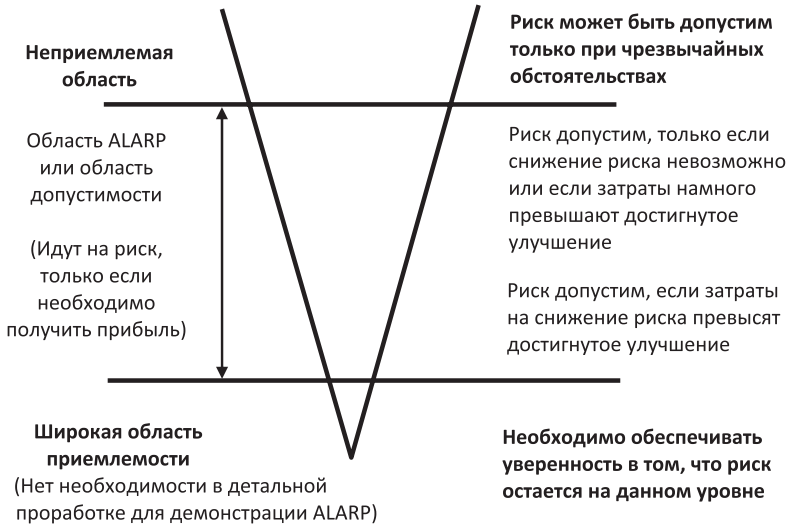


Рис. III.1.1. Диаграмма ALARP

При установлении допустимого уровня риска предусматривается наличие затрат на проведение мероприятий по достижению этого уровня риска для стороны организации, несущей ответственность за риск.

Допустимый уровень риска $R_{\text{доп}}$ в соответствии с принципом ALARP, это такой уровень риска, для которого затраты на его достижение являются экономически эффективными, то есть:

$$C(R_{\text{ДОП}}) \le C_{\text{ОГР}} \text{ при условии, что } C_{\text{ЧП}} > 0,$$

где $C_{\text{ОГР}}$ – ограничивающий фактор, например, максимальный размер затрат, которые готова понести организация при возникновении данного вида риска; $C_{\text{ЧП}}$ – чистая приведенная стоимость, которая является разностью между дисконтированной выгодой от достижения более низкого уровня риска и затратами на проведение мероприятий по достижению этого уровня риска за период рассмотрения. Положительная величина $C_{\text{ЧП}}$ указывает на то, что затраты оправданы по данной учетной ставке.

Уровень риска, при котором значение $C_{\text{ЧП}} = 0$ представляет собой нижнюю границу допустимости для принципа ALARP, т.к. дальнейшие затраты по его еще большему снижению не будут оправданы.

Чтобы определить верхнюю границу допустимости для принципа ALARP необходимо определить допустимый уровень риска как можно более близкий к оптимальному по критерию $C_{\text{ЧП}} \rightarrow \max$ и удовлетворяющий указанным ограничениям.

III.1.3. Жизненный цикл надежности информационной системы

Различные решения и действия, направленные на достижение и сохранение надежности информационной системы, принимаются и проводятся на разных стадиях ее разработки и эксплуатации. Отсюда уже давно возникла хорошая идея описать жизненный цикл системы [6, 7, 8, 9 и др.].

Под жизненным циклом системы понимается последовательность стадий на протяжении всей жизни системы, от планирования до вывода ее из эксплуатации и утилизации. Жизненный цикл состоит в планировании управлении, проверке и наблюдении всех аспектов системы, включая информационную защищенность, надежность и функциональную безопасность, так

как система проходит через отдельные стадии для того, чтобы желаемый продукт с желаемым уровнем безопасности и по желаемой цене был предоставлен в течение оговоренного срока.

На рис. III.1.2 показан жизненный цикл надежности информационной системы. Он неразрывно связан с жизненным циклом самой системы и отражает современные воззрения к организации жизненного цикла для широкого класса сложных технических систем, ярким представителем которого является ИС.

На ***первой стадии*** формируется **концепция** создания ИС. Цель этой стадии состоит в создании достаточного представления о системе для того, чтобы все последующие стадии жизненного цикла надежности ИС могли удовлетворительно выполняться. При этом решаются три группы задач: 1) *Определение* области применения, концепции и целей данного проекта; анализ необходимости и достаточности финансирования и регламентация управления проектом; 2) *Проверка* связи проекта с надежностью; определение целей структурной и функциональной надежности, выявление источников угроз надежности, в первую очередь при взаимодействии с другими системами и при взаимодействии с людьми; получение и анализ информации по известным ранее источникам угроз, достигнутым уровням надежности для аналогичных или родственных систем; 3) *Документирование* всех результатов этой стадии (эти документы должны быть ключевыми входными данными для последующих стадий жизненного цикла); 4) *Верификация* соразмерности входных данных, корректности и согласованности с требованиями документации по поддержке надежности жизненного цикла, квалификации всех сотрудников, участвующих в данной работе, а также оценка гарантии того, что система разрабатывается в соответствии с требованиями к ней и применяемые при этом технологии, методы, способы, инструментальные средства являются адекватными.

На ***второй стадии*** производится **описание системы и устанавливаются технические условия.** Решаются три группы задач: 1) *Определение* профиля эксплуатационного задания, разработка описания системы, определение стратегий и условий эксплуатации и технического обслуживания ИС, определение ограничений связанных с инфраструктурой ИС; 2) *Оценка* имеющейся статистики надежности, предварительный анализ угроз, разработка политики обеспечения надежности на уровне всей системы, выбор критериев риска; 3) *Документирование* всех полученных результатов, включая политику безопасности и программу работ по надежности и безопасности ИС; 4) *Верификация* полученных документов и разработанного задания на последующие стадии жизненного цикла, оценка квалификации сотрудников, выполняющих задания данной стадии.

На ***третьей стадии*** выполняется **анализ риска**. Эта стадия предназначена для решения следующих задач: идентификация угроз, которые связаны с системой; идентификация событий, которые вызывают эти угрозы; определение риска, связанного с угрозами; разработка процесса непрерывного управления риском.

На этой стадии решаются следующие группы задач: 1) Систематический поиск и классификация всех рисков, возможных при нормальных условиях, идентификация скрытых угроз, выявление частоты возникновения событий, связанных с существующими угрозами, выявление / оценка размера воздействий существующих угроз, выявление риска для системы, связанного с каждой угрозой; 2) Анализ и классификация допустимости рисков, соответствующих каждой известной угрозе, разрабатывается протокол угроз как базис для управления риском. После того, как выявлены недопустимо высокие риски, подлежащие уменьшению или ликвидации средствами обеспечения надежности ИС, рассматриваются ожидаемые остаточные риски и

подтверждается их приемлемость. 3) Документирование всех полученных результатов, включая протокол угроз безопасности; 4) Верификация полученных документов, разработанного задания на последующие этапы жизненного цикла, оценка полноты оценки риска, оценка классификации допустимости рисков, оценка пригодности протокола угроз для рассматриваемой системы, оценка пригодности используемых на этом этапе методов (способов, процессов), инструментов и технических приемов (технологий), оценка квалификации сотрудников, выполняющих задания данного этапа.

На ***четвертой стадии*** разрабатываются требования к ИС. Решаются три группы задач: 1) Разработка требований ко всей системе с учетом специфики окружающей среды, выбор информационных технологий, разработка требований к качеству и организации управления, конфигурации ИС, разработка критериев доказательства выполнения требований и приемки, разработка и согласование плана аттестации системы; 2) Разработка требований к структурной и функциональной надежности системы, а также критериев приемки надежности, регламентация управления надежностью, разработка так называемой программы RAMS [6] (программы комплексного обеспечения надежности, эксплуатационной готовности, восстанавливаемости и безопасности).

Разработка системного задания по безопасности, содержащего описание системных требований безопасности, в том числе перечень рисков, которым необходимо противостоять, и целей безопасности, которые необходимо достичь с помощью функций безопасности, а также описание процедурных и административных регуляторов. 3) *Документирование* полученных результатов; 4) *Верификация* всех видов разработанных требований, критериев доказательства, системного задания по надежности, а также оценка использованных методов, способов, средств и оценка квалификации сотрудников.

На ***пятой стадии*** **распределяются требования** к подсистемам ИС и их надежности, в том числе определяются политики обеспечения надежности для функционально независимых подсистем с собственными доменами надежности, разрабатываются профили защиты, функции (технические регуляторы) безопасности. Уже на этой ранней стадии жизненного цикла следует начинать проведение оценки и доказательства структурной и функциональной надежности ИС. Это позволит разработчикам и заказчикам лучше понимать систему и предполагаемую эксплуатационную среду, устранить несоответствия и упущения в требованиях безопасности и предлагаемых функциях безопасности и регуляторах безопасности. Это в дальнейшем облегчит оценщикам анализ проектной и эксплуатационной документации, получение свидетельств доверия к достигнутым уровням структурной и функциональной надежности.

На ***шестой стадии*** производится непосредственная **разработка** (при необходимости конструирование отдельных составляющих системы) и ее **реализация.** На этой стадии решаются также задачи архитектурного и детального проектирования программных средств ИС, кодирования и тестирования программных компонентов, разработки интерфейсных средств, выбора и адаптации информационных технологий, протоколов обмена информацией, а также интеграции программных и аппаратных средств, верификации, выполнения планов обеспечения надежности системы и программы RAMS [6], выработки общего доказательства надежности ИС, документирования программных и аппаратных средств, разработки эксплуатационной документации.

На ***седьмой стадии*** предусматривается **изготовление** системы. Решаются также задачи реализации плана обеспечения надежности, тестирования аппаратных компонент и отладки программных средств, разработки документации, обучения персонала.

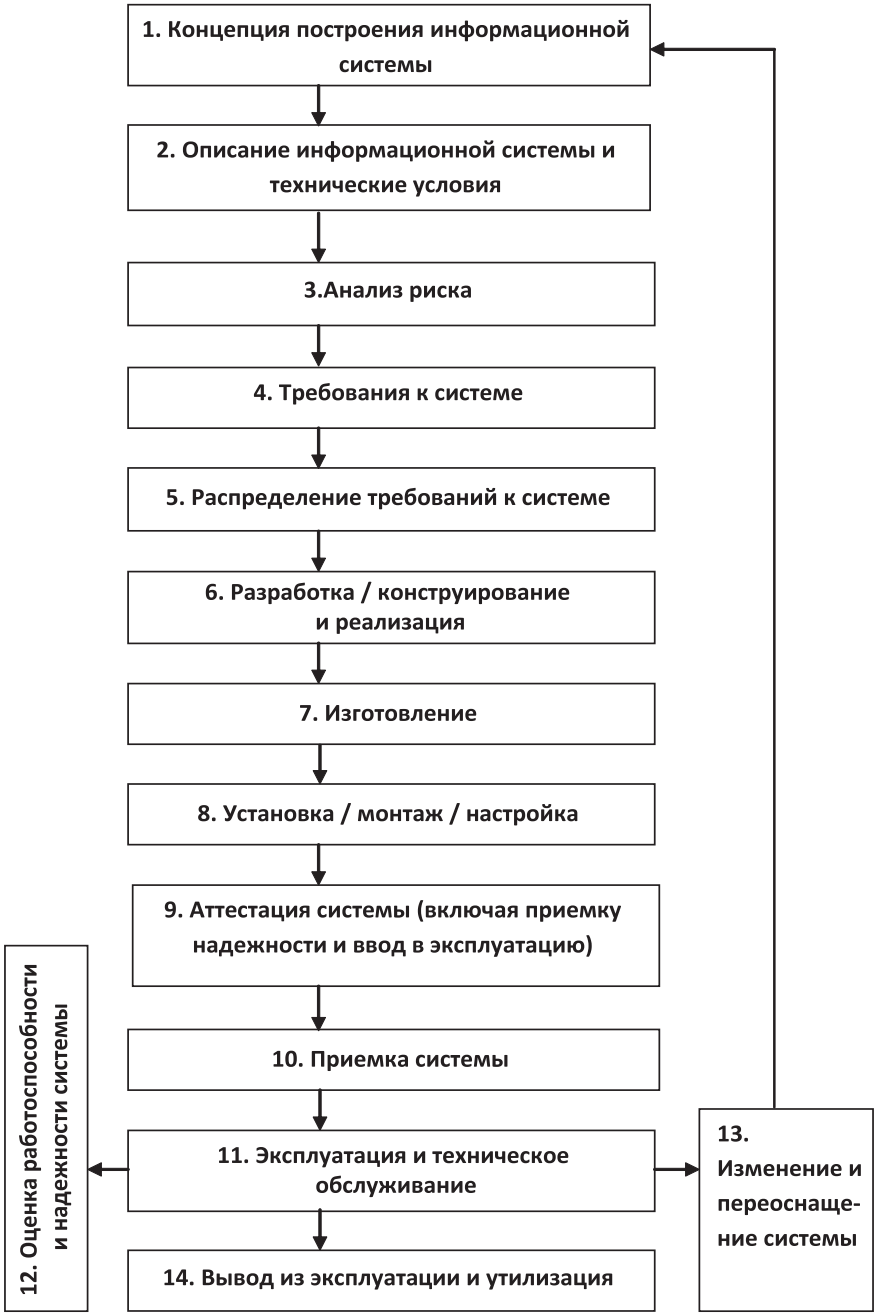


Рис. III.1.2. Жизненный цикл надежности информационной системы

На этой стадии также решаются задачи закупки и адаптации необходимого базового и прикладного программного обеспечения, включая программные средства обеспечения надежности, а также системная интеграция, конфигурирование и тестирование разработчиком / системным интегратором.

На ***восьмой стадии*** производится ***монтаж, установка и настройка системы***, обучение обслуживающего персонала, доработка эксплуатационной документации. На этой стадии также создается структура обеспечения надежности для административного и процедурного уровней, документируются политики, правила и процедуры обеспечения надежности с учетом задач системной интеграции

На ***девятой стадии*** производится ***подтверждение соответствия системы, включая приемку надежности и ввод в эксплуатацию.*** Решаются также задачи опытной эксплуатации, обучения персонала, детальной проработки и реализации методов аналитического и экспериментального доказательства надежности системы.

На ***десятой стадии*** осуществляется ***приемка системы*** в постоянную эксплуатацию. При этом решаются задачи обоснования возможности и целесообразности ввода системы в постоянную эксплуатацию, интеграции результатов доказательства надежности и функциональной безопасности. Решаются задачи оценки достигнутых уровней структурной и функциональной надежности элементов ИС. Затем осуществляется оценка надежности всей ИС. Это даст заказчику ИС независимое подтверждение, что все определенные ранее риски благодаря применению функций и регуляторов надежности уменьшены до приемлемого уровня. В протоколе испытаний перечисляются все обнаруженные уязвимости, а в материалах аудита описываются рекомендуемые действия по их устранению. Формируется план устранения недостатков, выявленных при оценивании надежности системы. Оп-

ределяется допустимость остаточных рисков после реализации плана устранения выявленных недостатков. Затем реализуется данный план, оценивается эффективность его выполнения. После чего производится ввод ИС в постоянную эксплуатацию.

Одиннадцатая стадия – стадия ***эксплуатации и технического обслуживания*** ИС охватывает подавляющую часть жизненного цикла и по стоимости находится на уровне (60 – 70)% от общей стоимости жизненного цикла системы. Она охватывает задачи текущего технического обслуживания и материально – технического обеспечения, мониторинга надежности системы, текущего обучения персонала, ведения протокола угроз. На этой стадии решаются задачи оптимизации технического обслуживания в целях снижения стоимости жизненного цикла системы при условии сохранения ее надежности на приемлемом уровне.

На ***двенадцатой стадии*** производится ***оценка состояния надежности системы*** на основе сбора, обработки и анализа текущих данных по эксплуатации ИС, оценки достигнутых уровней технической эффективности и надежности ИС с учетом предыдущих материалов доказательства надежности системы. Рассматриваются и анализируются все предлагаемые или сделанные изменения в ИС, включая изменения политик, правил и процедур. На данной стадии, также как и на других стадиях жизненного цикла, производится корректировка документа «Доказательство надежности системы». На этой стадии также определяются необходимые работы по модификации системы.

На ***тринадцатой стадии*** осуществляется ***модификация и переоснащение*** системы. Это естественный процесс для эксплуатации информационных систем, вызванный непрерывным совершенствованием и развитием информационных технологий, созданием более эффективных технологий, непрерывным совершенствованием вычислительных и телекоммуникационных средств, активным развитием математического обеспечения. В

связи с модификациями ИС возникает потребность в оценках их влияния на функциональную надежность, безотказность и другие характеристики RAMS. Создание новых версий программного обеспечения, изменение информационных технологий, изменения в оборудовании влечет за собой необходимость повтора жизненного цикла, начиная с корректировки концепции и завершая этапами эксплуатации, контроля структурной и функциональной надежности, а затем снова модификации системы и т.д.

Жизненный цикл информационной системы завершается стадией (***четырнадцатой*** согласно рассматриваемой нумерации) ***планирования и вывода из эксплуатации***, архивированием, ликвидацией или перемещением данных на другие системы, а также возможной утилизаций системы. Эти процедуры могут вызвать ущербы как экономического, так и социального характера. Отсюда необходимость в оценках угроз, рисков и безопасности вывода системы из эксплуатации.

Следует отметить, что не только на первых четырех, но и на всех последующих стадиях жизненного цикла производится *документирование* полученных результатов и *верификация* разработанных материалов, проектов, технологий, а также оценка использованных методов, способов, средств и оценка квалификации сотрудников.

III.1. 4. Программа обеспечения и доказательство надежности информационной системы

III.1.4.1.Политика и программа обеспечения надежности

Для создания надежной информационной системы следует разработать и согласовать Политику и Программу обеспечения надежности системы.

Программа обеспечения надежности (ПОН) – это документированный перечень запланированных по времени мероприятий, ресурсов и событий, направленных на внедрение организационной структуры, распределения ответственности за их распределение и исполнение для того, чтобы обеспечить необходимый уровень надежности ИС. Программа может содержать следующие разделы:

- Политика обеспечения надежности;
- Программа работ.

В *Политике* должны быть отражены цели и задачи обеспечения структурной и функциональной надежности ИС, основные принципы и подходы к решению этой задачи; принципы и подходы к управлению рисками др. Политика обеспечения надежности – это официально утвержденный руководством организации – заказчика документ, в котором отражены общие намерения и направления деятельности организации в части обеспечения структурной и функциональной надежности ИС. В этом документе должны указываться принципы и методы обеспечения надежности, используемые организацией (например, принцип минимизации издержек на обеспечение безошибочности выходных результатов, механизм перераспределения риска, метод стимулирования персонала, механизмы экономической ответственности предприятия и пр.). В документе «Политика обеспечения надежности» следует привести требования к комплексу организационно-технических мероприятий по обеспечению надежности функционирования ИС, распределению ответственности и полномочий, порядок взаимодействия между внутренними структурными подразделениями организации; порядок взаимодействия со сторонними организациями. Должен быть также изложен подход к управлению рисками, принятый организацией, принятый принцип допустимости риска, перечень потенциальных опасностей

(угроз) для ИС, пути и способы определения приоритетов при реализации мероприятий по обработке риска (в том числе, защитных мер), оценка остаточного риска.

В *Программе* должны содержаться методы, способы и технические решения по обеспечению безотказности, готовности, ремонтопригодности, функциональной безопасности составных элементов и системы в целом и безошибочности выполнения информационных процессов в разрабатываемой ИС (стадии 1-5 жизненного цикла, рис. III.I.2), организация работ по обеспечению надежности ИС на этапах производства (стадии 6-9 жизненного цикла, рис. III.I.2), а также меры по обеспечению надежности ИС в процессе ее эксплуатации (стадии 10-13 жизненного цикла, рис. III.I.2).

При изложении мероприятий по обеспечению надежности ИС в Программе должны быть отражены следующие главные положения:

- описание жизненного цикла надежности системы и задач по обеспечению структурной и функциональной надежности, которые должны быть выполнены в процессе жизненного цикла ИС;
- результаты анализа и оценки надежности ИС в течение жизненного цикла, включая следующие основные позиции:
  - идентификация и анализ опасностей;
  - оценка риска и текущее управление риском;
  - определение критериев допустимости риска;
  - принципы разработки требований к структурной и функциональной надежности ИС и доказательство их адекватности;
  - верификация составных объектов и валидация ИС в целом;
- результаты анализа показателей эксплуатации и технического обслуживания ИС;
- спецификация документации по надежности ИС;
- описание взаимосвязи с другими объектами и системами.

III.1.4.2. Доказательство надежности информационной системы

Доказательство надежности (Паспорт надежности) ИС – документированное подтверждение того, что система выполняет все заданные требования к структурной и функциональной надежности на всех этапах жизненного цикла. Этот документ представляет собой документированную совокупность доказательств, которые обеспечивают убедительные и достоверные аргументы, что система является адекватно надежной для данного применения в данной окружающей среде. Данный документ должен сопровождать систему на всех этапах жизненного цикла от начала разработки до завершения ее эксплуатации.

Цели доказательства надежности:

- проверка выполнения концепции обеспечения надежности системы;
- проверка соответствия ИС требованиям надежности;
- проверка соответствия показателей надежности ИС заданным нормам.

Документ «Доказательство надежности» аккумулирует всю совокупность материалов доказательного характера и отражает результаты работ по управлению качеством, управлению надежностью, проводимых на всех этапах жизненного цикла ИС. В Доказательстве надежности в письменной форме следует обосновать, что ИС соответствует заданным качественным и количественным требованиям.

Для обеспечения доказательств надежности необходимо:

- оформить в явном виде набор требований к системе;
- произвести содержательное доказательство (очевидность);
- обеспечить ряд аргументов надежности, которые связывают требования к системе с доказательствами;

• сделать понятными предположения (допущения) и суждения, лежащие в основе аргументов;

• обеспечить различные точки зрения и уровни детализации.

Чтобы выполнить указанные требования доказательства надежности должны быть структурированы следующим образом:

Требования к свойствам системы или некоторых подсистем.

Доказательство, которое используется как основа аргумента надежности. Оно может содержать либо *факты,* (например, факты, основанные на установленных научных принципах и предварительных исследованиях), либо *зависимые аргументы,* полученные из аргументов низшего уровня.

Аргумент, соединяющий доказательство с требованием, который может быть детерминированным, вероятностным или качественным.

Умозаключение – механизм, который обеспечивает правила преобразования, т.е. ***вывод*** для аргумента.

Доказательства надежности могут быть разделены на утверждения для различных свойств разных крупных узлов системы, например: *готовность, защита (от внешней атаки*), *функциональная корректность, ремонтопригодность, обеспечение надежности при отказе отдельных элементов* и др.

Перечень представленных выше атрибутов (характеристик) является только примером.

Отношение к надежности может иметь и другие атрибуты.

Рассмотрим процесс осуществления доказательств надежности. Многие проблемы при реализации приемлемого доказательства надежности являются результатом неконструктивной позиции, которая расценивает доказательства надежности как дополнительный аксессуар к системе (часто производимый после создания системы). На этом этапе часто обнаруживается, что «подгонка» поддерживающих доказательств надежности является и дорогим и трудоёмким процессом. Мы рекомендуем

другой подход, где доказательства надежности принимаются во внимание на протяжении всего проекта. В данном подходе рекомендуется:

• интегрировать доказательства надежности информационной системы и ее составных частей в проект и процесс его разработки.

• применять «поуровневые (слоистые)» доказательства надежности, то есть высокоуровневые доказательства надежности со вспомогательными доказательствами надежности для крупных подсистем.

• обеспечивать отслеживаемость между уровнями системы и подсистем.

• применять «проект для оценки», который принимает во внимание затраты и сложность доказательств надежности так же, как и проекта.

Доказательство надежности является «живым документом», который развивается в процессе всего жизненного цикла системы. Так как в нем регистрируются аргументы надежности, базовая структура должна оставаться в значительной степени подобной в течение долгого времени, но статус доказательства может измениться. Например, запланированные уровни тестового покрытия заменяются очевидностью результатов испытаний на достигнутом уровне охвата. На практике, конечно, доказательства надежности могут быть разделены на несколько документов (например, охватывающие конкретные подсистемы), а также они могут ссылаться на поддерживающие документы (например, документация проекта, отчеты анализа, отчеты об испытаниях и т.д.).

Доказательство надежности в общем случае должно содержать следующие основные разделы [11]:

– назначение, функции и характеристики ИС, внешнее оборудование, интерфейсы, режимы работы;

– архитектура системы, подсистем, взаимосвязь, ограничения проекта.

– описание окружающей среды.

– отчет о мерах по управлению качеством.

– запланированный подход обеспечения надежности (в частности, поддерживающие доказательства и методы анализа, например, RAMS).

– отчет о мерах по управлению структурной надежностью.

– отчет о мерах по управлению функциональной надежностью.

– доказательства структурной и функциональной надежности составных частей.

– долгосрочные требования поддержки. Техническое обслуживание системы.

– заключение по надежности.

Содержание документа «Доказательство надежности ИС» должно обновляться на всех этапах жизненного цикла системы по результатам текущих оценок состояния надежности системы.

Выводы, полученные из доказательства надежности, должны позволять судить о том, что:

- требования, предъявляемые к ИС, заданы корректно и в полном объеме;
- требования, предъявляемые к ИС, в полном объеме и корректно реализованы в технических решениях;
- технические решения не привносят дополнительных негативных свойств относительно первоначальных требований надежности;
- представленные доказательства обоснованы и достоверны.

Изложенный выше подход к доказательствам надежности информационной системы и ее составных частей делает главный акцент на утверждениях о поведении системы (то есть, функ-

циональное поведение и характеристики системы) и соответствующих аргументах для поддержки указанных утверждений. Идеи структурирования (использующие утверждения, аргументы и доказательства) весьма просты, но они должны обеспечить убедительные доказательства надежности, которые будут построены и которые будут понятными и прослеживаемыми. Для того, чтобы выполнить доказательства надежности, мы рекомендовали интегрирование доказательств надежности в процесс осуществления проекта.

Уровневое построение доказательств надежности позволяет этим доказательствам развиваться в течение долгого времени и помогает устанавливать требования на каждом уровне. Для больших проектов с субподрядчиками, этот «нисходящий» подход к доказательствам надежности помогает определять требования к подсистемам, и доказательство надежности подсистемы может быть сделано явным договорным требованием, которое будет поставлено субподрядчиком.

III.1.5. Избыточность и надежность

Если избыточность в системе возникла вопреки целям его разработчика, в результате незнания или неумения сделать более эффективную архитектуру информационной системы, то такую избыточность можно именовать как ***«дикая избыточность»***. Избыточность некоторых ресурсов может быть и опасной для системы – например, если у человека или животного слишком много мышц, то требуются и большие дополнительные усилия чтобы их «прокормить». Избыточное оборудование, омертвленные финансовые ресурсы, раздутые штаты предприятий вовсе не идут никому на пользу. Однако, есть и интересное отклонение от этого правила – большинство микрочипов, интеллектуальных устройств, начиная от смартфонов и заканчивая большими ком-

пьютерами, мозг человека и многих животных заведомо обладают избыточной памятью и «вычислительными мощностями». И здесь, по-видимому, «запас карман не тянет».

Если избыточность применительно к определенным функциям явилась постольку, поскольку введена избыточность по отношению к другим функциям, то такую избыточность следует именовать как ***«естественная избыточность».*** Например, в многоканальной информационной системе количество каналов обработки информации рассчитано либо на максимальную информационную нагрузку, либо на наиболее вероятную нагрузку. При невысокой информационной нагрузке свободные каналы могут использоваться как естественно избыточные ресурсы для контроля состояния работоспособности занятых обработкой информации каналов, или для оперативного резервирования отказавших каналов. Другой пример. В информационных системах реального времени каждый цикл обработки информации обычно имеет постоянную и равную длительность, рассчитанную на максимальную информационную нагрузку. В нормальных условиях функционирования, как правило, часть цикла обработки информации свободна. Это означает наличие определенного временного ресурса, который может использоваться для обеспечения надежности системы.

Часто избыточность вводится разработчиками умышленно для обеспечения надежности и функциональной безопасности, для обеспечения дальнейших улучшений системы, для универсализации ее функций, расширения возможностей ее адаптации, применения с различными целями и т.д. В таких случаях будем пользоваться термином «***искусственная избыточность».***

В процессе развития системы всегда происходит уменьшение избыточности как за счет уменьшения ненужных ресурсов, так и за счет превращения избыточности в ресурсы, которые необходимо использовать в системе для обеспече-

ния ее нормального функционирования. Эволюция избыточности системы может быть описана следующими этапами:

1. Система с высокой исходной избыточностью

2. Снижение избыточности системы – превращение избыточности в полезные ресурсы, обеспечивающие:

- повышение качества, надежности, безопасности и эффективности системы;
- увеличение количества и качества выполнения полезных функций системы;
- снижение вредных факторов;
- снижение материалоемкости системы;
- снижение трудоемкости изготовления, маркетинга и эксплуатации системы;
- повышение эффективности изготовления системы;
- повышение эффективности эксплуатации системы.

3. Возникновение противоречий между повышением качества и снижением затрат.

4. Разрешение противоречий между качеством и затратами путем создания систем нового поколения (имеющих, как правило, «запас избыточности», который потом будет превращаться в используемые ресурсы).

С позиции обеспечения надежности информационных систем различают следующие основные виды избыточности:

1. *Структурная избыточность.*
2. *Временная избыточность.*
3. *Информационная избыточность.*
4. *Функциональная избыточность.*
5. *Алгоритмическая избыточность.*
6. *Программная избыточность.*
7. *Многоуровневая избыточность.*

Структурная избыточность (Structure Redundancy или Hardware Redundancy) – наличие у объекта избыточных элемен-

тов, узлов, устройств, с помощью которых можно своевременно заменить отказавшие основные узлы или устройства, для предотвращения выхода из строя всего изделия, системы. Избыточные узлы или устройства могут работать параллельно с основными или могут быть подключены и использоваться в режиме ожидания или применяться с помощью иного, например, гибридного способа. Здесь мы уже затрагиваем технологию структурного резервирования, под которой понимаются методы, способы и/или технические решения по реализации структурной избыточности. Эти вопросы будут рассмотрены самостоятельно в главе III.2. Следует отметить, что возможность применения структурной избыточности находится в прямой зависимости от средств обнаружения отказавшего элемента системы. Проблема поиска процедуры замены «отказавшего» элемента в системе была осознана достаточно давно в военной науке и практике, свидетельство чему можно найти, например, у Макиавелли в книге [12]. Причем, процедура замены «отказавшего» элемента самым непосредственным образом связана со структурой системы. В качестве примеров здесь можно рассмотреть две основные военные структуры античности – легионы римлян и греческую фалангу.

Римский легион состоял из *гастатов* (авангард), *принципов* (основная часть или центр) и *триариев* (арьергард). *Гастаты* ставились в первую линию войск, образуя плотные сомкнутые ряды, за ними более редкими рядами располагались *принципы*; сзади всех находились *триарии*, построенные с еще большими промежутками, чем *принципы*. Такая структура позволяла римлянам иметь три независимые, достаточно компактные боевые единицы, обеспечивающие троекратное возобновление боевой линии. *Гастаты* вступали в бой только после отхода легковооруженных воинов, завязывающих бой с неприятелем. При неудаче *гастаты* отходили к *принципам*, занимая промежутки

в их рядах, и бой возобновлялся снова уже силами *принципов*. Если и вторая линия оказывалась разбитой, то она отступала к *триариям*, занимая широкие промежутки между их рядами, и все повторялось вновь. Такая структура обладала избыточностью, поэтому при отказе элементов (в данном случае, гибели воинов) они не заменялись новыми элементами, вводимыми в старую структуру. Наоборот, предполагалось возможным уничтожение части структуры, остатки которой занимали подготовленные места в новых структурах. Именно это обеспечивало надежность римских легионов в войне, ибо «поражение было почти невозможно, потому что счастье должно изменить тебе три раза подряд, а доблесть врага должна быть такова, чтобы трижды победить» [13].

У греческой фаланги был совсем иной способ возобновления боя и замены «отказавших» элементов. Она строилась большим числом шеренг со многими начальниками, но всегда располагалась в одну линию. Поэтому «отказавшие» элементы, т.е. выбывшие солдаты, непосредственно заменялись другими. Делалось это так. Греческая фаланга обычно состояла из 50 шеренг, но сражаться могли только первые 6. На место убитого или раненого бойца сразу же становился солдат из второй шеренги; его, в свою очередь, заменял стоявший за ним боец в третьей шеренге и т.д. Естественно, что постепенно задние шеренги убывали, потому что восполнять их было некому, но передние всегда оставались цельными. Надо сказать, что «эту фалангу можно было скорее истребить, чем прорвать, так как глубокий строй делал ее почти непроницаемой» [13]. Именно это обеспечивало надежность греческой фаланги. Римляне вначале использовали структуру греческих фаланг, но позднее вынуждены были от нее отказаться. Они поделили свои легионы на части – когорты и манипулы, ибо считали, что войско тем надежнее, чем больше в нем автономных частей, каждая из которых может действовать

самостоятельно. Здесь наблюдается еще один из принципов, необходимый при обеспечении надежности посредством структурной избыточности – модульность (раздельность).

Временная избыточность (Time Redundancy) заключается в использовании некоторой части производительности компьютера для контроля за исполнением программ и восстановления (рестарта) информационного процесса.

Различают ***естественную и искусственную*** временные избыточности. Рассмотрим пример *естественной* временной избыточности. В информационной системе реального времени обработка информации и выработка управляющих команд производятся в пределах очередного цикла управления. В связи с высокой скоростью обработки информации в современных компьютерах в пределах многих циклов обработки остаются неиспользованные отрезки времени, в течение которых компьютеры свободны от выполнения предусмотренных функциональных задач. Эти свободные отрезки времени можно использовать, в частности, для выполнения операций контроля. Их можно использовать для повторного выполнения отдельных операций обработки информации или управления и сравнения полученных результатов. При значительной величине свободных от выполнения функциональных задач отрезков времени создаются условия для реализации легкого рестарта информационного процесса.

Искусственная временная избыточность предполагает введение резерва времени для повторного выполнения предусмотренных функциональных задач и сравнения полученных результатов. При проектировании комплекса программ должен предусматриваться запас производительности, который будет использоваться для контроля и повышения надежности функционирования. Значение временной избыточности зависит от требований к надежности функционирования системы и со-

ставляет от 5-10% производительности компьютера в системе до трех- и четырехкратного дублирования производительности в мажоритарных вычислительных комплексах. При функционировании комплексов программ в реальном масштабе времени резерв времени для контроля и восстановления вычислительного процесса и информации не устанавливается заранее. Для диагностики искажений и операций восстановления требуется в общем случае небольшой интервал времени, который выделяется за счет резерва или сокращения времени решения функциональных задач.

Информационная избыточность (Information Redundancy) – некоторое повторение информации в той или иной форме, позволяющее восстанавливать исходные данные в случае каких-либо нарушений в работе системы. Информационная избыточность – это процесс дублирования части данных информационной системы для обеспечения надежности и контроля данных. Избыточность используется для обеспечения достоверности входных и промежуточных данных, которые в наибольшей степени влияют на нормальное функционирование комплекса программ, а также данных, которые требуют значительного времени восстановления. Для восстановления таких данных требуется их длительное накопление и обработка. Информационная избыточность способствует не только обнаружению искажения данных, но и устранению ошибок. Данные защищают двух- и трехкратным дублированием с соответствующей дисциплиной контроля сохранности и периодическим обновлением. Для менее важных данных информационная избыточность используется в виде помехозащитных кодов, позволяющих только обнаруживать искажения. Искаженные данные исключаются из обработки, и происходит их естественное обновление в процессе последующего функционирования. Многие данные информационной системы не защищаются вследствие их частого обновления или слабого

влияния возможных искажений на решение основных функциональных задач. Информационная избыточность – это увеличение количества информации с целью повышения надежности и достоверности передачи или хранения данных.

Информационная избыточность обычно реализуется поэтапно путем применения двух типов методов:

1. Методы обнаружения ошибок.
2. Методы коррекции ошибок.

Методы обнаружения ошибок позволяют только зафиксировать факт наличия ошибки. Обычно, они применяются в совокупности с программами восстановления данных. Например, если ошибка обнаружена при пересылке данных, то запрашивается повторная отправка. А если ошибка обнаружена в блоке сохраненной информации, то производится ее восстановление из резервной копии. Например, к методам обнаружения ошибок относится проверка на четность. В этом случае блоки информации при отправке или перед сохранением дополняются до четного количества единиц (в бинарном коде), а при приеме или считывании проверяется, является ли количество единиц четным. Также можно дополнять блоки информации до нечетного количества единиц. Однако, этот метод может фиксировать только нечетное количество ошибок. Если одновременно произошло четное количество сбоев, то такая ошибка не будет зафиксирована, так как количество единиц останется четным (или нечетным, если используется дополнение до нечетности).

Более надежным является метод контрольного суммирования. Простейшая реализация метода – это сложение байтов массива данных с переносом разряда переполнения. Есть более сложные и надежные методы контрольного суммирования, например, *контрольное суммирование CRC***,** базирующееся на методах циклического кодирования. Фактически CRC

является остатком от деления многочлена, соответствующего исходным данным, на порожденный многочлен фиксированной длины. В зависимости от длины порожденного многочлена, различают алгоритмы контрольного суммирования CRC8, CRC16, CRC32 и т.д.

К методам коррекции ошибок относится такое *кодирование информации*, которое позволяет не только обнаружить ошибку, но и исправить ее. Число ошибок, которое можно исправить, ограничено и зависит от вида применяемого кодирования. Естественно, коды коррекции можно использовать и для обнаружения ошибок, причем они могут обнаружить одновременных ошибок больше, чем в состоянии исправить. Например, к таким кодам относятся коды Хэмминга и сверточные коды.

Другим аспектом понятия *информационная избыточность* является сложившееся в теории информации представление о превышении количества информации, используемой для передачи или хранения сообщения, над его информационной энтропией. Здесь под информационной энтропией понимается мера неопределенности информации, неопределенность появления какого-либо символа. Данное представление появилось в теории электросвязи. Для администратора баз данных информационную энтропию следует интерпретировать немного по-другому: информационная энтропия всё также мера неопределенности информации, но, какая информационная неопределенность может возникнуть в базе данных? Любая база данных предназначена для хранения информации. И при проектировании базы данных следует учесть то, что какая-то информация может повторяться несколько раз. А каждая повторяющаяся запись – это занятое место на диске. То есть превышение количества информации необходимого для хранения данных.

Конечно, можно сказать, что сейчас, с появлением терабайтных накопителей отпала необходимость экономить место на

диске. Но избыточность информации ведет не только к тому, что требуется увеличение объема накопителей, но и приводит к аномалиям в базе данных. *Аномалии в базе данных* – это проблемы связанные с обработкой информации, а точнее с удалением данных из базы данных, с модификацией данных в таблице базы данных, а также проблемы с добавлением данных в базу данных, содержащую избыточные данные.

Остановимся на избыточности источника сообщений. Из энтропийных оценок источников сообщений, ясно, что их избыточность зависит от статических характеристик самих сообщений. Энтропия максимальна при равномерном появлении букв на любом месте сообщения. Для характеристики источника сообщений с различным алфавитом представляет интерес сравнение фактической энтропии источника $H(X)$ с максимально возможной $H_{max} = \log M$, где M – количество различных букв в алфавите.. В этом смысле введено понятие избыточности источника сообщений или избыточности алфавита как $R=(H_{max}–H(X))/H_{max}$, Избыточность источника R показывает, насколько хорошо используются буквы в данном источнике. Чем меньше избыточность источника информации R, тем большее количество информации вырабатывается источником на одну букву. Однако, не всегда необходимо стремиться к $R = 0$. С повышением избыточности повышается помехоустойчивость (надежность) источника. Выяснение количества избыточности важно потому, что мы должны вводить ее разумно, чтобы получить максимальный эффект помехозащищенности, а не полагаться на стихию. Например, избыточность любого языка оказывается порядка (50-70)%, то есть, если бы все буквы имели одинаковую вероятность использования и можно было бы использовать любые комбинации букв, то среднюю длину слова можно было бы значительно уменьшить. Однако разбираться в этой записи было

бы значительно труднее, особенно при наличии ошибок (лектора или студента).

Современные системы связи построены без учета ограничений, существующих в языке, а поэтому не достаточно эффективны, так как они приспособлены для передачи равновероятных букв алфавита, которые могут следовать друг за другом в любых комбинациях.

Функциональная избыточность (Functional Redundancy)**.** Этот вид избыточности учитывает возможность проведения одной и той же работы различными средствами. Например, в состав операционной системы (ОС) может входить несколько типов мониторов (модулей супервизора, управляющих тем или другим видом ресурса), различные средства организации коммуникаций между вычислительными процессами. Наличие нескольких типов мониторов, нескольких систем управления файлами позволяет пользователям быстро и наиболее адекватно адаптировать ОС к определенной конфигурации вычислительной системы, обеспечивать максимально эффективную загрузку технических средств при решении конкретного класса задач, получать максимальную производительность при решении заданного класса задач.

Функциональная избыточность может быть реализована с помощью различных средств, отличающихся разными уровнями эффективности в проведении работы. Так, информационное управление безопасностью движения локомотива на участке пути может осуществляться с помощью специализированного локомотивного устройства безопасности или путем передачи команд машинисту с помощью мобильных средств связи. В первом случае достигается высокая гарантия безопасности движения. Во втором случае возможность возникновения опасной ситуации существенно выше. Вместе с тем, в случае отказа устройства безопасности сохраняется (хотя и с пониженной эффективнос-

тью) возможность осуществлять информационное управление безопасностью движения локомотива.

Мы рассмотрели примеры *искусственной* функциональной избыточности. Однако в самой природе большинства реализуемых в цифровой технике логических функций присутствует ***естественная*** функциональная избыточность. Например, в базовых логических элементах И и ИЛИ, таблицы истинности которых показаны на рис. III.1.3 а и рис. III.1.3 б соответственно, содержатся логическая избыточность по входным сигналам x_1 и x_2. Так, в случае элемента И при входном наборе **00** трансформация сигнала x_1 или x_2 в результате помехи не приведет к ошибке в выходном сигнале y**.** При входном наборе **01** или **10** трансформация сигнала x_2 в первом случае или сигнала x_1 во втором случае также не приведут к ошибке в выходном сигнале y**.** Это естественно, поскольку элемент И обладает функциональной избыточностью по входным сигналам x_1=0 или x_2=0. В свою очередь, элемент ИЛИ обладает избыточностью по входным сигналам x_1=1 или x_2=1

Элемент «И»

x_1	x_2	y
0	0	**0**
0	**1**	**0**
1	0	**0**
1	1	**1**

Элемент «ИЛИ»

x_1	x_2	y
0	0	**0**
0	1	**1**
1	**0**	**1**
1	1	**1**

Рис. III.1.3. Таблицы истинности для логических функций «И», «ИЛИ»

Функциональная избыточность составных логических элементов развивается в функциональную избыточность цифровых устройств, микроопераций, операций. Она может быть реализо-

вана на уровне макроопераций и алгоритмов. При глубоком развитии этой темы функциональная избыточность может служить эффективным инструментом для обеспечения функциональной надежности информационных систем.

Алгоритмическая избыточность (Algorithmic Redundancy)**.** Этот вид избыточности означает наличие в алгоритме дополнительных правил и предписаний сверх минимально необходимых. Мерой чувствительности алгоритма может являться погрешность вычислений. Результаты вычислений искажаются вследствие погрешностей:

а) исходных данных, трансформированными в ходе вычислений;

б) округления;

в) погрешностей метода;

г) погрешностей, обусловленных отказами, сбоями и ошибками в программе.

Для устранения искажений в результатах вычислений в каждом конкретном случае можно применять специальные алгоритмы, нечуствительные к различного рода нарушениям информационного процесса. Например, при решении задачи оптимального управления с помощью фильтра Калмана – Бьюси [14] возможно вследствие сбойной ошибки искажение экстраполированных параметров. Результатом этого будет в лучшем случае искажение результатов оценивания параметров траектории управления, а возможно и потеря траектории управления. Для исключения этого и других негативных и опасных явлений, которые могут возникнуть в процессе управления объектом вследствие сбойных ошибок, применяют следующую алгоритмическую избыточность. Наряду с модулем алгоритма, который реализует оптимальную фильтрацию, вводят модуль, в котором реализуется сглаживание параметров траектории управления методом наименьших квадратов на основе скользящей фикси-

рованной выборки. Эта выборка содержит результаты сглаживания (оценивания) параметров траектории в нескольких последних циклах обработки, полученных с помощью алгоритма оптимальной фильтрации. В случае обнаруженного искажения в текущем цикле обработки сглаженных или экстраполированных параметров, или новых данных производится автоматический переход на алгоритм управления на основе метода наименьших квадратов. По сохраненным в фиксированной выборке параметрам производится повторная экстраполяция параметров траектории, которая позволяет другим способом устранить обнаруженные искажения параметров траектории управления. Для обнаружения искажений параметров возможны различные способы, например, алгоритмический контроль путем стробирования параметров.

Данный пример интересен тем, что описанный способ обеспечения устойчивости к ошибкам может применяться и в других приложениях. Например, при обработке информации о движущемся объекте, в метрологии при обработке результатов измерений и др. Он характеризует *искусственную алгоритмическую избыточность*. В информационных системах довольно часто встречается *естественная алгоритмическая избыточность*. Таким свойством обладают алгоритмы обработки потоков данных, алгоритмы решения систем дифференциальных (алгебраических) уравнений, алгоритмы быстрого преобразования Фурье и многие другие. При реализации этих алгоритмов одиночные ошибки нейтрализуются безошибочными результатами параллельной обработки других входных данных, которые в совокупности с приемлемой погрешностью обеспечивают безошибочный выходной результат.

Программная избыточность (Program Redundancy) – наличие в программе процедур (операций) сверх минимально необходимых. Возможный вариант программной избыточности

состоит в том, чтобы периодически [15] запускать эталонные задачи и сравнивать полученный результат с предполагаемым. Существуют другие способы обеспечения надежности программ на основе их избыточности. Многие из них изложены в главе III.2. Существенное повышение функциональной надежности программ возможно на основе методов *многоверсионного программирования*. Рассмотрим эти методы. Но прежде всего, вспомним специфические особенности программ по отношению к аппаратуре информационных систем. Эти особенности описаны в работе [16] и сводятся к следующему:

- программа физически не стареет и не изнашивается. Возможно только моральное старение программы;
- программа не требует ремонта и профилактического обслуживания;
- программные ошибки есть результат создания и корректировки ПО. Они могут не проявляться, если входные данные не инициируют выполнение тех участков программ, в которых содержатся ошибки. В отличие от технических средств некоторые входные потоки данных могут привести к частым ошибкам, в то время как другие потоки могут вообще не привести к ошибкам;
- каждая программа является уникальным продуктом. Этот факт можно объяснить следующим образом. Во-первых, любые две копии программы абсолютно идентичны. Во-вторых, программа никогда не дает ухудшение характеристик без внешнего вмешательства. В-третьих, процесс отладки программы никогда не повторяется;
- любая корректировка программы (даже, если это относится только к устранению обнаруженной ошибки) приводит к новой версии программы, поскольку обязательно приводит к изменению хотя бы одного элемента объектного кода программы. Это важное обстоятельство исключает возможность использования

ранее имевшихся статистических данных и принципиально не позволяет производить статистическую оценку надежности программ;

▪ при исправлении обнаруженной ошибки программист может внести новые ошибки, которые могут со временем проявиться при определенном наборе входных данных, что отличается от последствий ремонта технических средств.

Из перечисленных особенностей программ по отношению к аппаратуре следует, что традиционное резервирование применительно к программам не имеет смысла. Действительно, при резервировании программы на одном вычислительном модуле информационной системы и сравнении полученных результатов проявившаяся в программе ошибка при определенном наборе входных данных приведет к одинаковым ошибкам в выходных результатах обеих абсолютно аналогичных программ и не будет замечена при сравнении результатов. Если применить другую тактику: реализовать только одну программу и контролировать ее с помощью эталонной задачи, а другую абсолютно аналогичную ей запускать только в случае несовпадения полученного результата с предполагаемым, то при одинаковых исходных данных реализация второй программы приведет к такому же несовпадению полученного результата с предполагаемым, как это имело место при реализации первой программы. Если еще изменить тактику и запускать вторую программу при следующем наборе входных данных, то возможно отсутствие ошибки и совпадение полученного результата с предполагаемым. Однако, такой же результат мог иметь место и при запуске первой программы при новом наборе входных данных. Приведенные рассуждения убеждают нас в том, что традиционное структурное резервирование не может быть распространено на программные средства. Здесь следует искать иной путь.

В работах [17, 18 и др.] для обеспечения надежности программ предложен принцип многоверсионного программирования. Цель: обнаружить и замаскировать остаточные ошибки проекта программного обеспечения в течение выполнения программ, чтобы предотвратить отказы системы, и продолжать работу с высокой надежностью. В многоверсионном программировании данная спецификация программы реализуется поразному N раз. Те же самые значения входных данных задаются N версиям, и результаты, произведенные N версиями, сравниваются. Если результат считается достоверным, то он передается вычислительному модулю в качестве выходных данных. N версий могут выполняться параллельно на отдельных модулях. Альтернативно все версии могут выполняться на одном вычислительном модуле. Выходные результаты подвергаются внутреннему голосованию. Различные стратегии голосования могут быть использованы на N версиях в зависимости от требований применения. Для критически важных информационных систем необходимо полное согласие всех N версий. Для других информационных систем может быть использована стратегия голосования путем простого большинства. Для случаев, где не имеется коллективного согласия, могут быть использованы вероятностные подходы, чтобы максимизировать шанс выбора правильного значения, например, взяв среднее значение, временно заморозив выходные данные до возвращения согласия, и т.д.

При дуальном программировании (если разрабатываются две версии программы) в случае обнаружения расхождения в результатах, необходимо определить по дополнительным критериям, какой результат правильный и отбросить другой результат. При N-версионном программировании правильный результат определяется по мажоритарному признаку, т.е. выбирается тот результат, который наблюдается в большинстве вариантов программы.

Многоуровневая избыточность

Многоуровневая избыточность – это комплексное применение нескольких видов избыточности. Так, возможно сочетание алгоритмической, программной, функциональной, и даже временной избыточности. Широко применяется сочетание структурной и информационной избыточности, сочетание структурной и программной избыточности. Наглядным примером введения многоуровневой избыточности в систему, для достижения отказоустойчивости, может послужить система контроля и управления авиалайнера Airbus 320 (fly-by-wire flight control system). В процессе функционирования системы управления, и обеспечения взаимосвязей между различными компонентами и контроля за последними, в Airbus 320 задействовано 5 различных независимых компьютеров. Система управления авиалайнером строилась из расчета, что обнаружение ошибок должно осуществляться как в аппаратной, так и в программной части системы. По этой причине, в процессе управления полетом, дополнительно задействовано два типа программного обеспечения, от двух независимых разработчиков.

Многоуровневая избыточность представляет собой мощный инструмент обеспечения надежности информационных систем. Проиллюстрируем это на следующем примере. Пусть информационная система выполняет задачу с вероятностью безотказной работы $P_з=1–g_з$ (для наглядности предполагаем, что время работы фиксировано). В системе предусмотрено 2 уровня защиты от отказов таким образом, что первый уровень защищает средства выполнения задачи с вероятностью β_1 и функционирует с вероятностью безотказной работы $P_1=1–g_1$. Второй уровень защищает как средства выполнения задачи, так и средства первого уровня защиты с вероятностью β_2, и функционирует с вероятностью безотказной работы $P_2=1–g_2$. При этом сам второй уровень работает без прикрытия средств защиты. Послед-

нее условие означает, что отказ средства защиты второго уровня расценивается как отказ системы при выполнении задачи. Тогда показатель безотказного выполнения задачи с двумя уровнями защиты ($P_{з2}$) определяется как сумма вероятностей следующих четырех благоприятных событий:

A – задача выполнена безотказно и средства и первого и второго уровня работали безотказно. Вероятность этого события равна $P(A)=P_з P_1 P_2$;

B – задача выполнена безотказно, возник отказ средства первого уровня защиты с вероятностью g_1. Средства второго уровня защиты работали безотказно и нейтрализовали отказ первого уровня защиты с вероятностью β_2. Вероятность этого события равна $P(B)=P_з g_1 P_2 \beta_2$;

C – при выполнении задачи возник отказ, однако он был нейтрализован работоспособными средствами защиты первого уровня с вероятностью β_1. При этом средства защиты второго уровня работали безотказно. Вероятность этого события равна $P(C)=(1-P_з)P_1 P_2 \beta_1$;

D – при выполнении задачи возник отказ. Отказали также средства защиты первого уровня. Средства защиты второго уровня оставались работоспособными. Отказ в работе системы нейтрализован средствами защиты второго уровня с вероятностью β_2. Вероятность этого события равна $P(D)=(1-P_з)g_1 \beta_1 P_2 \beta_2$.

Следовательно,

$P_{з2}=P(A)+P(B)+P(C)+P(D)=P_2(P_1+g_1\beta_2)(P_з+g_з\beta_1)$.

Предположим, что пользователя не удовлетворяет вероятность безотказного выполнения объектом задачи на уровне $P_з=0{,}9$. Для защиты от функциональных отказов на первом уровне защиты приняты эффективные меры, обеспечивающие высокий уровень обнаружения отказа и его устранения с результирующей вероятностью $\beta_1=0{,}9$. При этом разветвленные средства защиты первого уровня имеют недостаточную надеж-

ность P_1=0,95. Для защиты от функциональных отказов средств защиты первого уровня (СЗ 1) введен второй уровень защиты СЗ 2, который хотя и обеспечивает недостаточно высокую вероятность адаптации к отказам СЗ 1 β_2=0,8, но функционирует надежно (P_2=0,99).

При указанных условиях показатель безотказного выполнения задачи оценивается как $P_{з2}=P_2(P_1+g_1\beta_2)(P_з+g_з\beta_1)=0{,}97$.

Заметим, что, если в условиях данной задачи ограничиться только одним уровнем защиты, то показатель безотказного выполнения задачи на основании (1.2) равен $P_{з1}=1-g_1-g_3(1-\beta_1)=0{,}94$ т.е. $P_{з1}<P_{з2}$, из чего следует целесообразность введения второго уровня защиты.

Из рассмотренного примера следует, что *излишняя избыточность может даже навредить – может привести к снижению надежности*. Напомним, что избыточные программно – аппаратные средства не являются идеально надежными, они отказывают, сбоят, вызывают ошибки в результатах функционирования. *Поэтому важно определить* ***условие, при котором многоуровневая избыточность повышает надежность системы.***

При n уровнях защиты показатель безотказного выполнения задачи определяется выражением:

$$P_{зп} = P_n\left(P_{n-1}+g_{n-1}\beta_n\right)\left(P_{n-2}+g_{n-2}\beta_{n-1}\right)\ldots$$

$$\ldots\left(P_1+g_1\beta_2\right)\left(P_з+g_з\beta_1\right)=\left(1-g_n\right)\prod_{i=0}^{n-1}\left(P_i+g_i\beta_{i+1}\right), \qquad (1.5)$$

где $P_0=P_з$; $g_0=g_з$.

Очевидно, что на каждом уровне защиты должно выполняться следующее условие, которое диктуется неравенством (1.4) (см. постулат 4 в пункте III.1.2 и его формульное представление),

$$g_i < g_{i-1}\beta_i \text{ где } i = \overline{1,n}. \quad (1.6)$$

На основании условия (1.6) и выражения (1.5) получим новое неравенство

$$g_i < g_з \prod_{j=1}^{i} \beta_j, \text{ где } i \geq j,\ g_0 = g_з, \quad (1.7)$$

Теорема. *Если выполняется неравенство (1.7), то допустимо введение в объект дополнительных программно-аппаратных средств, необходимых для создания нескольких уровней защиты (контроля, диагностики, исправления ошибок), что приводит не к снижению, а к повышению вероятности безотказного выполнения задачи* [4].

Доказательство. Согласно (1.5)

$$p_{зп} = (1 - g_n)\prod_{i=0}^{n-1}\left[1 - g_i\left(1 - \beta_{i+1}\right)\right].$$

Из неравенства (1.7) имеем

$$g_n < g_з \prod_{i=1}^{n} \beta_i \text{ и } g_i < g_з \prod_{j=1}^{i} \beta_j.$$

После подстановки этих выражений в предыдущую формулу получим

$$P_{зп} > (1 - g_з \prod_{i=1}^{n} \beta_i)\prod_{i=0}^{n-1}\left[1 - g_з\left(1 - \beta_{i+1}\right)\prod_{j=1}^{i}\beta_j\right].$$

Обозначим

$$g_з(1 - \beta_1)\prod_{j=1}^{i}\beta_j = A_i, \quad \text{где } A_i < 1.$$

При $A_i<1$ ряд $\prod_{i=0}^{n-1}\left[1 - A_i\right] = 1 - \sum_i A_i + \sum_{ij} A_{ij} - \ldots + (-1)^{n-1}\prod_{i=0}^{n-1} A_i$ является знакопеременным.

В соответствии с признаком Лейбница для знакопеременных рядов данный ряд сходится, поскольку его члены стремятся к нулю, а остаток ряда R3 имеет тот же положительный знак, что и первый отбрасываемый член $\sum_{ij} A_{ij}$ и меньше его по абсолютному значению. Тогда с погрешностью, не превышающей второго порядка малости, справедливо выражение

$$P_{зп} > (1 - g_з \prod_{i=1}^{n} \beta_i) \cdot \left[1 - g_з \sum_{i=0}^{n-1} (1 - \beta_{i+1}) \prod_{j=1}^{i} \beta_j \right], \qquad (1.8)$$

причем эта оценка вероятности $P_{зп}$ занижена.

Теорема справедлива, если выполняется неравенство $P_{зп} > P_з$, или $g_{зп} < g_з$, которое означает, что вероятность безотказного выполнения задачи системой с n уровнями защиты выше вероятности безотказного выполнения задачи без защиты.

$$g_з \left[\prod_{i=1}^{n} \beta_i + \sum_{i=0}^{n-1} (1 - \beta_{i+1}) \prod_{j=1}^{i} \beta_j - g_3^2 \sum_{i=0}^{n-1} (1 - \beta_{i+1}) \prod_{j=1}^{i} \beta_j \prod_{i=1}^{n} \beta_i \right] < g_з. \quad (1.9)$$

Для этого достаточно убедиться, что выражение в квадратных скобках меньше 1. В квадратных скобках выражения (1.9) имеет место следующий любопытный результат :

$$\left[\prod_{i=1}^{n} \beta_i + \sum_{i=0}^{n-1} (1 - \beta_{i+1}) \prod_{j=1}^{i} \beta_j \right] =$$

$$= \prod_{i=1}^{n} \beta_i + \beta_1 - \beta_1\beta_2 + \beta_1\beta_2 - \beta_1\beta_2\beta_3 + \ldots - \prod_{i=1}^{n} \beta_i = \beta_1.$$

Кроме того, в квадратных скобках формулы (1.9) имеет место следующее неравенство: $\beta_1 >> g_3^2 \sum_{i=0}^{n-1} (1 - \beta_{i+1}) \prod_{j=1}^{i} \beta_j \prod_{i=1}^{n} \beta_i$. Из этого

неравенства следует, что выражение в квадратных скобках и положительно и меньше 1. Теорема доказана.

Таким образом, только при выполнении неравенства (1.7) введение в информационную систему дополнительных программно – аппаратных средств, необходимых для создания нескольких уровней защиты (контроля, диагностики, исправления ошибок), приводит не к снижению, а к повышению вероятности безотказного выполнения задачи.

III.1.6. Отказоустойчивость

Под ***отказоустойчивостью технических систем обычно подразумевают их способность выполнять предусмотренные функции в реальных условиях эксплуатации при наличии внутренних возмущающих воздействий (отказов и/или сбоев составных технических средств, программных ошибок).*** Отказоустойчивость – это свойство технической системы сохранять свою работоспособность после отказа одного или нескольких составных компонентов. Здесь речь идет об *аппаратной отказоустойчивости*. *Программная отказоустойчивость* имеет место тогда, когда система имеет возможность сохранять работоспособность после возникновения одной или нескольких программных ошибок.

Отказоустойчивость устройства или системы основывается на наличии избыточности. Здесь уместно различать три типа избыточности: *искусственную, естественную, гибридную.* Искусственная избыточность имеет место тогда, когда в объект (устройство или в систему) для обеспечения отказоустойчивости вводят дополнительные ресурсы сверх минимально необходимых. При этом дополнительные ресурсы расходуются на парирование отказов компонентов. Это обеспечивает функционирование объекта с постоянным максимальным уровнем

эффективности. После того, как все дополнительные ресурсы исчерпаны, объект уже не является отказоустойчивым. Данный тип отказоустойчивости наиболее затратный, поскольку связан с поддержкой эффективности объекта на максимальном уровне. Простейший пример искусственной избыточности – структурное резервирование.

Естественная избыточность характерна для многофункциональных или многоканальных систем. Объект с этим видом избыточности обладает свойством *отказоустойчивости на основе постепенной деградации*. Слово деградация заимствовано из польского языка (в свою очередь, degradacja – от латинского degradatio «разжалование, постепенное понижение»). Деградация – постепенное ухудшение, утрата ценных свойств и качеств объекта. Это постепенное сокращение возможностей или постепенный выход из работы. Иначе говоря, это плавное снижение эффективности объекта. Например, деградация в телекоммуникациях – потеря качества сигнала. Деградация в системах массового обслуживания – это снижение эффективности обслуживания заявок (пропускной способности обслуживания заявок) вследствие отказов каналов обслуживания. Деградация в многофункциональных системах – это снижение эффективности вследствие невыполнения отдельных функций из-за отказов компонент объекта. Для любых типов объектов *обеспечение отказоустойчивости на основе постепенной деградации оправдано* тогда *и только тогда, когда эффективность объекта снижается до уровня не менее допустимого.* В противном случае имеет место разрушение объекта, т.е продолжение его работы с неприемлемым постоянно понижающимся уровнем эффективности. Данный тип обеспечения отказоустойчивости наименее затратный, однако, часто не применим из-за существующих для объекта требований по сохранению эффективности.

Наиболее рациональный путь обеспечения отказоустойчивости основан на применении *гибридной избыточности*. В этом случае можно минимизировать количество дополнительных ресурсов и вместе с этим повысить порог допустимого уровня снижения эффективности объекта.

Реализация отказоустойчивости в информационных системах может осуществляться на основе обеспечения их *наблюдаемости* и *управляемости*.

III.1.6.1. Наблюдаемость информационных систем

Под наблюдаемостью ИС будем подразумевать ее способность своевременно обнаруживать нарушения информационных процессов, вызванных отказами и сбоями технических средств, ошибками в программах, ошибками операторов и во входной информации.

Наблюдаемость информационных систем достигается в результате выполнения этапов обнаружения факта неисправности, локализации неисправности, классификации неисправности и, затем, обнаружения ее местонахождения.

I. Обнаружение факта неисправности может осуществляться как в рабочем, так и в нерабочем состоянии. Обнаружение в рабочем состоянии происходит в информационной системе (ИС) одновременно с выполнением функциональных алгоритмов и обеспечивает способность выявления неисправностей в режиме реального времени. При обнаружении в нерабочем состоянии устройство проходит тестирование и, естественно, устройство неспособно выполнять полезную работу. Следовательно, обнаружение в нерабочем состоянии обеспечивает сигнализацию о неисправности устройства или программы ИС перед выполнением задачи и в начальный момент ее выполнения, но не в течение всего периода задачи.

При обнаружении неисправности подразумевается, что зарегистрирован момент ее возникновения. В информационных

системах применяются различные аппаратные и программные средства обнаружения факта неисправности. Обеспечение качества этих средств – многокритериальная задача. Она формулируется следующим образом. Достижение минимального времени обнаружения факта неисправности при максимальных уровнях достоверности и полноты контроля в условиях заданных ограничений по объему оборудования, габаритам, весу и, особенно, по дополнительному количеству аппаратных отказов и программных ошибок, привнесенных в ИС средствами контроля. Данная задача пока еще не решена, не разработана также научно обоснованная техническая политика в области наблюдаемости информационных систем. Вместе с тем, заметны определенные тенденции и в применении известных аппаратных и программных способов обнаружения фактов неисправностей в информационных системах. Среди аппаратных способов наиболее широкое распространение получили:

Ошибкообнаруживающие коды (коды с проверкой на четность в устройствах управления, обработки, хранения и передачи информации, коды Хэмминга, циклические и итеративные коды для контроля хранения и передачи информации, равновесные коды для контроля управляющих автоматов);

– *дублирование технических средств* с целью обеспечения их параллельной работы и сравнения полученных результатов;

– *самотестирование микропроцессоров* на основе аппаратно-микропрограммных средств.

Аппаратные способы контроля во многих случаях не обеспечивают обнаружения большого многообразия ошибок, возникающих в ходе работы информационных систем. Основные из них [19]:

– зацикливания и остановы исполнения программ;

– самоблокировки (клинчи) исполнения программ в ИС;

– перегрузки ИС по пропускной способности;

– нарушение последовательностей вызова подпрограмм;

– ошибки взаимного прерывания программ;

– искажения и потери накопленной информации о состоянии внешних абонентов.

В целях оперативного обнаружения факта подобных ошибок в рабочем режиме ИС применяются программные методы логического контроля и охранного таймера. Эти методы не разрушают информацию, накопленную в процессе функционирования внешних абонентов. Они допускают проверку в пределах небольших квантов времени. В отличие от этих методов контроля традиционные методы двойного или тройного счета требуют больших затрат производительности и памяти ИС. Поэтому методы повторного счета практически не применяются для оперативного обнаружения фактов ошибок в ИС.

Метод охранных таймеров основывается на программной реализации таймеров совместно со счетчиками относительного или текущего времени. Рассмотрим два характерных способа реализации данного метода: без прерывания программ и с прерыванием выполняемых программ.

Первый способ применяется для контроля программ построенных по модульному принципу. В этом случае перед включением *i*-го программного модуля на счетчике относительного времени устанавливается предельно допустимое время его реализации, рассчитанное на выполнение наиболее длинного маршрута модуля. В процессе счета показание счетчика равномерно убывает, что обеспечивается подачей на этот счетчик сигналов точного времени. При достижении нулевого времени на счетчике вырабатывается сигнал ошибки. Если же при полной реализации *i*-го программного модуля охранный таймер не выработал сигнал ошибки, то на счетчике вырабатывается новая предельная длительность реализации, рассчитанная на выполнение наиболее длинного маршрута (i+1)-го программного модуля.

Второй способ заключается в сравнении текущего времени таймера с показанием допустимого времени окончания работы подпрограммы, или допустимого времени ожидания заявки в очереди, или длительности запаздывания включения периодических программ и т. д.

В целом метод охранных таймеров эффективен для решения задач обнаружения фактов зацикливания, остановов и самоблокировок исполнения программ, а также перегрузки ИС по пропускной способности.

Для обнаружения нарушений последовательностей вызова программ кроме метода охранных таймеров применяется также контроль *ключевых кодов,* при котором формируется таблица ключевых кодов и количества включений каждой программы.

Логический контроль основывается на избыточности исходной, промежуточной и результирующей информации. Он включает в себя ряд способов:

– *проверки на логические несоответствия*;

– *проверки по предельным значениям вычисляемых параметров.* Они состоят в определении ряда условий, которые характеризуются граничными значениями контролируемого параметра. Например, значения исходных данных, промежуточных и выходных результатов не могут выходить за пределы разрядной сетки вычислительного модуля (ВМ), в противном случае фиксируется ошибка; или, вычисляемое значение вероятности p какого-либо события должно находиться в пределах $0 \leq p \leq 1$. Одним из частных случаев этого способа следует считать *способ статического контроля стробированием, в* котором контролируемые параметры сравниваются с допустимыми значениями. Допустимые значения определяются заранее или рассчитываются в ходе выполнения вычислительного процесса. Они, как правило, меньше предельных. С помощью

данного способа могут контролироваться не только значения переменных, но и скорости их применения в течение определенного кванта времени;

– *динамический контроль стробированием* (контроль гладкости изменения переменных с течением времени);

– *проверки контрольных соотношений.* Существует множество вариантов этого способа, которые можно разделить па две группы. Первая группа вариантов основывается на *сравнении с эталоном,* т, е. сравнении рассчитанных значений переменных или массивов с их заранее заданными (эталонными значениями). Сравнение с эталоном может осуществляться не только применительно к значениям переменных, но и к результатам операций, выполняемых над их адресами. Вторая группа вариантов способа контрольных соотношений основывается на вычислении *контрольных функций* [19]. При этом наряду с вычисляемой функцией по иной программе определяется другая функция, находящаяся с основной вычисляемой функцией в контрольных соотношениях. Простейшим примером применения метода контрольных соотношений является вычисление функций $\sin x$ и $\cos x$ по отдельным программам. Контрольным соотношением в данном случае будет $\sin^2 x + \cos^2 x = 1$.

Контроль с использованием избыточных переменных. Суть способа состоит во введении избыточного преобразования в исходный алгоритм, а также во введении избыточных переменных, которые удовлетворяют определенным контрольным условиям.

Способы логического контроля применяются в управляющей ИС для обнаружения фактов искажений и потерь накопленной информации о состоянии внешних абонентов, для оперативного контроля состояния и изменения данных в памяти информационной системы, для обнаружения фактов ошибок, возникающих при выполнении функциональных программ.

II. Локализация неисправности заключается в ограничении ее распространения в конкретной подсистеме и предотвращении порчи других системных составляющих. Ограничение сферы влияния неисправности должно осуществляться как в пространстве, так и во времени. Типичные примеры распространения неисправностей в пространстве: отказы или ошибки аппаратуры или программ ВМ вызывают потери или искажения результатов работы одних ВМ и искажают данные других ВМ, а в итоге приводят к ошибкам в результатах решения задач управления; некоторые неисправности вызывают искажение предусмотренных и разработанных циклов, а также образование непредусмотренных, ложных циклов, что в итоге может привести к блокировке и прекращению решения задач управления. Распространение неисправностей во времени обычно имеет место при решении задач, реализуемых с помощью итерационных вычислительных процессов. В этих задачах возможные небольшие искажения результатов отдельных итераций через несколько шагов (итераций) усиливаются и могут привести к срывам выполнения задач.

Локализация неисправностей осуществляется как в текущем состоянии, так и перед началом решения задач управления (предстартовые проверки и ограничения).

В рабочем состоянии ИС возможны следующие способы локализации неисправностей:

– *задержка к выдаче результатов* выполнения функций на время анализа величины их искажений и возможности распространения. В зависимости от результатов анализа в управляющих ИС могут быть приняты оперативные меры для ликвидации их последствий, такие как: игнорирование обнаруженной ошибки вследствие слабого влияния на процесс управления; исключение полученных результатов из последующей обработки вследствие их искаженности и или трудности восстановления вычислительного процесса;

– *кратковременное прекращение решения функциональной задачи* до момента обновления исходных данных.

Для предварительной локализации неисправностей (предстартовых проверки ограничений) решаются следующие задачи:

– *контроль сохранности программ.* Он обеспечивает проверку соответствия записи программ в памяти ИС исходному эталону. При этом обычно применяется логический контроль;

– *контроль исходных данных режима начального пуска.* Предназначен для проверки полноты состава и правильности значений исходных переменных, а также для проверки правильности времени начала функционирования, синхронности и синфазности показаний счетчиков реального времени всех компонент ИС. Первая задача решается с помощью логического контроля или путем сравнения дублированных файлов данных. Вторая задача решается с помощью охранных таймеров;

– *контрольная задача* предназначена для проверки функционирования комплекса программ по фиксированным исходным данным путем сравнения результатов с заранее рассчитанными эталонными значениями;

– *контроль обмена данными с внешними абонентами* предназначен для оценки готовности интерфейсных каналов ИС и каналов передачи данных к выполнению задач управления. Реализуется, главным образом, путем введения в системы передачи данных решающей, обратной связи в сочетании с блокировкой, переспросом и повторением сообщений, а также с ошибкообнаруживающими кодами.

При наличии достаточного времени до начала функционирования системы в ИС могут выполняться диагностические тесты для локализации неисправностей в аппаратуре и программах ВС.

III. Классификация неисправности предназначена, главным образом, для выявления отказа или сбойной ошибки со-

ставного устройства системы. Классификация неисправности играет важную роль в выборе стратегии восстановления вычислительного процесса..

Для решения данной задачи целесообразно, во-первых, установить, что длительность неисправности больше или меньше допустимого времени перерыва в работе системы, и, во-вторых, проанализировать отсутствие выходной информации или наличие искаженной информации на выходе устройства. Сбойные ошибки носят кратковременный характер, их длительность обычно много меньше допустимого интервала перерыва в выполнении заданной функции ИС. В том случае, когда длительность неисправности больше допустимого времени, существуют серьезные основания считать, что устройство отказало. В этом можно окончательно убедиться, если будет установлено, что на выходе устройства отсутствуют выходные результаты (такой отказ назовем тяжелым). В отличие от тяжелого возможен так называемый византийский отказ [20], означающий наличие ложных результатов на выходе устройства. При этом имеют место различные неверные результаты при одних и тех же исходных данных и программах.

Таким образом, если факт неисправности устройства обнаружен и устройство, по всей видимости, выдает ложные результаты, то для окончательного принятия решения об отказе или сбое достаточно реализовать механизм контрольной точки, что, в свою очередь, позволит устранить сбойную ошибку устройства (если неисправность такого типа).

Под контрольной точкой подразумевается некоторая точка в вычислительном процессе, для которой сформированы в локальной или общей памяти ИС начальные условия выполнения очередной части процесса и входная информация для него, полученная по результатам выполнения предыдущих частей. Таким образом, вычислительный процесс разделен контрольными

точками на составные части, Начальными условиями являются, например, начальный адрес очередной части программы и исходное состояние схемы тактирования, задающей длительность исполнения этой части программы. Возврат к контрольной точке производится по начальным условиям очередной части программы.

Если содержимое контрольной точки не испорчено, то данный способ пригоден для классификации неисправностей. Он заключается в повторном счете с контрольной точки и контроле состояния работоспособности устройства при повторном счете. Если в данном случае неисправность не обнаружена, то можно полагать, что имевшая место при первом счете сбойная ошибка ликвидирована. В противном случае следует фиксировать факт отказа устройства. В целом этапы локализации и классификации неисправностей позволяют не только ограничивать сферу влияния неисправностей и определять тип ошибок вычислительных средств, но и в отдельных случаях устранять сбойные ошибки.

IV. Обнаружение местонахождения неисправностей предназначено для выявления отказавшего типового элемента замены, модуля, блока, программного модуля или блока программных модулей. Требуемая глубина обнаружения неисправностей зависит от принятых принципов обеспечения отказоустойчивости каждой конкретной ИС.

Этот этап наблюдаемости ИС реализуется, главным образом, с помощью хорошо разработанных средств диагностирования их аппаратуры и программ [21-24].

Местонахождение и последующее исправление собственных программных ошибок в ИС в основном производится в интервалах времени между решениями задач управления. С этой целью применяются методы структурного и функционального тестирования программ.

Структурные методы тестирования программ заключаются в выборе набора путей в структуре программы, удовлетворяющего некоторым критериям тестирования, например, в отношении покрытия графа сформированным набором путей. Вследствие ограничения времени ожидания начала управления естественно применять выборочное тестирование программ в отдельных точках пространства исходных данных. Разработаны способы числового и символического тестирования. С помощью *числового тестирования* проверяют правильность числовых результатов работы программ при специально подобранных тестовых наборах входных переменных. *Символическое тестирование* заключается в выполнении процедур, основанных на символических входах и выходах, причем, для различных путей программы имеют место различные входы и выходы. Символическое тестирование снимает многие ограничения в отношении пространства исходных данных, присущие числовому тестированию.

Функциональные методы тестирования программ состоят в непосредственной проверке соответствия выполняемых программой функций поставленным требованиям. При этом необходимо решить вопросы выбора тестов и оценки правильности результата прохождения каждого теста. Применяются два подхода выборов тестов: *по содержательному признаку* и путем статистического моделирования входных данных *(стохастический выбор тестов)*. Первый подход трудно формализуется, зато при моделировании случайных исходных данных можно выбрать такие законы распределения этих данных, которые наиболее полно отражают содержание выполняемой программы и ее связи с другими программами и внешними абонентами.

Проверка правильности результатов функционального тестирования осуществляется с помощью методов охранного таймера и логического контроля.

В целом проблема оперативного обнаружения собственных программных ошибок в ИС требует дальнейшей проработки и, по-видимому, на основе иных нетрадиционных подходов.

III.1.6.2. Управляемость информационных систем

Под управляемостью ИС будем подразумевать ее способность своевременно адаптироваться к возникшим функциональным и структурным нарушениям.

Управляемость ИС можно рассматривать как реакцию на обнаруженный отказ. Она достигается путем выполнения этапов реконфигурации ИС, полного восстановления информационного процесса, восстановления аппаратуры ИС и, наконец, реинтеграции.

I. Реконфигурация предназначена для автоматического изменения архитектуры ИС с целью обеспечения ее работоспособного функционирования. Задачи реконфигурации: сохранение прежних возможностей системы в условиях отказов аппаратных или программных компонентов, либо выделение минимального приоритетного множества функций необходимого для продолжения работы в допустимых пределах сокращения возможностей системы. Первая задача решается с помощью специальных предусмотренных в архитектуре ИС резервных аппаратных и программных средств. При этом изолируются отказавшие компоненты системы и заменяются исправными резервными.

Вторая задача решается па основе имеющейся и модульной ИС естественной аппаратурной и программной избыточности. В зависимости от типов выполняемых вычислительных процессов возможны два варианта использования естественной избыточности. Первый вариант – исключение из обработки задачи с низшим приоритетом на время восстановления отказа компонента и использование освобожденных ресурсов аппаратуры для замены отказавшего компонента. В отношении отказа

программного компонента возможна его замена другой версией решения этой же задачи, если такая имеется. Второй вариант – исключение из архитектуры ИС отказавшего компонента без замены. Такой подход носит название постепенной (элегантной) деградации [25].

При реконфигурации ИС выполняются следующие операции: 1) отыскание исправного резервного ВМ из защитной среды; 2) исключение из функциональной структуры отказавшего ВМ; 3) включение в состав вычислительной структуры функциональной задачи резервного ВМ; 4) загрузка резервного ВМ программами и данными основного ВМ.

При решении ряда задач обработки информации и управления допускается постепенная деградация модульной управляющей ИС при отказах составных ВМ. Так, например, при решении задачи сглаживания и экстраполяции параметров движущихся объектов с помощью алгоритмов фиксированной выборки каждое единичное измерение может обрабатываться отдельным ВМ. В случае отказа этого модуля значение соответствующего измерения часто исключается из обработки [26] поскольку взвешенные суммы результатов других измерений будут в допустимых пределах отличаться от тех значений, которые имели бы место при безотказной работе всех ВМ использованных для решения задачи. Чем больше объем фиксированной выборки, т.е. чем больше ВМ используются для решения указанной задачи, тем больше естественная избыточность сформированной вычислительной структуры и тем больше возможностей для сохранения отказоустойчивости системы на основе ее постепенной деградации. Естественно, что при этом ухудшаются характеристики ВС вследствие исключения без замены отказавших ВМ.

В отказоустойчивых системах с постепенной деградацией вычислительных структур осуществляется также реконфигурация системных программных средств. Так может осуществляться

перемещение ОС в область памяти, где нет отказов, переотображение адресов памяти с целью исключения отказавшего модуля. Для сохранения работоспособности применяются альтернативные алгоритмы выполнения операций на другой аппаратуре при выходе из строя основной аппаратуры в составе ВМ. Так, например, при неисправности в блоке ускоренного умножения операция умножения может выполняться с помощью альтернативной микропрограммы и а основном сумматоре [25].

Широко применяется метод постепенной деградации для сохранения отказоустойчивости оперативной, постоянной памяти ВМ и общей памяти ИС.

Реконфигурация любых элементов системы выполняется специальными переключателями, под которыми подразумеваются аппаратные и программные управляющие средства. В качестве аппаратных переключателей реконфигурации используются составные процессоры пли вычислительные модули системы, а также специально синтезированные для этих целей устройства управления [25]. Программные средства реконфигурации должны быть составными частями общей или распределенной операционной системы. При построении переключателя реконфигурации ИС следует предусмотреть возможность появления в его работе ошибок типа «обрыв» или «короткое замыкание». Ошибки первого типа возникают в системе в случае отказов самих переключателей реконфигурации. Они могут привести к частичному или полному нарушению всей системы обеспечения отказоустойчивости ИС. Поэтому следует обратить особое внимание на достижение высоких уровней отказоустойчивости переключателей реконфигурации. Ошибки второго типа («короткое замыкание») приводят к несанкционированной активизации механизма переключения вследствие сбойных или собственных программных ошибок, как переключателя, так и сопряженных с ним устройств. Существуют различные способы устранения

указанных ошибок в работе переключателей реконфигурации. Так, например, в системе Pluribus доступ к переключателям реконфигурации контролируется механизмом паролей [27], а в системе FTMP предусмотрено трехкратное выполнение программ реконфигурации.

II. Полное восстановление вычислительного процесса непосредственно следует за этапом реконфигурации. *Восстановление вычислительного или информационного процесса* можно рассматривать в некоторой аналогии с восстановлением работоспособного состояния человека, которое, в свою очередь, достижимо только в том случае, если устранены нарушения функций организма, имеющих отношение к выполняемой им работе. Естественно и проблему восстановления вычислительного процесса следует рассматривать с позиций единой системы «аппаратура – алгоритм». Этот подход, по – видимому, впервые был изложен в работе [28] и определяет необходимость исправления нарушенных функций в системе, а не только устранение обнаруженных отказов, сбойных или программных ошибок. С этих позиций любая ИС представляет собой кибернетическую систему с присущими ей свойствами наблюдаемости и управляемости.

Для восстановления правильного функционирования системы возможно либо продолжение выполнения вычислительного процесса (ВП) после устранения возможных последствий ошибки – метод FER (Foward Error Recovery), либо безоговорочный возврат ВП к некоторому предыдущему состоянию, для которого есть основания предполагать, что оно не подверглось влиянию ошибки – метод BER (Backward Error Recovery).

Метод FER применен, в частности, в системе ESS, предназначенной для обслуживания телефонной сети. На уровне программного обеспечения этот метод позволяет восстановить каждый модуль памяти с разрушенной информацией путем замещения информации на неповрежденную копию с диска [29].

Наиболее распространен метод BER, который реализует резервирование ВП с некоторого момента времени, предшествующего обнаружению неисправности. Существует два уровня реализации процесса «возврат ВП». Первый уровень возможен, если ошибкой испорчен небольшой объем информации, который допустим для реализации механизма контрольной точки. Второй уровень восстановления необходим в случае порчи ошибкой большого объема информации или если в системе не предусмотрен механизм контрольных точек. В этих случаях применяется рестарт (повторный запуск). «Легкий» рестарт основывается на том, что все выполняемые ранее процессы сохранены, и поэтому с его помощью возобновляются все операции с момента обнаружения неисправности. «Глубокий» рестарт применяется в том случае, если ряд процессов потеряны, при этом сохраненные процессы при необходимости возобновляются от моментов обнаружения неисправностей, а потерянные процессы возобновляются сначала. Наконец, если в результате неисправности (например, типа «останов» или «зацикливание») ни один из процессов не сохраняется, то «глубокий» рестарт трансформируется в «перезапуск» системы, при котором осуществляется ее полная перезагрузка.

В отказоустойчивых управляющих ИС реального времени восстановление ВП на уровнях «глубокого» рестарта или «перезапуска» обязательно связано с возникновением в ИС частичного или полного функционального отказа. Поэтому система защиты от отказов должна иметь возможности поддерживать процесс восстановления на первом уровне, либо на уровне «легкого» рестарта.

III. Восстановление вычислительных структур ИС заключается в ремонте отказавших составных компонент, выведенных из конфигурации системы. Вычислительные структуры узкого

класса ИС, применяемых в бортовых автоматических системах автономного функционирования обычно не восстанавливаются. Широкий класс управляющих ИС в составе различных типов АСУ обслуживается ремонтными бригадами. В этих системах возможны восстановления отказавших компонент либо в рабочем состоянии, либо в интервалах времени между решениями задач управления. Ремонт в рабочем состоянии может осуществляться без задержки после вывода из конфигурации ИС отказавших компонент при совместном соблюдении следующих четырех условий: 1) вычислительные структуры скомпонованы в виде типовых элементов замены (ТЭЗов); 2) в конструкции устройств ИС предусмотрена возможность замены ТЭЗов без отключения питания других устройств; 3) в ЗИПе ИС имеется хотя бы один исправный ТЭЗ того же типа, что и выведенный из конфигурации системы; 4) есть свободная от обслуживания ремонтная бригада. Последнее условие менее жесткое, чем первые три, поскольку существует возможность освобождения ремонтной бригады (или хотя бы одного специалиста по эксплуатации в течение работы ИС после устранения отказов других компонентов ИС). Если перечисленные условия не выполняются, особенно это относится к первым двум условиям, то восстановление вычислительных структур возможно только в нерабочем состоянии.

IV. Реинтеграция. После физической замены компонента отремонтированный модуль должен быть воссоединен с системой. Для ремонта в рабочем состоянии реинтеграция должна быть выполнена без прерывания работы системы.

III.1.6.3. Системы обеспечения отказоустойчивости

Системы обеспечения отказоустойчивости (СОО) являются составной частью информационных систем (ИС) и формируются при их проектировании из предусмотренных избыточных ап-

паратных и программных средств. Все системы обеспечения отказоустойчивости (СОО) ИС по организации архитектуры этих систем можно разделить на следующие три группы:

- СОО закрытого типа;
- СОО открытого типа;
- СОО смешанного типа.

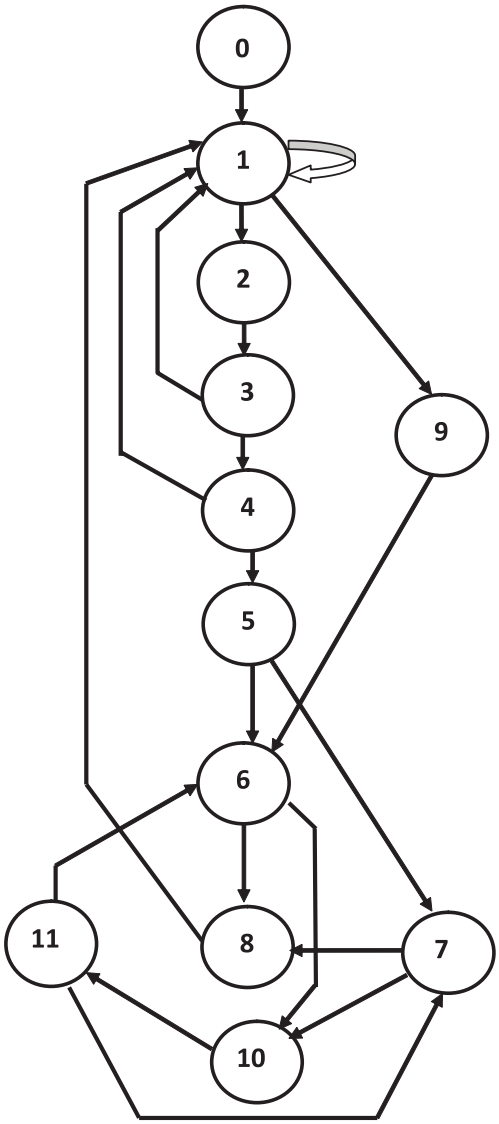
Системы обеспечения отказоустойчивости ***закрытого типа*** *строятся на базе статической избыточности.* В этих системах применяют жесткий алгоритм функционирования. Характерные приемы построения алгоритма функционирования СОО закрытого типа состоят в маскировании неисправностей на локальном (модульном) и/или глобальном (системном) уровнях. Маскирование неисправностей в чистом виде не выполняет функцию обнаружения неисправностей – его предназначение состоит в автоматической нейтрализации возникших неисправностей и факты возникновения не регистрируются. На локальном уровне для маскирования неисправностей применяются *корректирующие коды,* позволяющие как обнаруживать, так и исправлять ошибки. Для реализации таких кодов необходимы кодирующие и декодирующие устройства, содержащие mинформационных и $k<m$ контрольных разрядов. Следует отметить, что эффективные коды для исправления ошибок разработаны для условий, когда информация не преобразуется. Поэтому их применение ограничено цифровыми устройствами хранения и передачи информации и, кроме того, связано с необходимостью введения большого объема аппаратурной избыточности. На глобальном уровне для маскирования неисправностей применяется мажоритарное резервирование, в том числе искусственное или естественное гибридное мажоритарное резервирование. В п. III.2.4 подробно рассмотрены возможности мажоритарного резервирования и условия, при которых этот вид маскирования неисправностей эффективен.

Достоинство СОО закрытого типа заключается в оперативности устранения неисправностей. Недостатки: большая (порой чрезмерная) избыточность, специфическая пригодность для объектов. В СОО ***открытого типа*** реализуются свойства наблюдаемости и управляемости, возможен гибкий алгоритм функционирования. Следовательно, есть все исходные предпосылки для реализации механизма адаптации к неисправностям ИС (апостериорной адаптации), а также механизма адаптации к информационной нагрузке ИС с целью осуществления динамической стабилизации процессов обработки информации в ИС (априорной адаптации) [19] ограниченной группы задач, реализуемых устройствами ИС и др.

В таких СОО применяется *динамическая избыточность,* что позволяет значительно снизить избыточность по сравнению со статической ее организацией. Достоинства СОО открытого типа очевидны: значительно большая универсальность по сравнению с маскированием неисправностей, существование способности СОО приспособления к характеру неисправности, наличие возможности целенаправленного перестроения структуры ИС, чтобы при отказе модуля включить в работу другие имеющиеся ресурсы и т.д. Вместе с тем, проявляются и существенные недостатки: увеличивается время адаптации к отказам модулей ИС, значительно повышается роль средств обнаружения неисправностей, эффективность которых можно обеспечить опять – таки за счет введения значительной структурной, информационной или временной избыточности. Этот замкнутый круг удается разорвать с помощью оригинальных методов *адаптивной отказоустойчивости* (активной защиты).

В определенной мере недостатки, присущие закрытым и открытым СОО, устраняются в СОО ***смешанного типа***. Граф состояний этих систем показан на рис. III.1.4. В состоянии 0 информационная система находится в режиме подготовки к

функционированию. Предотвращаются неисправности путем создания комфортной работы аппаратуры и программ, а также осуществляется предварительная локализация неисправностей



0 – режим подготовки к функционированию ИС;
1 – состояние нормального функционирования, отказоустойчивая работа ИС;
2 – обнаружение факта неисправности;
3 – локализация неисправности;
4 – анализ неисправности, идентификация отказов или сбоев;
5 – обнаружение местонахождения отказавшего элемента;
6 – реконфигурация ИС с помощью резервных модулей;
7 – реконфигурация ИС на основе постепенной деградации системы;
8 – полное восстановление вычислительного процесса;
9 – маскирование неисправностей;
10 – состояние ремонта отказавшего элемента системы;
11 – реинтеграция отремонтированного элемента системы

Рис. III.1.4. Граф состояний системы обеспечения отказоустойчивости информационной системы

путем решения контрольных задач, контроля функционирования ИС, диагностического тестирования программ и аппаратуры.

В состояние 1 система переходит при поступлении заявок на обслуживание. В этом состоянии осуществляется динамическая стабилизация отказоустойчивости ИС. При возникновении неисправности происходит либо обнаружение и локализация неисправности (состояния 2 и 3), либо маскирование неисправности (состояние 9), если это предусмотрено. Если допустимо исключение искаженной информации или степень искажения незначительна и ею можно пренебречь, то происходит переход в состояние 1. В противоположных и наиболее характерных случаях требуется продолжить анализ неисправности и классифицировать ее на отказы или сбои (состояние 4). В случае сбоя происходит переход 4-1.

Если в результате классификации неисправности установлен отказ, то в состоянии 5 включаются механизмы обнаружения отказавшего модуля. Затем происходит переход 5-6 и производится реконфигурация системы с помощью резервных модулей. Если резервные модули отсутствуют (не предусмотрены или использованы), то происходит переход в состояние 7 для реконфигурации системы с помощью естественной избыточности (свободных основных модулей, либо путем постепенной деградации).

При любом варианте реконфигурации последует переход в состояние 8 полного восстановления вычислительного процесса с дальнейшим переходом 8-1, а также ремонт отказавшего модуля (состояние 10). Реинтеграция отремонтированного модуля (или замененного на исправный из ЗИПа) производится с приоритетом состоянию 7 системы по отношению к состоянию 6. Данная система обеспечения отказоустойчивости является основой для построения системы обеспечения адаптивной отказоустойчивости (активной защиты), описанной в главе III.3.

III.1.7. Отказобезопасность. Киберзащищенность

III.1.7.1. Отказобезопасность

Согласно [6, 8, 10] «опасность – ситуация, потенциально оказывающая вред человеку». Это, конечно, относится не только к человеку, но и к ущербам, которые могут быть причинены материальным ценностям или окружающей среде. Не каждая опасность всегда переходит в угрозу. Для этого необходимо, чтобы случилось инициирующее событие. Потом из угрозы может развиваться цепочка нежелательных событий, которая в конечном счете сведет к опасному событию, к аварии. Опасное состояние – это неисправное состояние объектов информационной системы, при котором возникают превышающие допустимые уровни риски причинения вреда жизни и здоровью граждан, имуществу физических и юридических лиц, государственному и муниципальному имуществу, окружающей среде, жизни и здоровью животных и растений.

Итак, безопасность – отсутствие неприемлемого риска. Риск – это комбинация ущерба и вероятности возникновения опасного события [10]. В зависимости от последствий можно отдельно рассматривать:

а) функциональную надежность системы, если она выполняет свою функцию (т.е. не теряет определенных свойств) в цепочке всех тех систем, которые участвуют в выполнении задачи;

б) функциональную безопасность системы, если последствия нарушения её функционирования не приведут к неприемлемым рискам.

Рис. III.1.5 показывает, что со стороны последствий функциональная надежность плавно переходит в функциональную безопасность, если тяжесть последствий возрастает. Отсюда и

понятно, что информационные системы, у которых вероятность отказа в выполнении функции должна быть не выше 10^{-5} [8] в течение часа работы, можно толковать с позиций функциональной безопасности. Если вероятность отказа выше указанного уровня, то такие системы представляют повышенную опасность для человека и окружающей среды (рис. III.1.6). Для них опасный отказ определяется как событие, в результате которого система переходит из исправного, работоспособного или частично работоспособного состояния в опасное состояние.

Отказобезопасность – способность информационной системы сохранять безопасное состояние и (или) обеспечивать безопасность управления подчиненными объектами в случае отказов самой системы или ее составных частей, а также в случае негативного внешнего информационного воздействия.

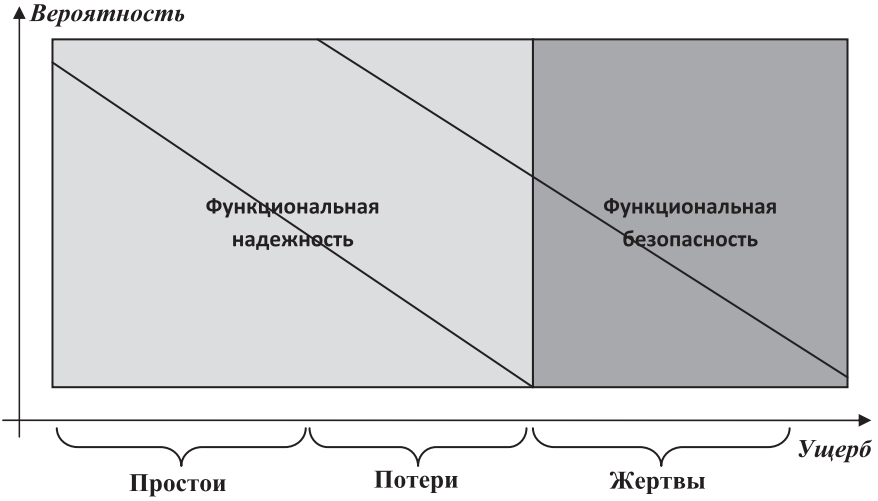


Рис. III.1.5. Функциональная надежность и функциональная безопасность

Проблема обеспечения функциональной безопасности управляющих ИС состоит в исключении влияния их отказов и ошибок в функционировании на объекты управления и окружа-

ющую среду [35, 36], т.е. в исключении так называемых опасных отказов (рис. III.1.6).

Кроме отказов и ошибок в функционировании ИС на ее безопасность могут повлиять информационные атаки. В работе [2] было отмечено, что путем, например, DoS-атаки можно нарушить доступность информации, а с помощью троянских программ – исказить целостность данных и программ. В итоге, как в результате угроз нарушения надежности системы, так и под воздействием информационных атак, нарушаются целостность и доступность информации. Это может привести к опасным отказам. В результате выдачи неправильных команд управления возможны столкновения поездов, разгерметизация цистерн, возможны пожары, взрывы и т. д.

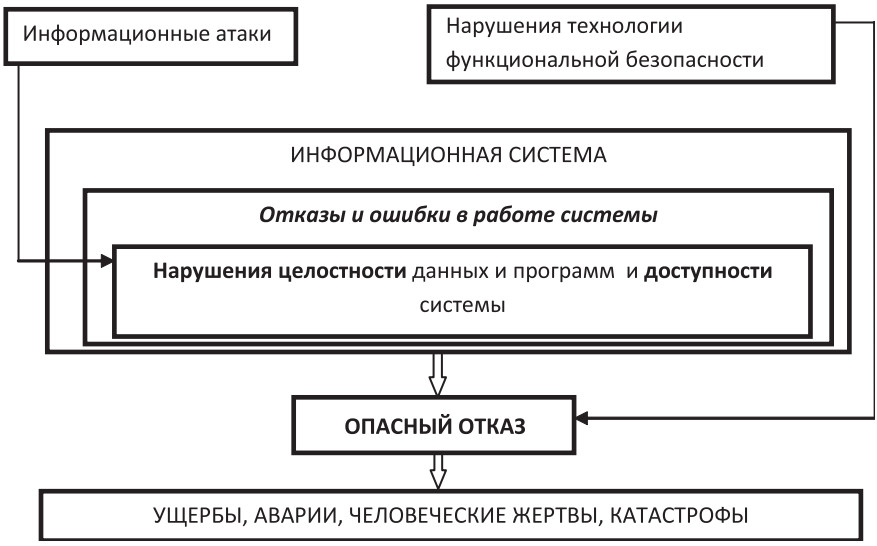


Рис. III.1.6. Угрозы безопасности в информационных системах

В итоге, возникают экономические и экологические ущербы, человеческие жертвы и даже катастрофы. Полностью исключить влияние отказов и ошибок систем на окружающую среду при-

нципиально невозможно – всегда существует некоторая вероятность возникновения подобных событий. Задача заключается в достижении минимально допустимых значений этих вероятностей. Одним из ключевых направлений в достижении этих целей остается обеспечение высокого уровня надежности систем. Однако возможности резкого повышения надежности известными алгоритмическими, структурными и др. методами ограничены, главным образом из-за неприемлемых экономических потерь.

Снижение частоты опасных отказов также возможно в результате построения эффективной системы защиты от информационных атак. Однако затраты на повышение эффективности защиты от информационных атак далеко не всегда соизмеримы с ущербами от опасных отказов.

Решение проблемы обеспечения отказобезопасности управляющей ИС заключается в комплексной реализации мер по обеспечению надежности, информационной защищенности и, особенно, функциональной безопасности. С позиций безопасности не может быть другой альтернативы как прекратить функционирование системы или понизить до предусмотренных пределов ее производительность при недопустимой вероятности возникновения в ней опасного отказа. Отсюда следует необходимость создания технологий гарантированного и достоверного обнаружения отказов в системах. Если правильно реализована технология обеспечения функциональной безопасности, то обеспечивается своевременное обнаружение и блокирование опасных состояний системы.

С позиций безопасности не может быть другой альтернативы как прекратить функционирование системы или понизить до предусмотренных пределов ее производительность при недопустимой вероятности возникновения в ней опасного отказа. Высокая эффективность обнаружения опасных состояний в ИС достигается с помощью технологий обнаружения отказов,

основанных на построении двух, трех и более параллельных каналов управления. Параллельное формирование и сравнение выдаваемых команд управления обеспечивает уверенность в обнаружении опасных состояний при условии построения безопасных алгоритмов или устройств сравнения (так называемых компараторов), обеспечения независимости каналов и данных, несимметричности отказов каналов и при выполнении целого ряда других условий. Вместе с тем, для реализации указанной технологии обнаружения отказов дополнительно требуется включение в состав системы значительного объема аппаратных и программных средств, что приводит к снижению ее надежности.

Из указанных положений следует:

– между содержанием безопасности и надежности систем имеют место принципиальные различия – если ненадежность приводит к неприемлемым уровням готовности, технического использования, безотказности и стоимости технического обслуживания, то недостаточная безопасность приводит к авариям и человеческим жертвам;

– между целями обеспечения надежности и функциональной безопасности систем существуют противоречия, устранения которых возможно на основе компромисса, требования к надежности и функциональной безопасности должны быть между собой сбалансированы;

– в объектах, представляющих повышенную опасность, в потенциально опасных и критически важных системах приоритеты отдаются задачам обеспечения безопасности, а требуемые уровни надежности должны задаваться с учетом ограничений по стоимости после выполнения требований по безопасности.

Для обеспечения отказобезопасности информационных систем нужно проделать, тот же путь, что и для обеспечения отказоустойчивости, т.е. создать условия для наблюдаемости и

управляемости системы. Кроме того, на основе принципа приемлемости остаточного риска при имеющихся ограничениях в затратах средств нужно построить защиту от внутренних возмущающих воздействий (отказов, сбоев, программных ошибок), от информационных атак и в полной мере реализовать возможности по обеспечению функциональной безопасности.

III.1.7.2. Киберзащищенность

Проблема обеспечения отказобезопасности в системах управления неразрывно связана с вопросами обеспечения их информационной защищенности, в первую очередь, от кибератак. Термин *кибер* определяется как «имеющий отношение к информационным технологиям» [30]. Информационные технологии реализуются в так называемом *киберпространстве*, под которым понимается «среда, созданная при помощи физических и не физических компонентов, характеризуемая использованием компьютеров и электромагнитного диапазона для хранения, изменения и обмена данными при помощи компьютерных сетей» [30]. Использование кибернетических возможностей с целью достижения задач в киберпространстве или при помощи использования киберпространства определяется как «кибероперация». Теперь мы вплотную подошли к определению понятия *кибератака* – это кибероперация, как наступательная, так и оборонительная, которая приводит к телесным повреждениям или человеческим потерям или нанесению ущерба или разрушению объектов.

Опираясь на приведенные выше понятия**,** можно определить **киберзащищенность как способность информационной системы управления успешно выполнять предусмотренные задачи в условиях деструктивных воздействий, вызванных кибератаками, а также технологическими нарушениями и/или отказами составных технических средств.**

Многие термины в области киберзащищенности информационных систем управления содержатся в стандарте [31].

В предлагаемой читателю книге под киберзащищенностью подразумевается комплексное понятие безопасного функционирования информационных систем управления

Основными угрозами нарушения киберзащищенности в информационных системах являются (рис. III.1.7):

- информационные атаки (в первую очередь кибератаки), т.е реализация несанкционированного доступа (НСД) вероятного противника к системе;
- недекларированные возможности (НДВ) в программах и устройствах систем;
- отказы и ошибки в работе системы, в том числе аппаратные и программные сбои и ошибки, ошибки операторов, ошибки данных.

Полное устранение опасных отказов в управлении теоретически возможно, но практически не осуществимо, поскольку потребует экономических затрат, заведомо больших, чем ожидаемый ущерб от воздействия опасных отказов. Реальный путь – это определение допустимого уровня риска от кибератак и создание эффективной защиты от опасных отказов.

Рассмотрим более подробно перечисленные угрозы.

Результатом успешной кибератаки может стать нарушение целостности или доступности информации. В качестве целей атаки могут рассматриваться серверы, рабочие станции пользователей или коммуникационное оборудование информационной системы. При организации кибератак злоумышленники часто используют специализированное ПО, позволяющее автоматизировать действия, выполняемые на различных стадиях атаки.

В общем случае в любой кибератаке можно выделить четыре стадии:

Рекогносцировка. На этой стадии нарушитель старается получить как можно больше информации об объекте атаки,

чтобы на ее основе спланировать дальнейшие этапы вторжения. Этим целям может служить, например, информация о типе и версии операционной системы; список пользователей, зарегистрированных в системе; сведения об используемом прикладном ПО и т. д.

Вторжение. На этом этапе нарушитель получает несанкционированный доступ к тем ресурсам, на которые совершается атака.

Атакующее воздействие. На данной стадии реализуются те цели, ради которых и предпринималась атака, – например, нарушение работоспособности системы, удаление или модификация данных и т. д. При этом атакующий часто выполняет операции, направленные на удаление следов его присутствия в системе. Всякая атака основана на наличии в системе управления уязвимостей, и «правильное» использование хотя бы одной из них открывает злоумышленнику вход в систему.

Развитие атаки. После атакующего воздействия нарушитель стремится перевести атаку в фазу дальнейшего развития. Для этого в систему обычно внедряется вредоносная программа, с помощью которой можно организовать атаку на другие средства системы. Основные угрозы нарушения киберзащищенности информационным системам (ИС) создают следующие группы вредоносных программ: ***DoS-атака*** (от англ. Denial of Service, – отказ в обслуживании) – атака на информационную систему с целью довести её до отказа, т.е. создание таких условий, при которых легитимные (правомерные) пользователи системы не могут получить доступ к предоставляемым системой ресурсам (серверам), либо этот доступ затруднён. Отказ «вражеской» системы может быть одним из шагов к овладению системой (если во внештатной ситуации ПО выдаёт какую-либо критическую информацию – например, версию, часть программного кода и т. д.);

троянские программы **–** после внедрения в систему нарушают целостность данных и программ или рассаживают вирусы в системе. Они также могут собрать сведения о хранящихся на компьютере профилях пользователей, паролях и другую конфиденциальную информацию и затем переслать ее в руки злоумышленников; ***программы несанкционированного управления компьютерами ИС*** *(загрузочные вирусы, программные вирусы, сетевые черви и др.).*

Рис. III.1.7. Концепция обеспечения гарантированного уровня кибезащищенности информационной системы реального времени

Недекларированные возможности – функциональные возможности программных или аппаратных средств, не описанные или не соответствующие описанным в документации, при использовании которых возможно нарушение доступности, целостности, а также конфиденциальности обрабатываемой информации. Реализацией недекларированных возможностей, в частности, являются программные или аппаратные закладки.

В результате реализации указанных угроз возникают опасные отказы, которые приводят к недопустимым ущербам для объектов, которые на уровне хозяйствующего субъекта относятся к

категории объектов с повышенной опасностью или к категории потенциально опасных объектов, а на уровне государственных или региональных органов опасные отказы могут приводить к недопустимым ущербам критически важных объектов. Последнее обстоятельство объясняется тем, что ответственность за защиту критически важных объектов возлагается на государственные или региональные органы.

Угрозы нарушения киберзащищенности систем управления аналогичны угрозам нарушения отказобезопасности. Принципиальное различие в том, что кибератаки – это специфический класс информационных атак, направленный на нанесение ущерба или разрушение объекта управления, который относится к одной из отмеченных выше трех групп важных объектов.

В вопросах обеспечения киберзащищенности, так же как и в вопросах обеспечения отказоустойчивости и отказобезопасности, целесообразно опираться на следующие основные **постулаты***:*

1. Не существует абсолютной киберзащищенности (отказоустойчивости, отказобезопасности) систем управления.

2. Чем более сложная система, чем больше задач она выполняет, тем ниже ее киберзащищенность.

3. Необходимым условием повышения киберзащищенности системы есть введение избыточности в сочетании с организацией эффективного контроля.

4. Киберзащищенность системы управления должна обеспечиваться на всех этапах жизненного цикла

5. Уровень киберзащищенности системы ограничен экономическими рисками заказчика и эксплуатирующей организации.

Абсолютной киберзащищенности невозможно достичь, поскольку устранение одних уязвимостей в системе не исключает возможности появления новых. Проблема обеспечения ки-

берзащищенности – это проблема совершенствования щита от нападения меча. Одновременно с повышением уровня защиты совершенствуются средства нападения и не факт, что эффективность средств защиты в определенные отрезки времени сколь угодно выше эффективности средств нападения.

Поскольку абсолютной безопасности не существует, то необходимо оценить уровень остаточных рисков на основании стандарта [10] и привести обоснование безопасности системы в виде документа, определенного стандартом [11].

III.1.8. Вопросы для самоконтроля

1. Перечислите постулаты обеспечения надежности ИС.
2. Приведите определение избыточности и сформулируйте условие, при котором применение средств защиты от отказов имеет смысл.
3. Перечислите виды избыточности.
4. Раскройте суть структурной и временной избыточности.
5. Раскройте суть информационной, функциональной, алгоритмической и программной избыточности.
6. Приведите и объясните условие, при котором применима многоуровневая избыточность.
7. Приведите и поясните диаграмму ALARP.
8. Перечислите этапы жизненного цикла надежности ИС.
9. Раскройте цели и задачи Программы и Политики обеспечения надежности ИС.
10. Раскройте цели, задачи и перечислите разделы Доказательства надежности и безопасности.
11. Раскройте суть отказоустойчивости ИС и поясните виды отказоустойчивости.
12. Что понимается под термином «наблюдаемость ИС» и какие этапы наблюдаемости Вам известны?

13. В чем суть понятий «управляемость и реконфигурация ИС»?

14. Приведите и поясните граф состояний системы обеспечения отказоустойчивости ИС.

15. В чем общность и различие надежности и безопасности ИС?

16. Поясните сущность отказобезопасности ИС.

17. Приведите определение киберзащищенности ИС и поясните стадии кибератаки.

18. В чем суть концепции обеспечения гарантированного уровня киберзащищенности ИС?

ГЛАВА III.2. РЕЗЕРВИРОВАНИЕ В ИНФОРМАЦИОННЫХ СИСТЕМАХ

В этой главе, не претендуя на общность, мы ограничимся рассмотрением двух видов резервирования в информационных системах: структурного и информационного. Эти виды резервирования распространены во всех классах информационных систем – от бортовых до стационарных, от систем сбора и хранения информации до управляющих систем. Структурное и информационное резервирование позволяют стандартными решениями добиться желаемого эффекта в обеспечении надежности информационных систем.

III.2.1. Классификация структурного резервирования в информационных системах

Под резервированием понимают способ обеспечения надежности объекта за счет использования дополнительных средств, избыточных по отношению к минимально необходимым средствам для выполнения предусмотренных функций. Структурное резервирование в информационных системах создается за счет дополнительных аппаратных средств.

Структурное резервирование предназначено для обеспечения отказоустойчивости систем. Уже на первом этапе развития вычислительной техники Дж. Фон Нейман сформулировал принцип мажоритарного преобразования, а К. Шеннон и Э. Мур разработали основы теории избыточных релейных схем, где релейной схемой можно назвать любое устройство дискретного

действия независимо от физической природы используемых в нем логических элементов. Они предложили метод поэлементного резервирования и исследовали гамакообразные схемы различной ширины и длины. Современная теория и практика структурного резервирования во многом основывается на указанных работах. В информационных системах большее влияние на эффективность структурного резервирования оказывают следующие основные факторы:

– количество резервных устройств;

– возможность и параметры восстановления отказавших устройств;

– надежность переключающих устройств;

– эффективность и надежность средств оперативного обнаружения отказов;

– длительность существования необнаруженных средствами контроля отказов (длительность скрытых отказов);

– допустимое время перерыва в работе резервированных устройств и систем, которое определяется допустимым временем перехода с отказавшего основного устройства на исправное резервное и др.

В соответствие с публикациями [32, 33, 34 и др.], структурное резервирование классифицируется по следующим признакам:

• ***по степени охвата резервом объекта и его составных элементов***. По этому признаку различают общее и раздельное резервирование. *Общее резервирование*: резервирование, при котором резервируется объект в целом. *Раздельное резервирование*: резервирование, при котором резервируются отдельные элементы объекта или их группы;

• ***по назначению элементов объекта в выполнении требуемой функции.*** По этому признаку различают основной и резервный элементы. *Основной элемент*: элемент объекта, необходимый для выполнения требуемых функций без использования

резерва. *Резервный элемент:* элемент, предназначенный для выполнения функций основного элемента в случае его отказа или замены;

• ***по степени привлечения элементов объекта к выполнению требуемой функции.*** По этому признаку различают резервирование замещением, резервирование *m* из *n,* смешанное резервирование, постоянное резервирование, мажоритарное резервирование. *Резервирование замещением:* резервирование, при котором часть элементов объекта, способных выполнять требуемую функцию, предназначена для работы, а остальная часть средств не работает до момента появления необходимости в ней. *Резервирование m из n*: резервирование, при котором *m* элементов объекта из общего их количества *n* должны функционировать для выполнения требуемой функции, остальные находятся в рабочем состоянии готовности. *Смешанное резервирование:* резервирование, обеспечивающее выполнение требуемой функции несколькими различными средствами и (или) способами. Например, часть элементов объекта, привлекаемых для выполнения функции, охвачена общим резервом, другие элементы имеют раздельное резервирование. Следует понимать, что при резервировании *m* из *n,* в отличие от резервирования замещением, не важно в каком состоянии работы или без работы находятся не привлеченные к функционированию *n* – *m* элементов объекта. При резервировании замещением не привлекаемая к работе часть элементов объекта находится в режиме ожидания без работы. *Постоянное резервирование* построено по жесткой схеме работы всех имеющихся в наличии *n* элементов. *Мажоритарное резервирование*, – это резервирование, при котором работают в нагруженном режиме три или более однотипных элемента с голосованием по большинству одинаковых результатов работы. С помощью, так называемого, мажорирующего органа производится сравнение результатов работы всех *n* од-

нотипных элементов и проверяется условие, что не менее *m* из *n* (где $m < n$) элементов выдают одинаковые выходные результаты работы. Эти результаты считаются истинными и принимаются для дальнейшей обработки в информационной системе. Следует обратить внимание на принципиальное различие между мажоритарным резервированием по логике *m* из *n* и резервированием кратности *m* из *n*. В первом случае к работе привлекаются все *n* имеющихся в объекте элементов и реализуются условия, когда все эти элементы исправны, или исправны *n-1* элемент, *n-2* элемента,…, *m* элементов. Во втором случае к работе привлекаются ровно *m* элементов, а остальные *n-m* элементов находятся в резерве. Говорить о преимуществе первого или второго способа структурного резервирования не следует, поскольку они преследуют разные цели – мажоритарное резервирование предназначено для маскирования не*исправностей, главным образом, сбоев, тогда как резервирование кратности m* из *n* предназначено для защиты от отказов элементов резервированного объекта;

• ***по качеству переключения отказавших основных элементов на резервные элементы.*** По этому признаку различают объекты *без оперативного времени переключения на резерв и с оперативным временем переключения на резерв*. Это оперативное время должно быть меньше допустимого значения перерыва в функционировании. Допустимое время переключения на резерв может быть постоянным или случайным. Резервирование замещением реализуется с помощью переключающего элемента. *Переключающий элемент:* элемент объекта, предназначенный для замещения основного или резервного элемента. Он может быть выполнен только аппаратными средствами, либо только программными средствами или программно – аппаратными средствами. Качество переключения отказавших основных элементов на резервные оценивается *вероятностью успешного перехода на резерв* – вероятностью того, что переход

на резерв произойдет без отказа объекта, т.е. за время, меньшее допустимого значения перерыва в функционировании, и во время перехода переключающий элемент находится в работоспособном состоянии;

• ***по однотипности характеристик надежности элементов объекта.*** По этому признаку различают *однотипные и разнотипные* элементы объекта. Широкое распространение получили полностью однотипные объекты, которые имеют идентичные аппаратные и программные средства и идентичные характеристики надежности. Примером однотипности элементов объекта является *дублирование* – резервирование 1 из 2, при котором оба элемента объекта (основной и резервный) однотипны. Дублирование есть частный случай резервирования *m* из *n*с однотипными элементами. Однако этот частный случай чрезвычайно распространен на практике. К классу однотипных элементов объекта с позиции структурного резервирования можно отнести элементы, имеющие характеристики надежности, которые с приемлемой погрешностью можно принять одинаковыми. При этом они могут иметь некоторые различия в составе аппаратуры и программ. Разнотипные элементы объекта существенно различаются своими характеристиками надежности;

• ***по степени нагруженности резервных элементов*** в информационных системах различают нагруженное или ненагруженное резервирование. Иногда облегченное резервирование. Часто в разговорной речи и даже в технической литературе пользуются сленгом типа «горячий резерв», «холодный резерв». Применение таких нестандартных терминов не способствует пониманию физической сущности этого свойства резервирования. *Нагруженное резервирование* – это резервирование, при котором все средства, способные выполнять требуемую функцию, работают одновременно. *Ненагруженный резерв* не используется для выполнения требуемых функций, более того, он

находится в выключенном состоянии. *Облегченное резервирование* применяется главным образом в средствах электропитания информационных систем;

• ***по степени ремонтопригодности*** различают *восстанавливаемые и невосстанавливаемые* резервированные объекты. Речь идет о возможности восстановления в процессе функционирования объекта. Обычно полагают, что резервирование с восстановлением – это резервирование, при котором восстановление отказавших основных или резервных элементов объекта технически возможно без нарушения его работоспособности и предусмотрено эксплуатационной документацией. Если восстановление в стационарных информационных системах возможно, то эффективность резервирования резко возрастает. В бортовых информационных системах ремонтировать отказавший элемент в процессе выполнения задачи (например, полета самолета, движения поезда и т.д.) нереально. Интерес представляет восстановление с задержкой времени, вызванной необходимостью завершения задачи бортовыми системами и последующей возможностью устранения отказа в стационарных условиях. В этой ситуации восстановление способствует повышению готовности резервированного объекта, однако не влияет на надежность выполнения задачи, поскольку она уже завершена.

Эффективность восстановления в резервированных объектах находится в прямой зависимости от интенсивности восстановления и, конечно, от количества *ремонтных бригад*. Под ремонтной бригадой обычно понимается функциональная единица, способная самостоятельно устранить отказ. Это может быть один специалист или несколько работников, которые совместными усилиями выполняют работу по восстановлению отказавшего элемента. Если в системе предусмотрены две и более ремонтные бригады, то это означает возможность параллельного и независимого восстановления двух и более отказавших элемен-

тов резервированного объекта. При наличии одной ремонтной бригады восстановление отказавших элементов производится последовательно;

• ***по степени охвата средствами контроля*** основных и резервных элементов объекта различают: объекты с контролем основных и резервных элементов, объекты с контролем основных элементов при отсутствии контроля резервных элементов. Уровень надежности резервированных объектов в информационной системе находится в сильной зависимости от эффективности средств обнаружения отказов элементов. Это в большей мере относится к эффективности средств контроля основных элементов объекта.

На рис. III.2.1 приведена схема классификации структурного резервирования. Она состоит из шести связанных между собой слоев. *Первый слой* – общее или раздельное резервирование. И тот и другой виды резервирования могут быть реализованы как по схеме резервирования замещением, так и по схеме постоянного, смешанного резервирования или резервирования типа *n*/*m,* (*второй слой*), а может быть по схеме голосования по большинству голосов (мажоритарное резервирование). Вместе с тем, ни постоянное резервирование, ни мажоритарное резервирование не связаны с временем переключения на резерв, поскольку при этих схемах не требуется переключение отказавшего основного элемента на резервный. Исключение может составить организация гибридного мажоритарного резервирования. В этой структуре в резервированной системе мажоритарную группу элементов дополняют одним или более однотипными элементами, которые включены по схеме резервирования замещением. Время переключения на резерв всегда ограничено, но в одном случае допустимы относительные большие временные рамки, в системах реального времени необходимо обеспечивать оперативное время переключения на резерв (*слой 3*) . Влияние

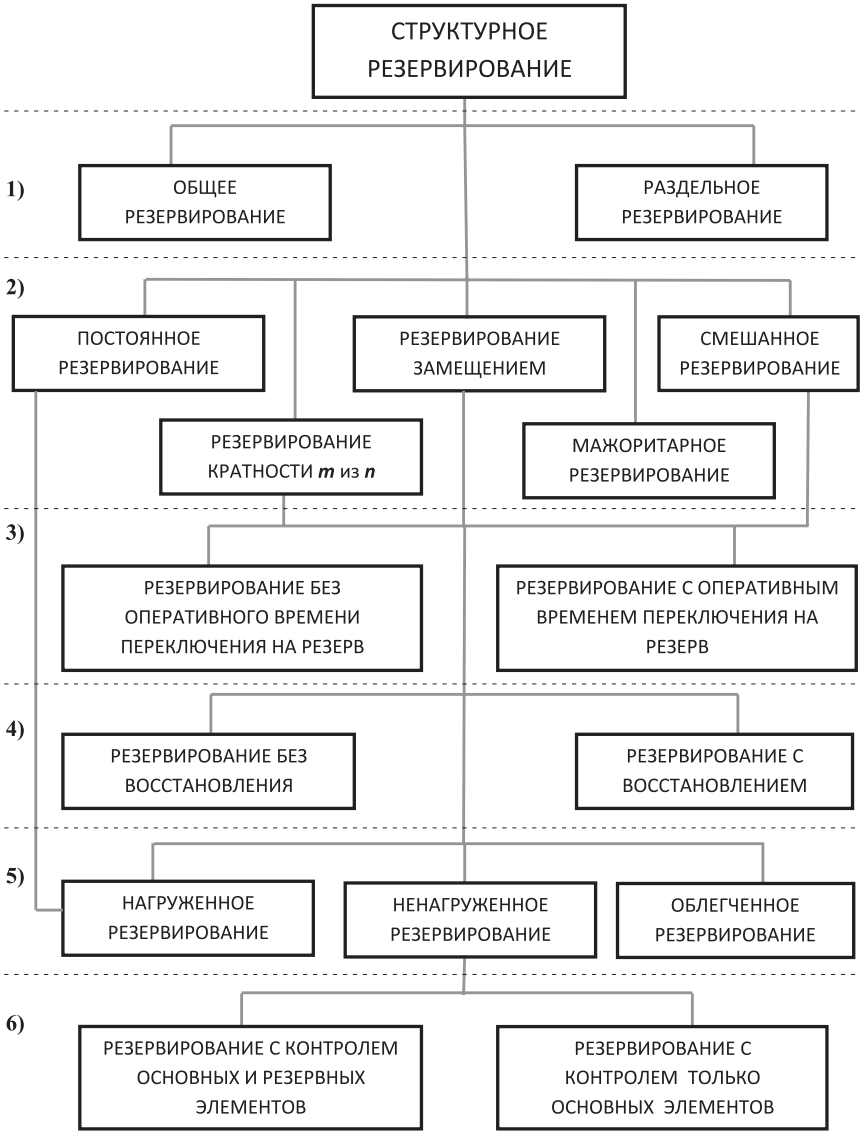


Рис. III.2.1. Классификация структурного резервирования

времени переключения на резерв особенно ощутимо в объектах с восстанавливаемым резервом (*слой 4*). При этом учитывается степень нагруженности объекта (*слой 5*) и эффективность средств контроля (*слой 6*).

III.2.2. Структурное резервирование без восстановления

Наглядное представление о виде резервирования дают так называемые структурно – логические схемы надежности. Рассмотрим структурно – логические схемы резервированных объектов без восстановления отказов (рис. III.2.2, III.2.3, III.2.4). Эти схемы отличаются от структурных схем тем, что в них кроме структур резервированных объектов продемонстрирована также логика подключения резервных элементов. Например, стрелки при резервных элементах означают резервирование замещением, а линии – постоянное резервирование.

III.2.2.1. Общее постоянное резервирование

При построении моделей надежности резервированных объектов без восстановления с постоянным резервированием будем руководствоваться постулатом 4 (III.1.1**),** согласно которому *необходимым условием повышения надежности системы есть введение избыточности в сочетании с контролем.* Для обеспечения успеха в реализации резервирования необходимо добиться правильного обнаружения любого отказа в модуле. Поясним эту мысль, Пусть средства контроля элемента объекта (аппаратные, программные или смешанные) построены таким образом, что отказы 60% аппаратуры элемента правильно обнаруживаются с вероятностью 0,95, а отказы оставшейся части элемента (40% аппаратуры) правильно обнаруживаются с вероятностью 0,6.Тогда средняя вероятность правильного обнаружения любого отказа элемента объекта (обозначим её как α) в условиях данного примера равна $\alpha = 0{,}6 \cdot 0{,}95 + 0{,}4 \cdot 0{,}6 = 0{,}81$.

Таким образом, если основной или резервный элемент охвачен L типами непересекающихся контролей, где каждый i-й тип

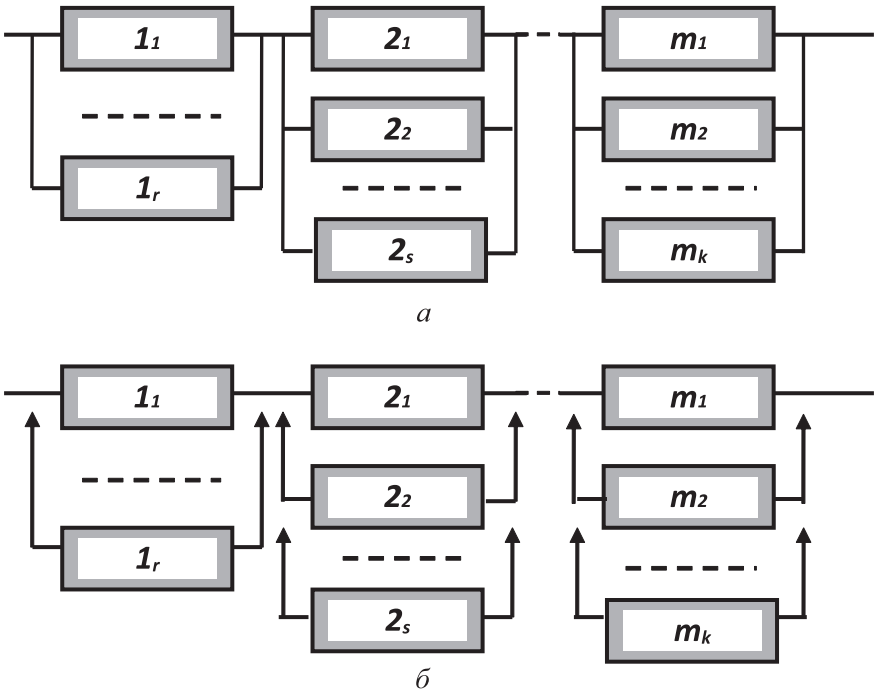


Рис. III.2.2. Структурно – логические схемы объектов с раздельным нагруженным постоянным резервированием (а) и резервированием замещением (б)

контроля ($i = 1, 2, \ldots, L$) предназначен для обнаружения отказов V_i части этого элемента ($0 \leq V_i \leq 1$) с вероятностью правильного обнаружения p_i, то средняя вероятность правильного обнаружения любого отказа элемента объекта определяется с помощью выражения

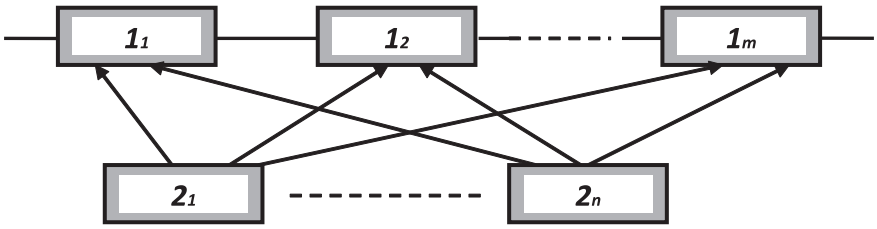


Рис. III.2.3. Структурно – логическая схема объектов с резервированием кратности n из m

$$\alpha = \sum_{i}^{L} p_i \cdot V_i .$$

Заметим, что вероятность ложной тревоги в резервированных объектах, с одной стороны, может приводить к дополнительным переключениям на резерв, что в целом практически не снижает их надежности, а с другой стороны, эта вероятность ничтожно мала. Эти соображения позволяют в дальнейшем при исследовании надежности резервированных объектов с реальными возможностями контроля ограничиться вероятностью прав ильного обнаружения отказов.

Показатель α характеризует эффективность обнаружения отказов и совместно с интенсивностью отказов элемента и значениями количества основных и резервных элементов может использоваться в качестве входного данного для построения модели надежности объекта с соответствующим видом резерва.

Предположим, что показатель эффективности обнаружения отказов равен α=1, что означает идеальную эффективность средств контроля. В этом случае вероятность безотказной работы объекта с общим постоянным резервированием кратности n (целая кратность) (рис. III.2.2) определяется в виде

$$P_{\text{О}}(t) = P_m^{n+1}(t) + (n+1)P_m^n(t)g_m(t) + \binom{n+1}{2} P_m^{n-1}(t)g_m^2(t) + \ldots$$
$$\ldots + \binom{n+1}{i} P_m^{n-i+1}(t)g_m^i(t) + \ldots + \binom{n+1}{n} P_m(t)g_m^n(t) = 1 - g_m^{n+1}(t) \qquad (2.1)$$

где $P_m(t) = \prod_{j=1}^{m} P_j(t)$ и $P_j(t)$ – вероятность безотказной работы j-го элемента одной линейки объекта; $g_m(t)$=1–$P_m(t)$.

Формула (2.1) основана на том, что отказ объекта возникает только тогда, когда отказал хотя бы один элемент в основной линейке и отказали хотя бы по одному элементу в каждой из имеющихся резервных линеек объекта. *Это необходимо учитывать,* ***но этого недостаточно, поскольку отказ объекта с общим постоянным нагруженным резервированием может возникать не только в указанном выше случае, но и при условии наличия исправной резервной линейки элементов, когда не обнаружен отказ элемента основной линейки объекта***. Учтем и это обстоятельство при следующих предпосылках:

– эффективности обнаружения отказа каждого элемента объекта идентичны, т. е $\alpha_1=\alpha_2=\ldots=\alpha_j=\ldots\alpha_m=\alpha$;

– устройства контроля надежности каждого элемента объекта идеально надежны;

– ошибка второго рода при обнаружении отказа (ложное обнаружение и, следовательно, ложное подключение резервной линейки элементов) не влияет на снижение надежности объекта.

При этих условиях формула расчета вероятности безотказной работы объекта с общим нагруженным резервом и контролем основных элементов может быть определена путем суммирования вероятностей следующих событий:

– событие А: основная линейка объекта работает в течение времени t безотказно. Вероятность этого события $P(A)=P_m(t)$;

– событие Б: при возникновении отказа основной линейки работу осуществляет любая одна из n исправных резервных линеек при условии, что отказ основной линейки обнаружен с вероятностью α. Вероятность этого события $P(Б)=\overline{P}(A)[1-\overline{P}^n(A)]\alpha=g_m(t)[1-g_m^n(t)]\alpha$

Результирующая вероятность равна

$$P_{\text{О}}(t)=P(A)+P(Б)=P_m(t)+\alpha\cdot g_m(t)[1-g_m^n(t)] \qquad (2.2)$$

При $\alpha=1$ формула (2.2) преобразуется в формулу (2.1).

При $\alpha=0$ вероятность безотказной работы резервированной системы равна вероятности безотказной работы основной (рабочей) линейки элементов, т.е. $P_{\text{О}}(t)=P_m(t)$

В целях иллюстрации практической применимости приведенных расчетных выражений примем экспоненциальными распределения отказов составных элементов объекта, а также их идентичность.

Пример.2.1.Для случая дублирования ($n=1$) требуется найти показатели безотказности объекта.

Решение. Из (2.2) получим

$$P_{\text{О}2}(t)=e^{-m\lambda t}+\alpha(1-e^{-\lambda mt})(1-1+e^{-\lambda mt})=e^{-\lambda mt}(1+\alpha-\alpha\cdot e^{-\lambda mt}).$$

Пользуясь соотношениями между показателями безотказности, определяем

$$\lambda_{\text{О}2}(t)=-\frac{P_2^{|}(t)}{P_2(t)}=\frac{\lambda m\cdot e^{-\lambda mt}}{e^{-\lambda mt}(1+\alpha-\alpha\cdot e^{-\lambda mt})}$$

$$T_{\text{О}2}=\int_0^{\infty}P_2(t)dt=\frac{1+\alpha}{\lambda m}-\frac{\alpha}{2\lambda m}=\frac{2+\alpha}{2\lambda m},$$

где m – количество элементов в основной и в каждой резервной линейках объекта.

Очевидно, что при идеальной эффективности контроля получаем известные формулы общего нагруженного постоянного дублирования без восстановления

$$P_{\text{О}2}(t)=2e^{-\lambda mt}-e^{-2\lambda mt};\ \lambda_{\text{О}2}(t)=\frac{2\lambda m(1-e^{-\lambda mt})}{2-e^{-\lambda mt}};\ T_{\text{О}2}=\frac{3}{2m\lambda}$$

При α=0

$$P_{O3}(t) = e^{-m\lambda t};\ \lambda_{O3}(t) = m\lambda;\ T_{O3} = \frac{1}{m\lambda}$$

Пример 2.2.При двух резервных каналах (*n=2)*имеют место следующие формулы расчета показателей надежности объекта с постоянным нагруженным двукратным резервом $P_{O3}(t)$= $e^{-\lambda mt}+\alpha(1-e^{-\lambda mt})[1-(1-2e^{-\lambda mt}+e^{-2\lambda mt})]= e^{-\lambda mt}(1+2\alpha-3\alpha\cdot e^{-\lambda mt}+\alpha\cdot e^{-\lambda mt})$

$$\lambda_{O3}(t) = -\frac{m\lambda\cdot(1+2\alpha-6\alpha\cdot e^{-\lambda mt}+2\alpha\cdot e^{-2\lambda mt})}{1+2\alpha-3\alpha\cdot e^{-\lambda mt}+2\alpha\cdot e^{-2\lambda mt}}$$

$$T_{O3} = \frac{1+2\alpha}{m\lambda} - \frac{3\alpha}{2m\lambda} + \frac{\alpha}{3m\lambda} = \frac{6+5\alpha}{6m\lambda}$$

При α=1

$$P_{O3}(t) = e^{-3m\lambda t} - 3e^{-2m\lambda t} + 3e^{-\lambda mt};\ \lambda_{O3}(t) = \frac{m\lambda[6e^{-\lambda mt} - 2e^{-\lambda t} - 3]}{e^{-2m\lambda t} - 3\alpha\cdot e^{-\lambda mt} + 3}$$

$$T_{O3} = \frac{1}{3m\lambda} - \frac{3}{2m\lambda} + \frac{3}{m\lambda} = \frac{11}{6m\lambda}$$

При α=0 и *m*=1

$$P_{O3}(t) = e^{-\lambda t};\ \lambda_{O3}(t) = \lambda;\ T_{O3} = 1/\lambda$$

Графические зависимости отношения средней наработки до отказа резервированного объекта к средней наработке до отказа этого же объекта без резерва в зависимости от вероятности правильного обнаружения отказов α и однократном (*n*=1) или двукратном резервировании (*n*=2) показаны на рис. III.2.4

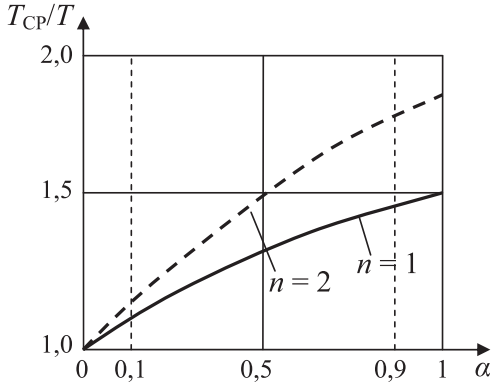


Рис. III.2.4. Зависимости относительной наработки до отказа объекта с резервом кратности n=1; 2 от эффективности обнаружения отказов при m=1

Чем ниже возможности средств обнаружения отказов, тем ниже эффективность резервирования, поскольку при низкой вероятности правильного обнаружения отказов исследуемые показатели резервирования кратности n=1 и n=2 отличаются между собой незначительно (см. рис. III.2.4, где m=1, $T_{СР}$ – средняя наработка до отказа резервированного объекта, T – средняя наработка до отказа объекта без резерва).

III.2.2.1. Раздельное постоянное резервирование

Для общего случая блок-схема раздельного постоянного резервирования показана на рис. III.2.2 а.

Рассмотрим частный случай, когда все основные и резервные элементы объекта однотипны. В каждой группе элементов объекта содержится n резервных элементов. Общее количество основных и резервных элементов в группе равно n+1. В этом случае имеют место следующие равенства:

$$P_{11}(t) = ... = P_{1(n+1)}(t) = P_1(t); P_{21}(t) = ... = P_{2(n+1)}(t) = P_2(t); ...;$$
$$P_{m1}(t) = ... = P_{m(n+1)}(t) = P_m(t)$$

а вероятность безотказной работы одной резервированной группы при идеальной эффективности контроля равна $P_j^n(t) = 1 - g_j^n(t), j = 1, 2, ..., m$

На основании формулы (2.1) при идеальной эффективности средств контроля с учетом того, что количество резервированных групп в объекте равно *m,* получим следующее выражение вероятности безотказной работы объекта

$$P_P(t) = \prod_{j=1}^{m} [1 - g_j^{n+1}(t)]$$

В более общем случае различного количества $R_j (j=1, 2, \ldots, m)$ резервных элементов в каждой группе и их разнотипности имеет место следующее выражение

$$P_P(t) = \prod_{j=1}^{m} [1 - \prod_{i=1}^{R_j} g_i(t)].$$

Если элементы всех групп объекта однотипны, то предыдущая формула преобразуется к виду

$$P_P(t)=[1-g^{n+1}(t)]^m \qquad (2.3)$$

Сравнение формул (2.1) и (2.3) при малой вероятности отказа элемента ($g(t)<0.1$) показывает, что вероятность отказа в случае раздельного резервирования уменьшается примерно в m^n раз.

$$\frac{1 - P_O(t)}{1 - P_P(t)} \approx \frac{g_m^{n+1}(t)}{m \cdot g^{n+1}(t)} \approx \frac{(1 - P^m(t))^{n+1}}{m \cdot g^{n+1}(t)} \approx \frac{((1 - 1 + mg(t))^{n+1}}{m \cdot g^{n+1}(t)} \approx m^n$$

Эта оценка впечатляет, но далека от реальности, поскольку сделана в предположении идеальной эффективности и надежности средств контроля.

Перейдем к более реальным оценкам вероятности безотказной работы объекта с раздельным постоянным резервом. Оставим без внимания реальную надежность средств контроля. Ограничимся

учетом главной характеристики эффективности средств контроля – вероятности правильного обнаружения отказов. Здесь опять-таки мы вынуждены акцентировать внимание на том обстоятельстве, что идеальный контроль исправности элементов не существует – возможна в зависимости от экономических затрат та или иная степень приближения вероятности α к единице. Итак, на основании формулы (2.2) можно установить, что

$$P_P(t) = \prod_{j=1}^{m} [P_j(t) + \alpha \cdot g_j(t)(1 - g_j^n(t)] \qquad (2.4)$$

Формула (2.4) получена в предположении, что внутри каждой из m резервированных групп одинаковое количество резервированных элементов. Если же принять, что все элементы объекта имеют одинаковую интенсивность отказа, то

$$P_P(t) = [P(t) + \alpha \cdot g(t)(1 - g^n(t)]^m \qquad (2.5)$$

В более общем случае разного количества $R_j(i=1,2,\ldots,m)$ резервных элементов в каждой группе и их разнотипности имеет место следующее выражение

$$P_P(t) = \prod_{j=1}^{m} [P_j(t) + \alpha \cdot g_j(t)(1 - g_j^{R_j}(t)] \qquad (2.6)$$

Пример 2.3. Примем $m=2$, резервирование раздельное. Требуется сопоставить общее и раздельное резервирование по показателю средней наработки до первого отказа объекта.

Решение.

1.Из примера (2.1) находим показатели вероятности безотказной работы и средней наработки до отказа объекта с общим однократным резервом двух составных элементов

$$P_{\text{O}}(t) = (1+\alpha) \cdot e^{-2\lambda t} - \alpha \cdot e^{-4\lambda t}; \; T_{\text{O}} = \frac{1+\alpha}{2\lambda} - \frac{\alpha}{4\lambda} = \frac{2+\alpha}{4\lambda}.$$

2. По формуле (2.5) находим вероятность безотказной работы объекта с раздельным резервированием кратности n=1 двух элементов

$$P_{\text{Р}}(t) = [e^{-\lambda t} + \alpha \cdot e^{-\lambda t}(1 - e^{-\lambda t})]^2 =$$
$$= e^{-2\lambda t}[1 + 2\alpha(1 - e^{-\lambda t}) + \alpha^2(1 - 2e^{-\lambda t} + e^{-2\lambda})]$$

3. Определяем среднюю наработку до отказа объекта с раздельным резервом

$$T_{\text{Р}} = \frac{1}{2\lambda} + \frac{\alpha}{\lambda} - \frac{2\alpha}{3\lambda} + \frac{\alpha^2}{2\lambda} - \frac{2\alpha^2}{3\lambda} + \frac{\alpha^2}{4\lambda} = \frac{6 + 4\alpha + \alpha^2}{12\lambda}$$

4. Построим и сопоставим графики нормированной средней наработки до отказа объекта с общим и раздельным резервом (рис. III.2.5). Нормировка осуществляется путем вычисления отношений $T_{\text{О}}/T(m=2)$ и $T_{\text{Р}}/T(m=2)$, где $T(m=2)=1/2\lambda$.

Полученные в примере 2.3 результаты свидетельствуют о том, что при малой кратности резервирования и малом числе резервируемых элементов раздельное резервирование имеет некоторое преимущество перед общим резервированием по показателю средней наработки до отказа.

Графические зависимости отношения средней наработки до отказа объекта с общим и раздельным резервированием к средней наработке до отказа этого же объекта без резерва в зависимости от вероятности правильного обнаружения отказов α и вида резервирования показаны на рис. III.2.5.

Преимущество раздельного резервирования перед общим резервированием становится существенным по мере роста эффективности средств контроля. Целесообразно изучить этот вопрос в условиях большой кратности резервирования и при большом количестве составных элементов объекта. Однако и при этих условиях надеяться на значительное преимущество

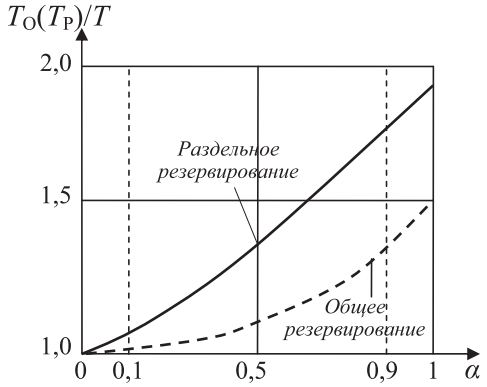


Рис. III.2.5. Зависимости относительной наработки до отказа объекта с однократным резервом от эффективности обнаружения отказов при общем и раздельном резервировании и при m=2

раздельного резервирования перед общим резервированием вряд ли следует, поскольку объем средств контроля при раздельном резервировании возрастает по сравнению с общим резервированием в m раз. Это может привести к значительному увеличению объема контрольного оборудования и в результате к снижению его надежности и достоверности получаемых результатов контроля.

III.2.3. Структурное резервирование без восстановления. Резервирование замещением

III.2.3.1.Проблема скрытых отказов при структурном резервировании

Любой отказ вычислительного объекта (вычислительного модуля, элемента) в информационной системе (ИС) – скрытый. Длительность его случайна и сверху ограничена. Это следует из того очевидною факта, что, с одной стороны, ни один отказ не обнаруживается мгновенно, а с другой стороны, отказ не может существовать бесконечно долго.

При отсутствии временной избыточности отказ рабочего элемента объекта всегда приводит к отказу системы. Это в равной мере относится к гипотетическим системам, в которых условно достигнуты сколь угодно малые длительности обнаружения и локализации отказов элементов. Действительно, в системах без резерва времени структурное резервирование объектов не защищает от их отказов, поскольку случайное время v приспособления ИС к отказам составных объектов включает в себя следующие основные четыре составляющие:

$\tau_{ОБН}$ – время обнаружения факта отказа (примерно половина интервала времени между контролями плюс длительность контроля);

$\tau_{ЛОК}$ – время локализации отказавшего элемента и принятия решении на выбор соответствующего способа устранения;

$\tau_{ПЕР}$ – время передачи операций и операндов от отказавшего элемента к исправному элементу;

$\tau_{ВОС}$ – время восстановления информационного процесса с помощью подключенного исправного элемента.

Здесь $v=\tau_{ОБН}+\tau_{ЛОК}+\tau_{ПЕР}+\tau_{ВОС}$.

Обозначим резерв времени как допустимое время $\tau_Д$ перерыва в работе ИС. Если резерв времени отсутствует, то $\tau_Д=0$. Очевидно, что справедливо неравенство $v>\tau_Д$ (поскольку $v>0$) и отказы любых работающих составных объектов вызовут отказы ИС – они не могут парироваться только структурной избыточностью.

Есть ли польза от структурного резервирования ИС в условиях, когда резерв времени сколь угодно мал, т.е. $\tau_Д \to 0$? В отношении обеспечения безотказности нет никакой пользы, но в части повышения готовности системы может быть достигнут определенный эффект. Это объясняется следующим. Время передачи операций и операндов отказавшего элемента объекта избыточному исправному много меньше времени его ремонта или

замены запасным. При значительном количестве резервных исправных элементов интенсивность потока восстановлений возрастает. Следовательно, среднее время до отказа ИС со структурным резервированием остается таким же, как и у полностью не избыточной ИС, тогда как среднее время до восстановления может быть существенно меньшим. Из этих рассуждений следует, что без временного резерва в ИС структурное резервирование составных объектов улучшает готовность, но не повышает безотказность системы.

Обычно в ИС имеется естественный резерв времени и, следовательно, выполняется неравенство $\tau_Д>0$. В этих условиях уже есть практическая возможность регулировать уровень безотказности ИС даже при отсутствии структурной избыточности. Покажем это, предварительно обозначив $\tau_{ОБН}+\tau_{ЛОК}=t_{ОБН}$ – время обнаружения отказа рабочего элемента, а $\tau_{ПЕР}+\tau_{ВОС}=t_У$ – время устранения отказа объекта. Если время адаптации ИС к отказу объекта меньше или равно допустимому времени ($v_i \le \tau_Д$), то этим событием в расчетах характеристик безотказности ИС можно пренебречь. В противоположном случае будет зафиксирован отказ ИС. Чтобы повысить безотказность системы нужно было бы, с одной стороны, увеличивать допустимое время $\tau_Д$, а с другой стороны, уменьшать время обнаружения $t_{ОБН}$ и время устранения $t_У$ отказа каждого объекта с резервом. Поскольку ИС взаимодействует и управляет конкретными внешними объектами, характеризующимися определенной инерционностью, то следует рассматривать время $\tau_Д$ как заданное и не подлежащее никакой искусственной регулировке.

Вместе с тем в арсенале разработчиков ИС есть много способов уменьшения времени v адаптации резервированной системы к отказам составных объектов. Большинство из них направлены на сокращение времени устранения отказов путем применения известных схем структурного резервирования. На рис. III.2.6

проиллюстрирована эта возможность путем переключения с отказавшего основного элемента на резервный (безотказная работа резервного элемента объекта отмечена на рисунке светлым цветом, а основного элемента – темным цветом). На данном рисунке обозначены символами v_i и v_i^* соответственно время адаптации к отказу объекта информационной системы без структурного резервирования и со структурным резервированием, а символом t_{yi}^* – время устранения отказа элемента объекта с помощью резервного элемента. Структурное резервирование позволяет значительно уменьшить длительность устранения отказа объекта ($t_{yi}^* < t_{yi}$) и довести это время до рамок, определенных длительностью допустимого перерыва в работе.

Вместе с тем длительность существования скрытого отказа ($t_{\text{ОБН}i}$) при традиционном структурном резервировании остается без изменения. Во многих ИС, работающих в реальном масштабе времени, резерв времени $\tau_{\text{Д}}$ обычно незначителен и поэтому многие скрытые отказы не удается парировать с помощью традиционных способов временного или структурно-временного

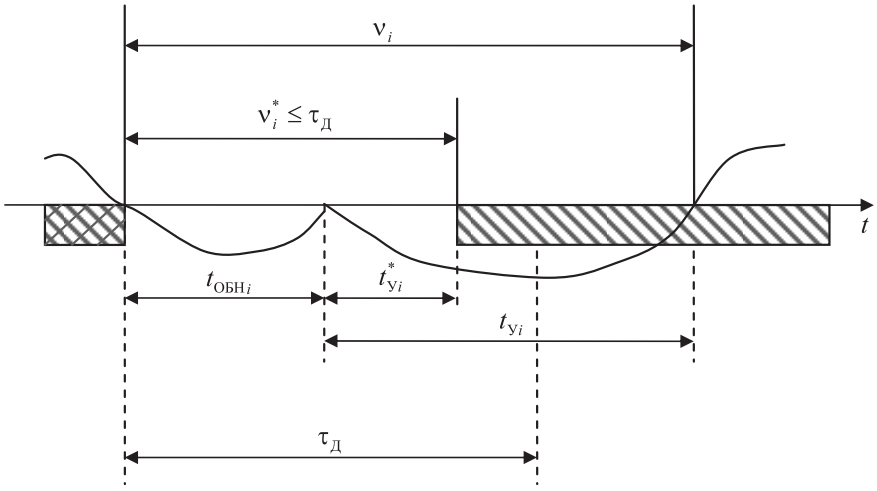


Рис. III.2.6. Временная диаграмма случайной длительности адаптации к i-ому отказу резервированного объекта ИС

резервирования. Поясним это на примере управляющих микропроцессорных ИС.

Аппаратные средства контроля микропроцессоров имеют ограниченные возможности. Для обнаружения сбоев и отказов периодически применяют программно-логический и тестовый контроль. Допустим, что за счет этого производительность микропроцессорных ИС снижена на 20%. Это означает, что в среднем каждый пятый цикл управления отведен для контроля работоспособности вычислительного модуля (ВМ) и, следовательно, каждый вычислительный модуль контролируется через четыре цикла управления.

Если же его отказ возник в следующем цикле после контроля, то будет обнаружен только через четыре управления. В случае возникновения отказа в предшествующем контролю цикле он будет существовать в скрытом состоянии в течение одного цикла управления. Учитывая вероятность отказа объекта в любом цикле управления между контролями, можно принять в среднем длительность скрытого отказа равной двум циклам управления. Допустим равенство этого интервала времени значению $\tau_д$. Отсюда следует, что длительность обнаружения примерно каждого второго отказа объекта превышает уровень $\tau_д$. В условиях данного примера даже при значительных временных затратах на программно-логический и тестовый контроль в управляющих микропроцессорных ИС удается с помощью традиционных способов структурно-временного резервирования защитить систему только от 50% отказов составных объектов. Для обеспечения гарантированного своевременного правильного обнаружения и локализации отказов требуется резко увеличить затраты вычислительных ресурсов на программный контроль, что приведет к снижению производительности ИС не менее, чем на (30 – 50)%. Для большинства информационно – управляющих систем такой путь борьбы со скрытыми отказами неприемлем.

Другая стратегия организации средств обнаружения отказов применяется в информационных системах, содержащих кластерную структуру. В этих ИС процессоры и модули памяти в кластерах содержат разветвленную систему аппаратных средств контроля. Из-за большой информационной нагрузки и жестких директивных сроков решения задач управления программные средства контроля привлекаются значительно реже, чем микропроцессорных системах, что обеспечивает и существенно меньшую длительность обнаружения отказов.

В качестве полной характеристики длительности v существования отказа ИС целесообразно рассматривать функцию распределения

$F_v(t)=P\{v\le t\}$, или же плотность распределения $f_v(t)=\dfrac{dF_v(t)}{dt}$.

Случайное время $v>0$ существования отказа ИС можно также представлять как время обнаружения и устранения отказа составного объекта системы.

Количественной мерой своевременности устранения отказа объекта является вероятность

$$\varphi = P\{\eta \le \tau_{Д} - t_{ОБН}\} = \int_0^{\tau_{Д}-t_{ОБН}} f_\eta(t)dt = F_\eta(\tau_{Д} - t_{ОБН}),$$

где $t_{ОБН}$ – время, затраченное на обнаружение отказа с вероятностью α, которая оценивается как:

$$\varphi = P\{\eta \le \tau_{Д} - t_{ОБН}\} = \int_0^{\tau_{Д}-t_{ОБН}} f_\eta(t)dt = F_\eta(\tau_{Д} - t_{ОБН}).$$

Причем, $v=\eta+\vartheta$.

В целом результирующая вероятность правильного и своевременного обнаружения отказа и устранения отказа элемента объекта может рассчитываться как произведение вероятностей независимых событий

$$\gamma=\alpha\cdot\varphi. \qquad (2.7)$$

Если процедуры обнаружения и устранения отказа зависимы и реализуются комплексно с помощью технологии адаптивной отказоустойчивости (см. главу III.3), то вместо вероятности γ следует для оценки эффективности успешного обнаружения отказа применять вероятность успешной адаптации системы к отказам составных объектов β, т.е.

$$\beta = P\{\nu \le \tau_{Д}\} = \int_0^{\tau_{Д}} f_{\nu}(t)dt = F_{\nu}(\tau_{Д})$$

или

$$\beta = P\{\nu \le \tau_{Д} - t_{У}\} = \int_0^{\tau_{Д}-t_{У}} f_{\nu}(t)dt = F_{\nu}(\tau_{Д} - t_{У}),$$

поскольку при реализации алгоритмов адаптивной отказоустойчивости время устранения обычно постоянно

К сожалению, встроенные средства аппаратного и программного контроля составных объектов информационных систем (ИС) не обеспечивают сколь угодно высокую эффективность правильного обнаружения отказов. Это объясняется как малыми значениями времени $\tau_{Д}$ в управляющих ИС, так и практическими ограничениями в применении встроенных средств контроля. Эти ограничения обусловлены следующим. Программные способы обнаружения отказов объектов связаны с недопустимыми затратами ресурса времени ИС, а аппаратные контроли – с большим объемом дополнительного оборудования. Резкое увеличение объема аппаратуры ИС вызывает значительное повышение интенсивности отказов системы. По этим причинам применяется сочетание аппаратных и программных средств контроля с акцентом на тех или иных средствах в зависимости от типа информационной системы. Такие компромиссные решения естественно затрудняют возможности повышения вероятности γ.

III.2.3.2. Общее резервирование замещением

При *нагруженном* резервировании все резервные элементы объекта функционируют абсолютно так же, как и основные (иногда для повышения наглядности того, что резервный элемент находится в состоянии выполнения всей работы основного элемента, этот вид резервирования называют «горячим резервом»). Если резервные элементы находятся в облегченном режиме функционирования, то такое резервирование называют *облегченным*. В том случае, когда резервные элементы выключены, находятся в режиме ожидания потребности в них в случае отказов основных элементов, то такой вид резервирования называют *ненагруженным* (не стандартизованное название – «холодный резерв»). В зависимости от вида перечисленного резервирования интенсивность отказов резервного элемента связана с интенсивностью отказов основного элемента соотношением $\lambda_P = k\lambda$, где $0<k\leq 1$, λ – интенсивность отказов основного элемента объекта с резервом. Это соотношение охватывает все три перечисленные выше виды нагруженности резервных элементов. В случае $k=1$ интенсивности отказов основных и резервных элементов одинаковы – резерв нагруженный. При промежуточном значении коэффициента k предполагается облегченный резерв. Однако в этом случае следует определить меру «облегченности» и оценить значение коэффициента k.

Объект с общим ненагруженным резервом описывается следующей моделью. Основная группа состоит из m в общем случае разнотипных элементов. Вероятность безотказной работы $P_m(t) = \prod_{j=1}^{m} P_j(t)$; где $P_j(t)$ – вероятность безотказной работы j-го основного группы.

Кроме основной группы в объекте содержится n-1 идентичных резервных групп. Предполагается, что отказывать могут

только элементы основной группы. Предполагается также, что надежность средства контроля составного элемента объекта намного выше надежности этого элемента, т.е. $1–P_k(t)<<1–P(t)$ и, следовательно, ненадежностью средств контроля при предварительном анализе можно пренебречь. При отказе любого элемента основной группы вся основная группа замещается резервной. Это обеспечивает безотказность объекта при следующих условиях:

– отказ правильно обнаружен (отсутствует ложное обнаружение), время его обнаружения и время переключения основной группы на резерв меньше допустимого, что характеризуется вероятностью успешного перехода на резерв γ;

– в резерве сохранилась хотя бы одна исправная резервная группа.

При выводе формулы вероятности безотказной работы объекта будем полагать, что интенсивности отказов всех элементов основной группы постоянны и вероятность безотказной работы j-го элемента основной группы и вероятность безотказной работы всей основной группы определяются следующими формулами:

$$P_j(t)=\exp(-\lambda_j t);\ P_m(t)=\exp(-\sum_{j=1}^{m}\lambda_j t).$$

Вероятность того, что основная группа элементов объекта за время t откажет ровно k раз (с учетом замен его резервными при условии успешного перехода на резерв k раз) определяется на основании закона Пуассона

$$P_t(z=k)=\gamma^k\frac{(\sum_{j=1}^{m}\lambda_j t)^k}{k!}\exp(-\sum_{j=1}^{m}\lambda_j t),$$

где z – случайное число отказов элементов за время t.

Вероятность безотказной работы объекта с ненагруженным резервом есть суммарная вероятность того, что не произойдет ни одного отказа элементов основной группы (k=0), либо произойдет только один отказ (k=1), либо дважды из-за отказов элементов основной группы потребуется замещать ее резервными группами и т.д. до полного использования n-1 резервных групп:

$$P_{\text{ОЗ}}(t) = \sum_{k=0}^{n} \gamma^k \frac{(\sum_{j=1}^{m} \lambda_j t)^k}{k!} \exp(-\sum_{j=1}^{m} \lambda_j t)$$

В частности, для дублирования (n=1) вероятность безотказной работы можно рассчитать по формуле:

$$P_{\text{ОЗД}}(t)=e^{-\Lambda t}(1+\gamma\cdot\Lambda\cdot t), \text{ где } \Lambda = \sum_{j=1}^{m} \lambda_j$$

С помощью данной формулы возможна верхняя оценка вероятности безотказной работы объекта с общим ненагруженным резервом. На практике следует ожидать менее высокий уровень безотказности объекта.

Объект с общим нагруженным или облегченным резервом описывается следующей моделью.

Предполагается, так же, как и в случае ненагруженного резерва, что кроме основной группы в объекте содержится nидентичных резервных групп.

Условия безотказности объекта:

– отказ правильно обнаружен (отсутствует ложное обнаружение), время его обнаружения и время переключения основной группы на резерв меньше допустимого, что характеризуется вероятностью успешного перехода на резерв γ;

– в резерве сохранилась хотя бы одна исправная резервная группа.

При построении модели надежности резервированного объекта для наглядности примем, что все линейки однотипны и все элементы линейки представляются одним эквивалентным элементом с вероятностью безотказной работы $P(t)$.

Безотказная работа объекта в течение времени t будет иметь место при следующих условиях:

А- основной элемент исправен; *Б* – отказ основного элемента правильно обнаружен (отсутствует ложное обнаружение), время его обнаружения и время переключения основной группы на резерв меньше допустимого, что характеризуется вероятностью успешного перехода на резерв γ и в резерве сохранился хотя- бы один исправный элемент.

Суммируя указанные вероятности, находим, что

$$P_{\text{О3}}(t)=P(A)+P(Б)=P(t)+\gamma\cdot g(t)(1-g^{n}(t)) \qquad (2.8)$$

Формула (2.8) предназначена для оценки вероятности безотказной работы объекта с общим нагруженным резервом и замещением отказавшего основного элемента на общий нагруженный резерв.

Для расчетов надежности объекта с общим облегченным резервом можно рекомендовать следующую формулу:

$$P_{\text{О3}}(t)=P(t)+\gamma\cdot g(t)(1-g_1^{n}(t)) \qquad (2.9)$$

где $g_1(t)$ – вероятность отказа эквивалентного резервного элемента объекта, находящегося в облегченном режиме. Например, при экспоненциальном законе распределения отказов, вычитая из вероятности $g(t)=1-e^{-\lambda t}$ вероятность $g_1(t)=1-e^{-\nu\lambda t}$, $0<\nu\le 1$, убеждаемся, что $g_1(t)< g(t)$, поскольку $e^{-\lambda t}>e^{-\nu\lambda t}$. Здесь под символом ν подразумевается уровень нагруженности резервного элемента. Если $\nu=1$, то резерв нагруженный. В противном случае имеет место облегченный резерв.

Рассмотрим пример.

Пример 2.4. Пусть m=1, n=1. Требуется определить показатели надежности объекта с однократным резервом в зависимости от нагруженности резерва при условии, что вероятность успешного перехода на резерв принимает значения γ=0; 0,5; 1.

Решение.

1. Из (2.9) получим

$$P_{\text{ОЗ}}(t) = e^{-\lambda t} + e^{-\nu\lambda t}\gamma(1-e^{-\lambda t}) = \gamma \cdot e^{-\nu\lambda t} + e^{-\lambda t}(1-\gamma \cdot e^{-\nu\lambda t}).$$

Пользуясь соотношениями между показателями безотказности, определяем

$$\lambda_{\text{ОЗ}}(t) = -\frac{P'_{\text{ОЗ}}(t)}{P_{\text{ОЗ}}(t)} = -\frac{\lambda e^{-\lambda t} + \gamma\nu\lambda \cdot e^{-\nu\lambda t} - (1+\nu)\gamma\lambda \cdot e^{-(1+\nu)\lambda t}}{\gamma \cdot e^{-\nu\lambda t} + e^{-\lambda t}(1-\gamma \cdot e^{-\nu\lambda t})} =$$

$$= \frac{\lambda \cdot [e^{-\lambda t} + \gamma \cdot e^{-\nu\lambda t}(\nu - (1+\nu)e^{-\lambda t})]}{\gamma \cdot e^{-\nu\lambda t} + e^{-\lambda t}(1-\gamma \cdot e^{-\nu\lambda t})}$$

$$T_{\text{ОЗ}} = \int_0^\infty P_{\text{ОЗ}}(t)dt = \frac{\gamma}{\nu\lambda} + \frac{1}{\lambda} - \frac{\gamma}{(1+\nu)\lambda} = \frac{\nu(1+\nu)+\gamma}{\nu(1+\nu)\lambda} = \frac{1}{\lambda} + \frac{\gamma}{\nu(1+\nu)\lambda},$$

Если резерв находится в облегченном режиме так, что интенсивность отказов резерва, например, вдвое меньше интенсивности отказов основного элемента, то имеют место следующие отношения:

$$\gamma = 0 : T_{\text{ОЗ}} / T(m=1) = 1,$$
$$\gamma = 0{,}5 : T_{\text{ОЗ}} / T(m=1) = 1{,}67,$$
$$\gamma = 1 : T_{\text{ОЗ}} / T(m=1) = 2{,}33,$$

где $T(m$=1) – средняя наработка до отказа объекта без резерва. Заметим, что в нагруженном режиме средняя наработка до от-

каза резервированного объекта превышает наработку до отказа нерезервированного объекта не более чем в 1,5 раза даже при вероятности успешного перехода на резерв на уровне γ=1, т.е. облегченный режим работы резервного элемента может дать существенный выигрыш в надежности, если на практике такая возможность может быть реализована.

III.2.4. Мажоритарное резервирование

III.2.4.1. Мажоритарный объект

Мажоритарной называется система выборов, при которой в число избранных попадают лишь кандидаты партии, получившей большинство голосов в данном округе. Этот подход применяется и в технике для повышения надежности ответственных объектов. Так, в бортовых информационных системах широко применяется трехкратная структурная избыточность объектов в сочетании с восстанавливающим органом (ВО). На этой основе создается троированный мажоритарный объект (ТМО) с ВО, который определяет выходные результаты по *большинству голосов (мажоритарным методом)*. В результате реализуется логика 2/3 (совпадение хотя бы двух выходных результатов из трех при трех параллельно работающих однотипных устройствах). В этом случае при условии идеальной надежности восстанавливающего органа вероятность безотказной работы и вероятность отказа ТМО соответственно равны

$$P_{\text{ТМО}}(t) = P^3(t) + 3P^2(t)G(t);\; G_{\text{ТМО}}(t) = 3P(t)G^2(t) + G^3(t),$$

где $P(t)$ и $G(t)$ – вероятности безотказной работы и вероятности отказа любого одного из трех параллельно работающих устройств.

В общем случае резервирования с логикой n/m, когда $n \geq 2$ и $n<m$ вероятность безотказной работы N-го мажоритарного объекта (NМО) имеет следующий вид

$$P_{N\text{МО}}(t) = P^m(t) + \binom{m}{1} P^{m-1}(t)G(t) + \binom{m}{2} \cdot P^{m-2}(t)G^2(t) + \ldots$$
$$\ldots + \binom{m}{i} \cdot P^{m-i}(t)G^i(t) + \ldots + \binom{m}{n} \cdot P^n(t)G^{m-n}(t) = \qquad (2.11)$$
$$= \sum_{i=0}^{n} \binom{m}{i} \cdot P^{m-i}(t)G^i(t)$$

При экспоненциальном законе распределения отказов с интенсивностью λ показатели безотказности ТМО имеют вид

$$P_{\text{ТМО}}(t) = 3e^{-2\lambda t} - 2e^{-3\lambda t};\ \lambda_{\text{ТМО}}(t) = \frac{6\lambda(1-e^{-\lambda t})}{3-2e^{-\lambda t}};\ T_{\text{ТМО}} = \frac{5}{6\lambda}$$

Приведенные выше значения показателей надежности мажоритарного объекта показывают, что мажоритарное резервирование может снизить его безотказность по отношению к исходному нерезервированному объекту. Вместе с тем, этот вид резервирования повышает достоверность передаваемой информации и, следовательно, повышает функциональную надежность исходного объекта.

III.2.4.2. Восстанавливающий орган

Восстанавливающий орган (ВО) может реализоваться аппаратно или программно. Как правило, предпочтение отдается программному варианту реализации ВО, как более дешевому и гибкому варианту построения мажоритарного объекта. Вследствие простоты логики построения программы ВО есть возможность практически исключить алгоритмичес-

кие и программные ошибки. По данной причине у некоторых специалистов создается иллюзия абсолютной надежности ВО. При этом упускается из виду то очень существенное обстоятельство, что функциональная надежность ВО и мажоритарного объекта в целом определяется не только ошибками в программе, но и, главным образом, сбойными ошибками, которые возникают значительно чаще отказов аппаратуры и могут существенно понизить эффективность мажоритарного резервирования.

По существу восстанавливающий орган представляет собой для мажоритарной логики 2/3 программно реализованную структуру, показанную на рис. III.2.7.

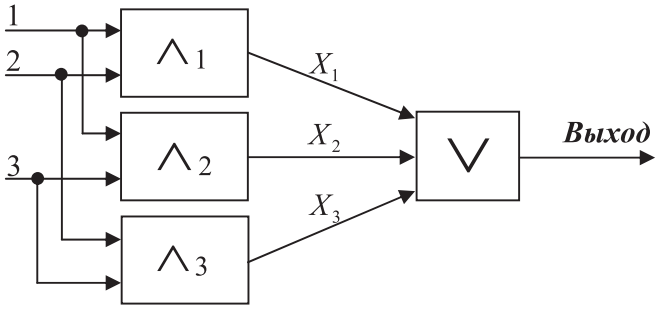


Рис. III.2.7. Мажоритарная логика 2/3

С помощью микроопераций $\wedge_1$, $\wedge_2$, $\wedge_3$ производится поразрядное сравнение выходных результатов работы цифровых однотипных устройств 1, 2 и 3. Количество бит в выходном результате обычно колеблется в пределах от 8 до 128. Эти цифры показывают количество сравнений, которое нужно однократно выполнить каждой из трех микроопераций. В результате сравнений формируются 3 одиночных сигнала X_i, каждый из которых представляет собой код 1 или 0. Модуль с функцией голосования V формирует на выходе 1 или 0, в зависимости от большинства логических сигналов на его входе.

Построим модель надежности мажоритарного объекта с восстанавливающим органом с логикой 2/3 в предположении, что микрооперации сравнения результатов работы цифровых однотипных устройств находятся под воздействием сбойных ошибок аппаратуры объекта и выполняются с вероятностью правильной однократной работы p. При этом для оценки сверху принимается, что алгоритмические и программные ошибки в микрооперациях отсутствуют и микрооперация дизъюнкции (рис. III.2.7) также выполняется безошибочно вследствие большой избыточности по кодам 1 [2].

Вероятность функционального отказа мажоритарного объекта в момент времени t_i, i=0, 1, 2, … определяется суммой вероятностей следующих событий:

– $G(A)=G^3(t_i)+3G^2(t_i)P(t_i)$ – вероятность того, что все три устройства отказали или отказали два любых из трех устройств (при этом состояние надежности восстанавливающего органа не оказывает влияния);

– $G(Б)=P^3(t_i)(g^3+3g^2p)$ – вероятность того, что исправны все три устройства, но неправильно выполнены все три микрооперации восстанавливающего органа, или хотя бы две микрооперации из трех;

– $G(B)=3P^2(t_i)G(t_i)(g^3+3g^2p+3gp^2)$ – вероятность того, что при условии исправности двух любых из трех устройств неправильно выполнены либо все три микрооперации или две из трех или хотя – бы одна из трех.

Вероятность отсутствия функционального отказа в работе мажоритарного объекта в момент времени t_i равна

$$P_{\text{МО}}(t_i) = 1 - G(A) - G(Б) - G(B) =$$
$$= P^2(t_i)[1 - g^2(1+2p) + G(t_i)p^2(2p-3g)]$$

Здесь подразумевается, что разность между соседними моментами времени $\Delta t = t_i - t_{i-1}$ есть такт обработки информации в системе.

Следовательно, вероятность функционального отказа в течение такта однократной обработки информации равна

$$P_{\text{МО}}(\Delta t) = P^2(\Delta t)[1 - g^2(1+2p) + G(\Delta t)p^2(2p-3g)] \quad (2.12)$$

Функция распределения времени до функционального отказа с учетом геометрического распределения ошибок в выполнении микроопераций восстанавливающего органа [2] имеет следующий вид: $F(n)=1-P^n_{\text{МО}}$. При этом математическое ожидание времени до функционального отказа мажоритарного объекта определяется в виде

$$T_{\text{МОф}} = \frac{\Delta t}{1 - P_{\text{МО}}(\Delta t)} \quad (2.13)$$

Учитывая, что геометрическое распределение есть дискретный аналог экспоненциального распределения, можно выразить интенсивность функциональных отказов мажоритарного объекта через величину, обратную средней наработке до его функционального отказа при высоком быстродействии цифрового устройства ($\Delta t \to 0$)

$$\lambda_{\text{МОф}} = \frac{1 - P_{\text{МО}}(\Delta t)}{\Delta t} = \frac{1 - P^2(\Delta t)[1 - g^2(1+2p) + G(\Delta t)p^2(2p-3g)]}{\Delta t}.$$

Для сравнения определим показатели функциональной надежности исходного объекта (цифрового устройства информационной системы) без резерва, имея в виду, что результат работы устройства представляет собой массив информации, который в каждом такте обработки информации считывается с помощью программы. Вероятность безотказной работы устройства в течение длительности такта $P(\Delta t)$. Вероятность правильного однократного считывания массива информации определяем в виде $p_1=p^{1/3}$. Эта связь с показателем правильного однократного

выполнения микрооперации объясняется тем, что при выполнении микрооперации считываются массивы информации с двух устройств, после чего эти массивы сравниваются между собой. В результате при выполнении микрооперации выполняются три примерно равных действия считывания массива информации.

Отсюда вероятность функционального отказа в течение такта однократной обработки информации нерезервированным объектом равна

$P_{\text{О}}(\Delta t)=P(\Delta t)p^{1/3}$. Другие два показателя функциональной безотказности имеют следующий вид:

$$T_{\text{Оф}} = \frac{\Delta t}{1-P(\Delta t)p^{\frac{1}{3}}};\ \lambda_{\text{Оф}} = \frac{1-P(\Delta t)p^{\frac{1}{3}}}{\Delta t} \qquad (2.14)$$

Рассмотрим пример.

Пример 2.5. Время такта обработки информации $\Delta t=10^{-6}$ с. Показатели правильного однократного выполнения микрооперации сравнения результатов работы цифровых однотипных компонент находятся на уровне правильной однократной работы большой интегральной схемы и составляют согласно [2,19] $p=1–5\cdot10^{-13}$; $g=5\cdot10^{-13}$. Отказы составных компонентов обработки информации распределены по экспоненциальному закону с интенсивностью $\lambda=10^{-6}1/\text{ч}=0{,}3\cdot10^{-9}1/\text{с}$. Требуется вычислить показатели функциональной безотказности мажоритарного объекта и объекта без резерва и сравнить эти результаты.

Решение. Вначале определяют вероятности безотказной работы и вероятности отказа исходного объекта в течение такта обработки информации

$$P(\Delta t) = \exp(-\lambda\Delta t) = 1-0{,}3\cdot10^{-15}1/\text{с};$$

$$G(\Delta t) = 1-\exp(-\lambda\Delta t) = 0{,}3\cdot10^{-15}1/\text{с}.$$

По формуле (2.12) находят вероятность правильной однократной работы мажоритарного объекта $P_{МО}(\Delta t)\approx(1–1{,}2\cdot10^{-15})$. Затем по формуле (2.13) определяют среднюю наработку до функционального отказа, а затем и интенсивность функциональных отказов мажоритарного объекта

$$T_{МОф} = 8\cdot10^{8}\text{с} = 2\cdot10^{5}\text{ч};\ \lambda_{МОф} = 5\cdot10^{-6}1/\text{ч}.$$

По формуле (2.14) определяют среднюю наработку до функционального отказа, а затем и интенсивность функциональных отказов исходного нерезервированного объекта

$$P_{О}(\Delta t)\approx(1–3{,}2\cdot10^{-15});$$
$$T_{Оф} = 3{,}12\cdot10^{8}\text{с} = 0{,}87\cdot10^{5}\text{ч};\ \lambda_{МОф} = 11{,}5\cdot10^{-6}1/\text{ч}$$

Полученные результаты убедительно подтверждают справедливость высказанного ранее тезиса о том, мажоритарное резервирование с логикой 2/3 снижает структурную надежность (показатель средней наработки до структурного отказа $T_{О}=1/\lambda$ исходного объекта снижается до значения $T_{МО}=5/6\lambda$ у мажоритарного объекта). Однако большое достоинство мажоритарного резервирования в том, что оно существенно повышает функциональную надежность объекта (показатель средней наработки до функционального отказа мажоритарного объекта более чем в два раза превышает тот же показатель исходного нерезервированного объекта, т.е. $(T_{МОф}/T_{Оф})>2$).

При низкой вероятности правильного выполнения микроопераций сравнения результатов работы компонент ($p\to0$) функциональная надежность мажоритарного объекта ничтожно мала ($P_{МО}(\Delta t)\to0$) даже при высокой вероятности безотказной работы составных однотипных компонент объекта ($P(\Delta t)\to1$). Отсюда следует, что для обеспечения функциональной надежности ма-

жоритарного объекта необходимо построить высоконадежный восстанавливающий орган.

Для защиты от функциональных отказов может применяться мажоритарное резервирование восстанавливающего органа. С этой целью в состав объекта вводится еще одно логическое устройство (или программное действие), которое предназначено для анализа логических сигналов с выходов трех восстанавливающих органов. Это устройство (действие) проще восстанавливающих органов.

Недостаток мажоритарного резервирования заключается в нарушении алгоритма при отказе двух из трех компонент (для троичной мажоритарной схемы), а в общем случае при отказе n+1 компонент (для n – ированного мажоритарного объекта в информационной системе). При всегда нечетном числе компонент N=2n+1 объект еще сохраняет работоспособность при отказе n компонент; если же откажет еще хотя – бы один компонент, то объект теряет свои функциональные возможности, хотя имеется n исправных компонент.

III.2.4.1.Гибридное мажоритарное резервирование

Эта схема предназначена для повышения структурной надежности мажоритарного резервирования. Применяются следующие два подхода к реализации рассматриваемого типа резервирования.

Естественное гибридное мажоритарное резервирование основывается на применении адаптивных восстанавливающих органов, которые изменяют свой порог по мере выхода компонентов из строя. Например, при использовании мажоритарного резервирования с логикой 5/7 в случае накопления двух отказов компонент производится переход на логику 3/5, а затем и на логику 2/3.

Искусственное гибридное мажоритарное резервирование основывается на введении в состав мажоритарного объекта допол-

нительных однотипных резервных компонент. Эти компоненты не подключены к восстанавливающему органу. Они предназначены для замещения отказавших компонент из состава исходного мажоритарного объекта. На рис. III.2.8 приведен пример искусственного гибридного ТМО, содержащего три рабочих компонента и один однотипный резервный компонент

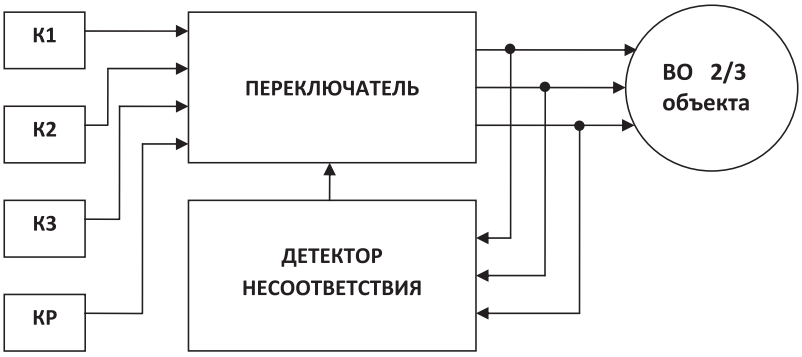


Рис. III.2.8. Пример искусственного гибридного троичного мажоритарного объекта

В этом объекте кроме восстанавливающего органа (ВО) содержатся переключатель и детектор несоответствия, сравнивающий выходные сигналы сортирующей схемы с сигналами каждого отдельного компонента.

Показатели структурной надежности этого объекта определяются на основании критерия отказа объекта, который определяется так. При трех основных и одном резервном однотипном компоненте обработки информации структурный отказ объекта наступает при наличии двух отказавших компонент или при наличии одного исправного основного компонента и исправного резервного компонента, но подключение резервного вместо отказавшего основного компонента не было успешным (либо по причине пропуска отказа, либо по причине отказа переключающего устройства, либо из-за несвоевременного переключения на резерв).

Вероятность безотказной работы искусственного гибридного троичного мажоритарного объекта есть сумма вероятностей событий:

$P(A)=P^3(t)P(t)=P^4(t)$ -вероятность того, что исправны все три основных и один резервный компонент;

$P(Б)=P^3(t)g(t)$ – вероятность того, что исправны все три основных компонента и отказал резервный компонент;

$P(B)=3P^2(t)g(t)P(t)=3P^3(t)g(t)$ – вероятность того, что исправны два любых из трех основных компонент и исправен резервный компонент;

$P(Г)=3P^2(t)g(t)g(t)=3P^3(t)g^2(t)$ – вероятность того, что исправны два любых из трех основных компонент и отказал резервный компонент;

$P(Д)=3P(t)g^2(t)\cdot\gamma\cdot P(t)=3P^2(t)g^2(t)\cdot\gamma$ – вероятность того, что исправен только один любой из трех основных компонент и переключение на исправный резервный компонент мажоритарного объекта прошло успешно с вероятностью γ.

Таким образом, вероятность безотказной работы резервированных компонент троичного гибридного мажоритарного объекта может быть определена по формуле

$$P_{\text{ГМО}}(t) = P(A) + P(Б) + P(B) + P(Г) + P(Д) = \\ = P^3(t) + 3P^2(t)\cdot g(t)\cdot[1 + g(t)\cdot\gamma] \qquad (2.15)$$

Следовательно, повышение надежности группы компонентов обработки информации в гибридном мажоритарном объекте по сравнению с троичным мажоритарным объектом составляет по показателю вероятности безотказной работы

$$P_{\text{ГМО}}(t) - P_{\text{ТМО}}(t) = 3P^2(t)\cdot g^2(t)\cdot\gamma$$

Рассмотрим пример.

Пример 2.6. Отказы компонентов обработки информации в мажоритарном объекте распределены по экспоненциальному закону с интенсивностью λ. Требуется выполнить сравнительную оценку структурной надежности исходного троичного мажоритарного органа и гибридного мажоритарного объекта по показателю средней наработки до отказа.

Решение. По формуле (2.15) находим вероятность безотказной работы резервированной группы компонентов

$$P_{\text{ГМО}}(t) = e^{-2\lambda t}[3(1+\gamma) - 2e^{-\lambda t}(1+3\gamma) + 3\gamma \cdot e^{-2\lambda t}]$$

Определяем среднюю наработку до структурного отказа резервированного мажоритарного объекта

$$T_{\text{ГМО}} = \int_0^\infty P_{\text{ГМО}}(t)dt = \frac{10+3\gamma}{12\lambda}$$

При отсутствии возможности успешного перехода на резерв ($\gamma \to 0$) имеет место равенство значений

$$T_{\text{ГМО}} = T_{\text{ТМО}} = \frac{5}{6\lambda}$$

При успешных переходах на резерв ($\gamma \to 1$) средняя наработка до структурного отказа объекта может быть повышена до 30%. Действительно, при этих условиях $T_{\text{ГМО}}=(13/12\lambda)$ и $T_{\text{ГМО}}/\ T_{\text{ТМО}}=1{,}3$.

Рассмотренную схему построения надежного мажоритарного объекта можно распространить и на случай гибридного избыточного объекта (ГИО), содержащего N основных и S резервных компонент. Подобный объект обозначается как ГИО(N,S). Существенную часть ГИО составляют переключательные схемы, обеспечивающие отключение отказавших и включение резервных компонент. Это обстоятельство отрица-

тельно сказывается на эффективности гибридного мажоритарного резервирования.

Недостатком как аппаратного, так и программного способов мажорирования является значительное количество оборудования, даже в минимальном варианте при n=1 (троирование). Другой недостаток способов мажорирования – значительные потери производительности. При аппаратной реализации потеря производительности связана с необходимостью синхронизации процессов в резервированных каналах. При программной реализации быстродействие системы снижается из-за затрат времени на обмен информацией между каналами.

Причина такой неэффективности состоит в том, что и при аппаратной, и при программной организации механизм маскирования сбоев – т.е. голосование, определение неисправного канала, его блокирование и последующее включение в нормальную работу – используется в каждом такте работы системы вне зависимости от наличия или отсутствия сбоев. Эти временные потери при практической реализации достигают 30-50%. К недостаткам мажорирования при его реализации следует отнести также большое количество связей между каналами и значительные трудности при проектировании. Следует отметить, что при аппаратном мажорировании нормальное функционирование при деградации системы до одного канала обеспечивается лишь при дополнительных аппаратных и временных затратах. При программном мажорировании в случае отказов каналов реконфигурация до одного исправного канала возможна без дополнительных аппаратных затрат. Но невозможно увеличение кратности маскируемых сбоев в отличие от аппаратного мажорирования, где это можно осуществить путем организации многократного голосования при прохождении сигналов по системе или соответственно путем введения аппаратной избыточности.

III.2.5. Структурное резервирование с восстановлением

Резервирование с восстановлением представляет собой очень действенный инструмент построения надежных информационных систем. Вместе с тем, возможности восстановления резервированных объектов в системе зависят от следующих основных условий:

1) конструкция аппаратуры позволяет ремонтировать отказавшее устройство без остановки функционирования всего резервированного объекта;

2) в наличии имеются требуемые запасные элементы, приборы, технические средства, подготовлен персонал, ремонтные бригады, созданы необходимые условия для оперативного ремонта;

3) средства контроля (как встроенного, так и функционального) позволяют своевременно обнаружить и идентифицировать отказавшие устройства и элементы объекта.

В зависимости от уровня проработанности перечисленных условий восстановление в резервированном объекте следует разделить на два вида:

– оперативное восстановление (условия восстановления обеспечены полностью). Если в восстановлении отказа задействована одна ремонтная бригада, то такое восстановление принято называть ограниченным; более одной бригады – неограниченным;

– восстановление с задержкой. При этом имеют место следующие причины задержки: длительная доставка запасных элементов и/или ремонтников, аппаратура резервированного объекта неремонтопригодна в процессе функционирования, недостаточно эффективен контроль работоспособности объ-

екта, что приводит к большой длительности скрытого отказа. Задержка в восстановлении может быть случайной или детерминированной. Второй случай возможен при постоянном времени доставки на объект ремонтников или запасных элементов (устройств).

Для обеспечения наглядности построения и решения моделей надежности резервированных объектов с восстановлением ограничимся рассмотрением объекта, содержащего m основных и два резервных (k =2) однотипных устройства (кратность резервирования k/m). Резерв нагруженный, общий, замещением, восстановление ограниченное, встроенные средства контроля каждого составного устройства (компонента объекта) имеют ограниченные возможности в обнаружении отказов, существуют скрытые (необнаруженные в течение определенного времени) отказы резервных устройств. Предполагается, что средства контроля и реконфигурации резервированного объекта ИС идеально надежны, а обнаруженные отказы своевременно устраняются (ϕ=1, γ=α). Необнаруженные отказы основных компонентов объекта приводят к его отказам.

Задача состоит в определении показателей безотказности этого объекта, который содержит $S_k = S_2 = 1 + \frac{2(2+3)}{2} = 6$ работоспособных состояний. Действительно, при k=0 возможно только одно работоспособное состояние, т.е. S_0=1. При этом оба основных устройства объекта исправны. При k=1 приемлем правильно обнаруженный отказ любого одного основного или резервного устройства и допускается необнаруженный отказ резервного устройства. В этом случае количество работоспособных состояний S_1=S_0+1+1=3.

При k=2 кроме перечисленных выше для k=1 работоспособных состояний приемлемы также следующие три состояния объекта: состояние отказов любых двух составных устройств

(предполагается, что эти отказы правильно обнаружены); состояние, при котором имеет место сочетание правильно обнаруженного отказа основного устройства и необнаруженного отказа резервного устройства; состояние, при котором отказали оба резервных устройства и их отказы не обнаружены. Следовательно,

$$S_2=S_1+3=S_0+2+3=6$$

В работе [19] нами показано, что при произвольном количестве резервных устройств в объекте количество его рабочих состояний в модели безотказности объекта определяется следующим выражением

$$S_k = S_{k-1} + k + 1 = S_0 + \frac{k(k+3)}{2}.$$

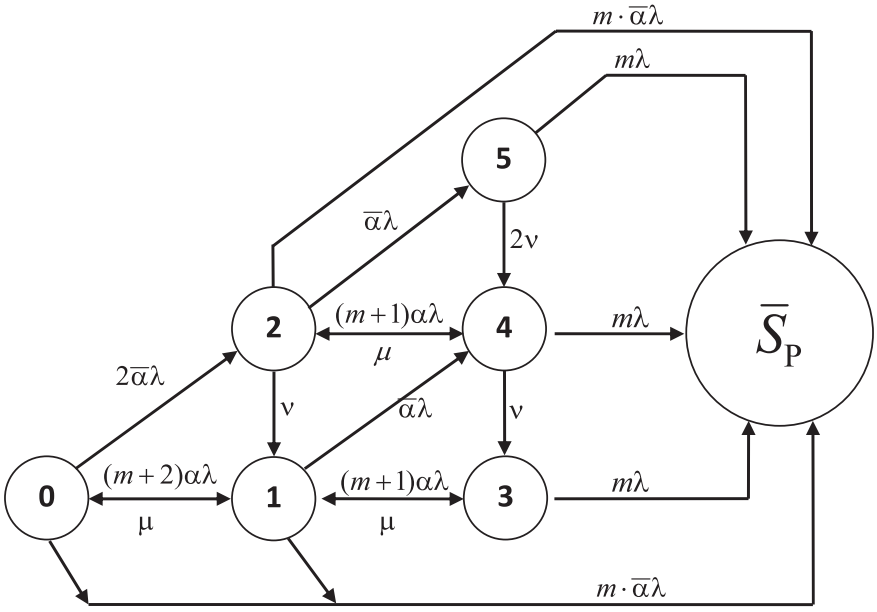


Рис. III.2.9. Граф безотказности резервированного объекта кратности 2/m.

На рис. III.2.9 построен граф безотказности рассматриваемого резервированного объекта, где вершинами определены следующие состояния:

0 – исправны m основных и оба резервных устройства объекта;

1 – отказало и восстанавливается одно устройство. Отказ своевременно обнаружен. Если отказало основное устройство, то оно замещается резервным;

2 – скрытый отказ одного резервного устройства, основные устройства исправны;

3 – отказали два основных устройства, резервные устройства исправны, одно из них замещает отказавшее основное устройство, второе основное устройство восстанавливается;

4 – отказало и восстанавливается одно основное устройство, одно резервное устройство в состоянии скрытого отказа;

5 – скрытый отказ двух резервных устройств, основные устройства исправны;

$\overline{S}_P$ – множество неработоспособных состояний резервированного объекта.

Ограничимся типичным для информационных систем экспоненциальным распределением случайного времени безотказной работы ξ устройства [1]. Примем также, что случайные величины времени его восстановления η и длительности скрытых отказов υ имеют экспоненциальные распределения. Все эти случайные величины взаимно независимы, а их распределения характеризуются интенсивностями λ, μ, ν соответственно.

Отказ объекта наступает при возникновении одного из следующих событий:

– отказали k+1=3 устройства. В этом случае объект переходит из состояний 4 и 5 в множество неработоспособных состояний $\overline{S}_P$ с интенсивностью $m\lambda$;

– отказали не более k =2 основных устройств, однако отказ любого одного из m устройств пропущен с вероятностью $\overline{\alpha}$.

Объект переходит из состояний 0, 1, 3 в множество неработоспособных состояний с интенсивностью $m \cdot \overline{\alpha}\lambda$;

– отказали не более k основных и резервных устройств или только k основных устройств. Объект переходит из состояния 2 в состояния отказа с интенсивностью $m \cdot \overline{\alpha} \cdot \lambda$.

Задача состоит в определении средней наработки до отказа $T_{\text{СР}}$ резервированного объекта, имея в виду, что начальным состоянием является состояние 0, в котором исправны все основные и резервные устройства.

Надежность рассматриваемого резервированного объекта моделируется с помощью Марковского случайного процесса. В работе [1] нами описан прямой графовый метод моментов и приведено утверждение, согласно которому, если надежность объекта моделируется с помощью графа состояний, полумарковского (в частном случае Марковского) случайного процесса на этом множестве состояний, то первый момент времени пребывания объекта в множестве работоспособных состояний при нулевом начальном состоянии определяется с помощью выражения

$$\bar{t}_0 = \frac{T_0 \Delta G^0_{\overline{S}_P} + \sum\limits_{(k)} \sum\limits_{0,i} l^{0i}_k T_i \Delta G^i_k}{\Delta G_{\overline{S}_P}}, \qquad (2.16)$$

где T_0 и T_i – математические ожидания безусловного времени пребывания в состояниях 0; $i \in S_P$; $\Delta G^0_{\overline{S}_P}$ вес разложения графа без вершин множества состояний отказа и начальной вершины; l^{0i}_k – $к$-й путь, ведущий из начального работоспособного состояния графа $0 \in S_P$ в состояние $i \in S_P$; ΔG^i_k – вес разложения графа без i-й вершины и вершин графа, расположенных на $к$-ом пути; $\Delta G_{\overline{S}_P}$ – вес разложения графа без вершин множества состояний отказа.

Поскольку первый момент времени пребывания объекта в заданном множестве состояний есть математическое ожидание времени его пребывания в этом множестве состояний, то имеет

место равенство средней наработки объекта до отказа и первого момента времени, определяемого формулой (2.16), т.е $T_{CP} = \bar{t}_0$.

Веса разложений графов в формуле (2.16) находят по формуле Мэзона (Мэйсона – *перевод*) [1]:

$$\Delta G = 1 - \sum_i C_i + \sum_{ij} C_i C_j - \sum_{ijk} C_i C_j C_k + \ldots$$

В этой формуле C_i, C_j, C_k и т.д. – это веса контуров на графе. Контуры не имеют общих вершин. C_i, C_j, C_k

На графе резервированного объекта, представленного на рис. III.2.10, имеются следующие контуры:

1: 02-21-10; **2**: 14-43-31; **3**: 14-42-21; **4**: 25-54-42; **5**: 02-25-54-43-31-10, **6**: 01-10; **7**: 13-31; **8**: 24-42, где 02, 21, 10 и т.д. – это пути на графе, связывающие две смежные вершины.

Заметим, что согласно теории графов пути на контуре, связывающие несколько вершин, не могут более одного раза проходить через каждую из этих вершин.

Веса перечисленных выше контуров определяются следующим образом:

$$C_1 = p_{02}p_{21}p_{10};\ C_2 = p_{14}p_{43}p_{31};\ C_3 = p_{14}p_{42}p_{21};\ C_4 = p_{25}p_{54}p_{42};$$
$$C_5 = p_{02}p_{25}p_{54}p_{43}p_{31}p_{10};\ C_6 = p_{01}p_{10};\ C_7 = p_{31}p_{31};\ C_8 = p_{24}p_{42}.$$

Прежде чем раскроем переходные вероятности и веса контуров на графе, убедимся, что помеченные переходы на графе не противоречат логике изменения надежности объекта при отказах и восстановлениях. Эту проверку сделаем путем суммирования интенсивностей выходов из каждой вершины графа:

0 $\Lambda_0 = (m+2)\alpha\lambda + 2\bar{\alpha}\lambda + m\bar{\alpha}\lambda = (m+2)\lambda$ – все устройства исправны;

1 $\Lambda_1 = (m+1)\alpha\lambda + \bar{\alpha}\lambda + m\bar{\alpha}\lambda + \mu = (m+1)\lambda + \mu$ – все основные и одно резервное устройства исправны, отказ основного обнаружен и это устройство восстанавливается;

2 $\Lambda_2 = (m+1)\alpha\lambda + m\bar{\alpha}\lambda + \bar{\alpha}\lambda + \nu = (m+1)\lambda + \nu$ – все основные и одно резервное устройства исправны, второе резервное устройство в состоянии скрытого отказа;

3 $\Lambda_3 = m\lambda + \mu$ – отказали два основных устройства, оба отказа обнаружены, они замещены резервными, но вследствие ограниченного восстановления ремонтируется только одно устройство;

4 $\Lambda_4 = m\lambda + \mu + \nu$ – отказали основное и резервное устройства, отказ основного устройства обнаружен,- устройство восстанавливается, отказ резервного устройства скрытый;

5 $\Lambda_5 = m\lambda + 2\nu$ – отказали два резервных устройства, оба отказа скрытые.

Функции распределении времени пребывания исследуемого объекта в каждом из перечисленных состояний определяются в виде:

$$F_i(t) = 1 - \exp(-\Lambda_i t), i = 0,1,...,5$$

Вероятности переходов в множестве работоспособных состояний имеют следующий вид

$$p_{01} = (m+2)\alpha\lambda\int_0^\infty dF_0(t) = (m+2)\alpha\lambda\int_0^\infty d[1-e^{-(m+2)\lambda\cdot t}] = \alpha;$$

$$p_{02} = 2\bar{\alpha}\lambda\int_0^\infty dF_0(t) = \frac{2\bar{\alpha}}{m+2};$$

$$p_{10} = \mu\int_0^\infty dF_1(t) = \mu\int_0^\infty d[1-e^{-[(m+1)\lambda+\mu]t}] = \frac{\mu}{(m+1)\lambda+\mu};$$

$$p_{13} = (m+1)\alpha\lambda\int_0^\infty dF_1(t) = \frac{(m+1)\alpha\lambda}{(m+1)\lambda+\mu}$$

$$p_{14} = \bar{\alpha}\lambda \int_0^\infty dF_1(t) = \frac{\bar{\alpha}\lambda}{(m+1)\lambda + \mu};$$

$$p_{21} = \nu \int_0^\infty dF_2(t) = \nu \int_0^\infty d[1 - e^{-[(m+1)\lambda + \nu]t}] = \frac{\nu}{(m+1)\lambda + \nu};$$

$$p_{24} = m\alpha\lambda \int_0^\infty dF_2(t) = \frac{m\alpha\lambda}{(m+1)\lambda + \nu}; \; p_{25} = \bar{\alpha}\lambda \int_0^\infty dF_2(t) = \frac{\bar{\alpha}\lambda}{(m+1)\lambda + \nu};$$

$$p_{31} = \mu \int_0^\infty dF_3(t) = \mu \int_0^\infty d[1 - e^{-(m\lambda + \mu)\cdot t}] = \frac{\mu}{m\lambda + \mu}$$

$$p_{42} = \mu \int_0^\infty dF_4(t) = \mu \int_0^\infty d[1 - e^{-(m\lambda + \mu + \nu)t}] = \frac{\mu}{m\lambda + \mu + \nu};$$

$$p_{43} = \nu \int_0^\infty dF_4(t) = \frac{\nu}{m\lambda + \mu + \nu};$$

$$p_{54} = 2\nu \int_0^\infty dF_5(t) = 2\nu \int_0^\infty d[1 - e^{-m(\lambda + \mu)\cdot t}] = \frac{2\nu}{m\lambda + 2\nu}$$

Математические ожидания времени пребывания в работоспособных состояниях:

$$T_i = \int_0^\infty dF_i(t), i = 0,1,...,5;$$

$$T_0 = \frac{1}{(m+2)\lambda}; \quad T_1 = \frac{1}{(m+1)\lambda + \mu}; \quad T_2 = \frac{1}{(m+1)\lambda + \nu};$$

$$T_3 = \frac{1}{m\lambda + \mu}; \quad T_4 = \frac{1}{m\lambda + \mu + \nu}; \quad T_5 = \frac{1}{m\lambda + 2\nu}.$$

Среднее время до первого отказа резервированного объекта равно

$$T_{\text{СР}} = \frac{\begin{array}{c} T_0\Delta G_1^0 + T_1[p_{01}\Delta G_2^0 + p_{02}(p_{21} + p_{25}p_{54}p_{43}p_{31})] + \\ + T_2[p_{02}\Delta G_3^0 + p_{01}p_{14}p_{42}] + T_3[p_{01}p_{13}\Delta G_2^0 + p_{01}p_{14}p_{43} + \\ + p_{02}(p_{24}p_{43} + p_{25}p_{54}p_{43})] + T_4[p_{01}p_{14} + p_{02}(p_{24} + p_{25}p_{54} + \\ + p_{21}p_{14})] + T_5[p_{02}p_{25}\Delta G_3^0 + p_{01}p_{14}p_{42}p_{25}] \end{array}}{\Delta G_{\overline{S_P}}} \quad (2.17),$$

где $\Delta G_1^0 = 1 - C_2 - C_3 - C_4 - C_7 - C_8 + C_7C_8 + C_4C_7$,

$\Delta G_2^0 = 1 - C_4 - C_8$, $\Delta G_3^0 = 1 - C_2 - C_7$,

$$\Delta G_{\overline{S_P}} = 1 - \sum_{i=1}^{8} C_i + C_6C_8 + C_6C_4 + C_7C_8 + C_4C_7$$

Таблица III.1

μ(1/ч)	**0,1**	**1**	**10**	**100**
p_{01} p_{02}	0,90 / 0,99 0,05 / 0,005	0,90 /0,99 0,05 / 0,005	0,90 / 0,99 0,05 / 0,005	0,90 /0,99 0,05 / 0,005
P_{10} P_{13} P_{14}	0,98522167 0,01330049 / 0,01463054 0,00024631 / 0,00002463	0,99850225 0,00134798 / 0,00148278 0,00002496 / 0,0000025	0,99985002 0,00013498 / 0,00014848 0,0000025 / 0,00000025	0,99998500 0,00001350 / 0,00001485 0,000000025 / 0
P_{21} P_{24} P_{25}	0,99850225 0,00089865 / 0,00098852 0,00002496 / 0,00000249	0,99850225 0,00089865 / 0,00098962 0,00002496 / 0,00000249	0,99850225 0,00089865 / 0,00098852 0,00002496 / 0,00000249	0,99850225 0,00089865 / 0,00098962 0,00002496 / 0,00000249
P_{31}	0,99009901	0,99900100	0,99990001	0,99999000

P_{42} P_{43}	0,09082652 0,90826521	0,49975012 0,49975012	0,90900827 0,09090083	0,99008921 0,00990089
P_{54}	0,99950025	0,99950025	0,99950025	0,99950025
T_0(ч)	500	500	500	500
T_1(ч)	9,866798	0,998652	0,999865	0,999987
T_2(ч)	0,998652	0,998652	0,998652	0,998652
T_3(ч)	9,866798	0,998652	0,099987	0,010000
T_4(ч)	0,907977	0,499663	0,090898	0,009901
T_5(ч)	0,499750	0,499750	0,499750	0,499750
C_1	0,04918732 / 0,00491873	0,04985035 / 0,00498503	0,04991762 / 0,00499176	0,04992436 / 0,004992436
C_2	0,00022150 / 0,00002215	0,00001246 / 0,00000125	0,00000023 / 0,00000002	0 / 0
C_3	0,000022234 / 0,0000022	0,00001246 / 0,00000125	0,00000227 / 0,00000027	0,00000002/0
C_4	0,00000223 / 0,00000022	0,00001247 / 0,00000125	0,00002268 / 0,00000227	0,0000247 / 0,00000247
C_5	0,00000111 / 0,00000011	0,00000061 / 0,00000006	0,00000011/ 0,00000001	0,00000001/0
C_6	0,88669948 / 0,97536942	0,89865203 / 0,98851723	0,89986502 / 0,98985152	0,89998650 / 0,98998515
C_7	0,01316880 / 0,01448568	0,00134663 / 0,00148130	0,00013497 / 0,00014846	0,0000135 / 0,00001485
C_8	0,00008162 / 0,00008978	0,0004491 / 0,00049401	0,00081688 / 0,00089957	0,00088974 / 0,00097872
C_6C_8	0,000073 / 0,00009111	0,00040410 / 0,00049802	0,00073501 / 0,00089044	0,00080101 / 0,00096892

C_6C_4	0,00000201 / 0,00000021	0,00001121 / 0,00000123	0,00002041 / 0,00000224	0,00002223 / 0,000002450
C_7C_8	0,00000107 / 0,0000013	0,00000060 / 0,00000073	0/0	0/0
C_4C_7	0,00000003/0	0,00000002/0	0/0	0/0
ΔG_1^0	0,98650468 / 0,98540124	0,99816750 / 0,99802558	0,99902297 / 0,99895197	0,99990720 / 0,99900396
ΔG_2^0	0,99991611 / 0,99990999	0,99953843 / 0,99953843	0,99916044 / 0,99909816	0,99908556 / 0,99901881
ΔG_3^0	0,98660970 / 0,98549217	0,99864091 / 0,99851745	0,99986480 / 0,99985152	0,99998650 / 0,99998515
$\Delta G_{\overline{S_p}}$	0,05069175 / 0,00522646	0,05000797 / 0,00501787	0,04999563 / 0,00499880	0,04998440 / 0,00499772
$T_{СР}$(ч)	**9939 / 96164**	**10238 / 102035**	**10240 / 102425**	**10243 / 102447**

Пример 2.7. Интенсивности отказов основных и резервных устройств объекта составляют $\lambda=5\cdot10^{-4}$1/ч. Интенсивность длительности скрытых отказов резервных устройств равна $\nu=1^1$/ч. Количество основных устройств – два, резервных – два. Требуется определить значения средней наработки до отказа объекта при различных интенсивностях восстановления устройств и при двух значениях вероятности правильного обнаружения отказов устройств $\alpha=0,9$ и $\alpha=0,99$.

Рассчитывают входные данные к формуле (2.17) – см. табл. III.1

Полученные в примере результаты характерны для многих резервированных цифровых объектов информационных систем. Прежде всего, в качестве исходных данных приняты типичные для цифровых устройств значения интенсивностей отказов,

восстановлений и эффективности контроля. Результаты решения модели приведены на графиках рис. III.2.10.

Эти результаты подтверждают следующее:

1. Возможность восстановления позволяет значительно (на порядок и более) повысить среднюю наработку до отказа резервированного объекта. Однако при этом не наблюдается линейной зависимости между ростом наработки до отказа и снижением длительности восстановления. Этот рост имеет место в пределах интервалов большой длительности восстановления и объясняется тем, что при этом в результате даже небольшого повышения интенсивности восстановления отказавших устройств снижается возможность возникновения в этих интервалах времени отказов других устройств объекта. По мере роста коэффициента загрузки обслуживания, начиная от уровня $\rho=\lambda/\mu\geq10^{-3}$, рост надежности практически приостанавливается и дальнейшее увеличение интенсивности восстановления (уменьшение

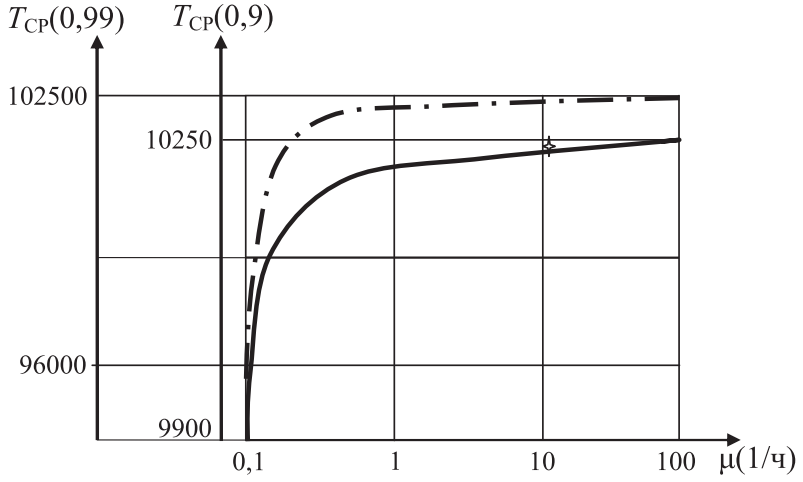


Рис. III.2.10. Графики зависимости средней наработки до отказа резервированного объекта ИС от интенсивности восстановления при двух уровнях эффективности обнаружения отказов (α=0,9 – сплошная линия в пределах значений от 9900 до 10250 часов; α=0,99 – штрих – пунктирная линия в пределах значений от 96000 до 102500 часов).

длительности восстановления) не имеет смысла (см. графики на рис. III.2.10).

2. Определяющее влияние на повышение надежности резервированных объектов с восстановлением оказывает эффективность обнаружения отказов устройств. Так, в рассматриваемом резервированном объекте повышение вероятности правильного обнаружения отказов от уровня α=0,9 до уровня α=0,99 позволяет на порядок увеличить среднюю наработку до отказа объекта, тогда как в резервированных объектах без восстановления этот способ повышения надежности приносит скромные результаты, не превышающие уровня (50-70)%. Следует обратить внимание на то обстоятельство, что в модели принята довольно жесткая гипотеза о возникновении отказа объекта в целом при пропуске отказа любого основного устройства. В действительности всегда есть некий допустимый интервал времени существования скрытого отказа, как это показано нами в п. III.2.3.1. С учетом этого обстоятельства следует ожидать еще большего эффекта от повышения вероятности правильного обнаружения отказов устройств объекта.

Комплексным показателем надежности резервированного объекта с восстановлением в установившемся режиме служит коэффициент готовности. Для того, чтобы получить представление о зависимости этого показателя от исходных параметров объекта информационной системы (интенсивностей восстановления и отказов составных компонент и, конечно, от вероятности правильного обнаружения) достаточно ограничиться простым с точки зрения техники анализа случаем однократного резервирования с ограниченным контролем и скрытыми отказами. Графовая модель такого объекта показана на рис. III.2.11.

На графе рис. III.2.11 вершинами определены следующие состояния:

1 – исправны m основных устройств и резервное устройство объекта;

2 – отказало и восстанавливается одно устройство. Отказ своевременно обнаружен. Если отказало основное устройство, то оно замещается резервным устройством;

3 – скрытый отказ одного резервного устройства, основные устройства исправны;

4 – скрытый отказ одного любого основного устройства – отказ объекта.

5 – отказали два любых устройства – отказ объекта. Восстанавливается одно устройство, второе отказавшее устройство в очереди на восстановление, m-1 устройства исправны.

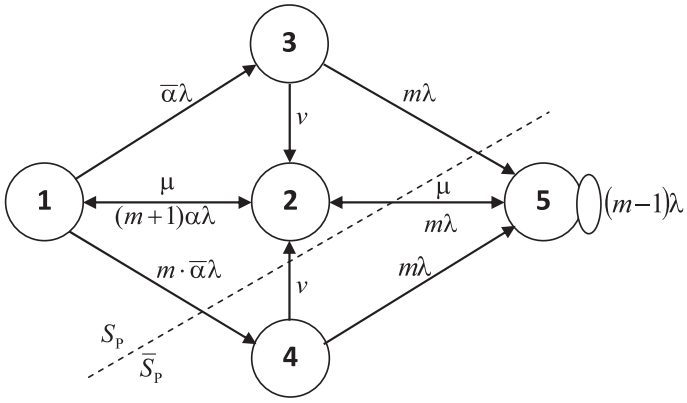


Рис. III.2.11. Граф надежности резервированного объекта с однократным резервом.

Так же как и в предыдущей модели, примем экспоненциальными распределения случайного времени безотказной работы устройства, времени его восстановления и длительности скрытых отказов с интенсивностями λ, μ, ν соответственно.

Отказ объекта наступает при возникновении одного из следующих событий:

– отказали k+1=2 устройства;

– отказало любое одно основное устройство и его отказ пропущен с вероятностью $\overline{\alpha}$. Объект переходит с интенсивностью

$m \cdot \overline{\alpha}\lambda$ из состояния 1 в состояние 4 множества неработоспособных состояний.

Все множество состояний объекта S={1, 2, 3, 4, 5} в соответствии с критерием отказа разделяется на множество работоспособных состояний $S_{\text{Р}}$={1, 2, 3} и множество неработоспособных состояний $\overline{S}_{\text{Р}} = \{4, 5\}$, где $S = S_{\text{Р}} \bigcup \overline{S_{\text{Р}}}$ (на графе рис. III.2.11 эти множества состояний разделены пунктирной линией).

Задача состоит в определении показателей готовности и безотказности, таких как коэффициент готовности $K_{\text{Г}}$ и средняя наработка до отказа $T_{\text{СР}}$ резервированного объекта, имея в виду, что начальным состоянием является состояние 1, в котором исправны все основные устройства и одно резервное устройство

Надежность рассматриваемого резервированного объекта моделируется с помощью Марковского случайного процесса. В работе [1] показано, что стационарные вероятности пребывания случайного полумарковского (в частном случае Марковского процесса) в различных состояних графа могут определяться по формуле

$$P_i = \frac{\Delta G^i T_i}{\sum\limits_{S} \Delta G^i T_i}, \qquad (2.18)$$

где ΔG^i – вес разложения графа без состояния i, T_i –математическое ожидание безусловного времени пребывания объекта в состоянии i (для Марковского процесса определение безусловного времени не требуется – достаточно установить математическое ожидание времени пребывания в каждом конкретном состоянии).

Коэффициенты готовности и неготовности объекта информационной системы рассчитывают по следующим формулам:

$$K_{\text{Г}} = \sum_{i \in S_P} P_i; \; K_{\text{НГ}} = K_{\text{ПР}} = 1 - K_{\text{Г}} = \sum_{i \in \overline{S}_P} P_i$$

Применительно к рассматриваемой модели надежности резервированного объекта с восстановлением (рис. III.2.11) исходные параметры имеют следующий вид:

$$p_{12} = (m+1)\alpha\lambda \int_0^\infty dF_1(t) = (m+1)\alpha\lambda \int_0^\infty d[1 - e^{-(m+1)\lambda \cdot t}] = \alpha;$$

$$p_{13} = \overline{\alpha}\lambda \int_0^\infty dF_1(t) = \frac{\overline{\alpha}}{m+1};\ p_{14} = \overline{\alpha}m\lambda \int_0^\infty dF_1(t) = \frac{\overline{\alpha}m}{m+1};$$

$$p_{21} = \mu \int_0^\infty dF_2(t) = \frac{\mu}{m\lambda + \mu};\ p_{25} = m\lambda \int_0^\infty dF_2(t) = \frac{m\lambda}{m\lambda + \mu};$$

$$p_{32} = \nu \int_0^\infty dF_3(t) = \frac{\nu}{m\lambda + \nu};\ p_{35} = m\lambda \int_0^\infty dF_3(t) = \frac{m\lambda}{m\lambda + \nu};$$

$$p_{42} = \nu \int_0^\infty dF_4(t) = \frac{\nu}{m\lambda + \nu};\ p_{45} = m\lambda \int_0^\infty dF_4(t) = \frac{m\lambda}{m\lambda + \nu};$$

$$p_{52} = \mu \int_0^\infty dF_5(t) = \frac{\mu}{(m-1)\lambda + \mu};$$

$$p_{55} = (m-1)\lambda \int_0^\infty dF_5(t) = \frac{(m-1)\lambda}{(m-1)\lambda + \mu}.$$

Математические ожидания времени пребывания в работоспособных состояниях:

$$T_i = \int_0^\infty dF_i(t), i = 1, 2, ..., 5;$$

$$T_1 = \frac{1}{(m+1)\lambda};\quad T_2 = \frac{1}{m\lambda + \mu};\quad T_3 = \frac{1}{m\lambda + \nu};$$

$$T_4 = \frac{1}{m\lambda + \nu}; \quad T_5 = \frac{1}{(m-1)\lambda + \mu}.$$

Найдем аналитические выражения весов разложения графа без состояний *i*:

$\Delta G^1 = 1 - p_{25}p_{52} - p_{55}$. Эта формула получена таким образом: на графе закрываем вершину 1 и определяем контуры в оставшейся части графа. В данном случае остается один контур 25-52 и петля 55. Аналогично определяются веса других разложений:

$$\Delta G^2 = 1 - p_{55}; \; \Delta G^3 = 1 - p_{14}p_{42}p_{21} - p_{14}p_{45}p_{52}p_{21} - p_{12}p_{21} - p_{25}p_{52} - p_{55};$$
$$\Delta G^4 = 1 - p_{13}p_{32}p_{21} - p_{13}p_{35}p_{52}p_{21} - p_{12}p_{21} - p_{25}p_{52} - p_{55};$$
$$\Delta G^5 = 1 - p_{13}p_{32}p_{21} - p_{14}p_{42}p_{21} - p_{12}p_{21}$$

Коэффициент готовности резервированного объекта имеет следующий вид:

$$K_\Gamma = \frac{\Delta G^1 T_1 + \Delta G^2 T_2 + \Delta G^3 T_3}{\sum_{i=1}^{5} \Delta G^i T_i} \qquad (2.19)$$

Средняя наработка до отказа объекта (определяется в множестве работоспособных состояний $S_P = \{1,2,3\}$)

$$T_{\text{СР}} = \frac{T_1 + T_2(p_{12} + p_{13}p_{32}) + T_3 p_{13}}{1 - p_{12}p_{21} - p_{13}p_{32}p_{21}} \qquad (2.20)$$

Пример 2.8. Интенсивности отказов основных и резервного устройства объекта составляют $\lambda = 5 \cdot 10^{-4}$ 1/ч. Интенсивности восстановления и длительности скрытых отказов резервных устройств соответственно $\mu = 1^1$/ч. $\nu = 1^1$/ч. Количество основных устройств – два, резервных – одно. Требуется определить значения коэффициента готовности и средней наработки до отказа

объекта при различных значениях вероятности правильного обнаружения отказа устройств

Результаты расчетов по формулам (2.19) и (2.20) приведены в табл. III.2 и показаны на рис. III.2.12 а и на III.2.12 б в виде графиков зависимостей показателей коэффициента готовности и средней наработки до отказа соответственно от вероятности правильного обнаружения отказа.

Табл. III.2

α	**0,80**	**0,90**	**0,95**	**0,99**
p_{12}	0,80	090	0,95	0,99
p_{13}	0,06666667	0,03333334	0,01666667	0,00333333
p_{14}	0,13333334	0,06666667	0,033333334	0,00666666
p_{21}	0,99900100	0,99900100	0,99900100	0,99900100
p_{25}	0,00099900	0,00099900	0,00099900	0,00099900
p_{32}	0,99900100	0,99900100	0,99900100	0,99900100
p_{35}	0,00099900	0,00099900	0,00099900	0,00099900
p_{42}	0,99900100	0,99900100	0,99900100	0,99900100
p_{45}	0,00099900	0,00099900	0,00099900	0,00099900
p_{52}	0,99950025	0,99950025	0,99950025	0,99950025
p_{55}	0,00049975	0,00049975	0,00049975	0,00049975
T_1	666,6667	666,6667	666,6667	666,6667
T_2	0,9990	0,9990	0,9990	0,9990
T_3	0,9990	0,9990	0,9990	0,9990
T_4	0,9990	0,9990	0,9990	0,9990
T_5	0,9995	0,9995	0,9995	0,9995
ΔG^1	0,99850175	0,99850175	0,99850175	0,99850175
ΔG^2	0,99950025	0,99950025	0,99950025	0,99950025
ΔG^3	0,06610088	0,03253407	0,01615074	0,00283076
ΔG^4	0,13271441	0,06581508	0,03281424	0,00617425
ΔG^5	0,00119859	0,00109880	0,00104844	0,00100898
$K_Г$	0,99979938	0,99989974	0,99994926	0,99998923
$K_{ПР}$	$2{,}01\cdot10^{-4}$	$1{,}00\cdot10^{-4}$	$5{,}10\cdot10^{-5}$	$1{,}10\cdot10^{-5}$
$T_{СР}$	4972,23	9871,50	14020,54	87029,55

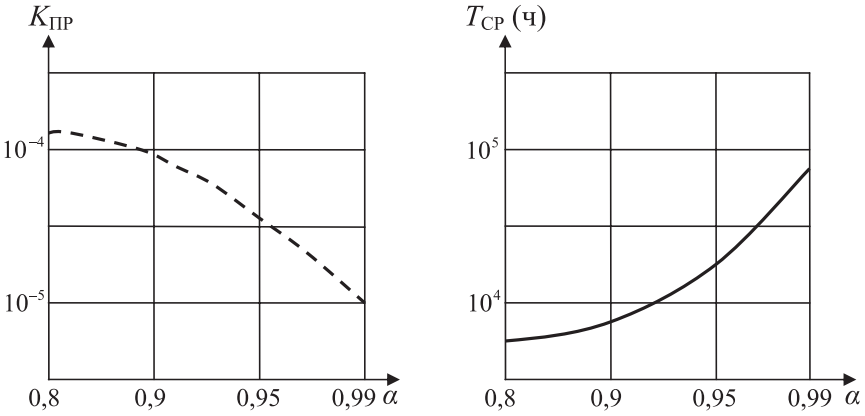


Рис. III.2.12. Зависимости коэффициента простоя (а) и средней наработки до отказа (б) резервированного объекта в примере 2.8

Полученные результаты свидетельствуют о следующем:

1. Определяющее влияние на готовность и безотказность резервированных объектов оказывает эффективность средств обнаружения отказов их компонентов. Так, при типичных для цифровых объектов значений параметров отказов и восстановлений составных компонент, повышение вероятности правильного обнаружения отказов от уровня α=0,8 до уровня α=0,95 позволяет более чем на порядок уменьшить коэффициент простоя и более чем на порядок повысить среднюю наработку до отказа объекта.

2. Наибольшая скорость роста надежности объекта в условиях данного примера, а также в условиях построения других подобных резервированных объектов, наблюдается при обеспечении вероятности правильного обнаружения отказов в диапазоне значений α=0,9–0,99. Однако, при этом следует учитывать негативный фактор снижения надежности за счет аппаратно – программных средств контроля, объем которых может составлять (15-20)% от объема резервированного объекта. Влияние этого негативного фактора может быть скомпенсировано за счет того,

что в работе резервированного объекта допустимо определенное время перерыва в работе. В результате этого часть скрытых отказов основных устройств не будет приводить к отказам объекта. Эти качественные соображения следует изучить с помощью математического моделирования.

III.2.6. Предельная надежность резервированных объектов

III.2.6.1. Предельная безотказность резервированных объектов

Рассмотренные ранее модели надежности невосстанавливаемых и восстанавливаемых резервированных объектов свидетельствуют о том, что параметр правильного обнаружения отказов компонентов оказывает существенное влияние на надежность объекта в целом. Однако остаются открытыми вопросы значимости других параметров надежности объекта, таких как параметры отказов и восстановлений, виды резервирования и, особенно, кратность резервирования. С этой целью построим следующую гипотетическую модель (рис. III.2.13).

Пусть объект информационной системы содержит m основных однотипных компонент. Предположим, что к каждому основному компоненту подключено бесконечное количество таких же однотипных резервных компонент. *В* ***этом заключается главная идея построения универсальной модели исследования предельной надежности резервированного объекта,*** *поскольку при бесконечном количестве резервных компонент совершенно безразлично возможно или невозможно восстановление отказавших компонент объекта, какое количество ремонтных бригад в системе, как организовано профилактическое обслуживание, как построен ЗИП и т. д., поскольку в объекте всегда существует достаточное количество исправных резер-*

вных компонент. Вместе с тем предусматриваются следующие реальные условия:

• контроль каждой компоненты не идеальный (осуществляется с вероятностью правильного обнаружения отказа α);

• существует ограничение по суммарному времени обнаружения отказа и переключения на резерв;

• переключение на другой компонент небезотказно. При этом под вероятностью отказа в подключении резервного компонента вместо отказавшего основного понимается вероятность того, что либо отказало переключающее устройство при условии правильного и своевременного обнаружения отказа (суммарное время обнаружения отказа компонента и переключения на резерв должно быть меньше допустимого), либо отказ не обнаружен при условии исправности переключающего устройства, либо одновременно не обнаружен отказ основного компонента и в состоянии отказа находится переключающее устройство. Эта результирующая вероятность отказа в переключении обозначается как γ;

• время выхода из строя всех резервированных компонентов статистически не зависит друг от друга;

• у всех резервированных компонентов одинаковое распределение срока службы;

• распределение срока службы компонентов непрерывно во времени, дифференцируемо, имеет ограниченное среднее число и стремится к 1, как время стремится к бесконечности.

Задача заключается в определении предельных верхних границ вероятности безотказной работы и средней наработки до отказа как отдельной резервированной группы, состоящей из одного основного и бесконечного количества резервных компонент, и объекта в целом, содержащего m таких резервированных групп. При этом для получения гарантированной верхней оценки принимается, что средства контроля, коммутации и уп-

равления реконфигурацией объекта в информационной системе идеально надежны

Оценим предельную вероятность безотказной работы резервированной группы, состоящей из одного основного и бесконечного количества резервных компонент, с помощью формулы полной вероятности:

$$P_{\text{П}}(t) \le \sum_{n=0}^{\infty} P(n,t) \cdot \gamma^{n} \qquad (2.21)$$

где $P(n,t)$ – распределение результирующего числа интервалов времени между заменами отказавшего устройства объекта, который до момента отказа выполнял функции основного модуля.

В формуле (2.21) просуммированы вероятности всех возможных событий для оговоренных выше условий, а именно:

– в течение времени t не отказало основное устройство (n=0, $P(0,t)\cdot\gamma^0 = P(0,t)$);

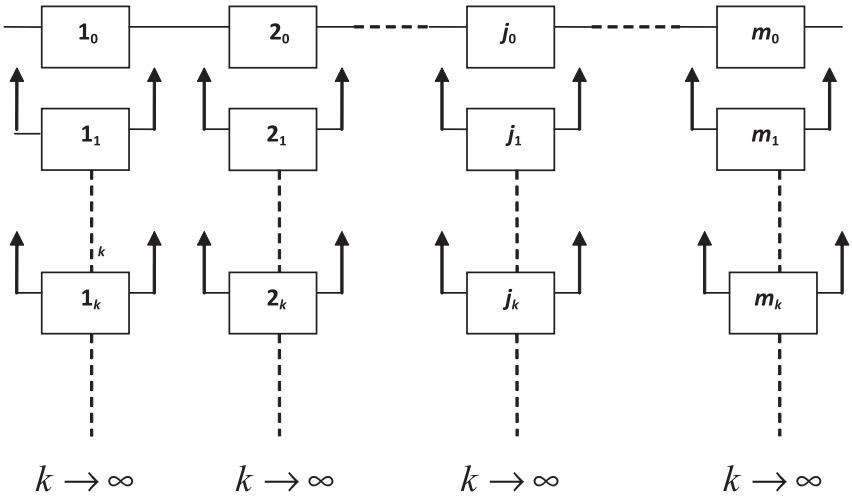


Рис. III.2.13. Блок-схема надежности резервированного объекта с раздельным резервированием в составе m основных и бесконечного количества резервных однотипных устройств

– в течение времени t отказало основное устройство группы. Отказ своевременно обнаружен и в течение допустимого времени успешно парирован своевременной заменой на исправный резервный с вероятностью γ (это событие фиксируется как n=1, $P(1,t)\cdot\gamma^1$);

– в течение времени t отказали ровно два компонента: основное устройство и затем другое устройство, которое было подключено из резерва вместо отказавшего основного. Оба отказа своевременно обнаружены и успешно устранены с вероятностью $\gamma\cdot\gamma=\gamma^2$ своевременной заменой на два любых исправных резервных устройства и т. д. до бесконечности, поскольку в течение времени t возможны отказы не только исходных, но и замененных основных устройств

Итак, для нахождения вероятности $P_{\text{П}}(t)$ достаточно раскрыть функцию распределения $P(n,t)$.

Рассмотрим следующий случайный процесс. Работа резервированной группы начинается с нового основного устройства в нулевой момент времени. Если основное устройство откажет в какой-то момент времени, то оно будет немедленно заменено новым однотипным устройством из резерва. При очередном отказе опять произойдет замена на очередное новое устройство из резерва и т. д. Интервалы времени между отказами устройств являются независимыми одинаково распределенными случайными величинами с одной и той же плотностью распределения. Система этих случайных величин представляет собой простой процесс восстановления.

Примем, что длительность времени между двумя соседними отказами устройств в резервированной группе имеет специальное распределение Эрланга a-го порядка (выбор данного распределения объясняется тем, что эрланговскими распределениями можно приблизить любые другие распределения).

В этих условиях распределение $P(n,t)$ результирующего числа интервалов времени между отказами устройств группы равно

распределению $P\{N_t=n\}$ вероятности того, что число N_t отказов основных устройств (как исходного, так и замененных) равно n, причем, последнее распределение может быть явно выражено для всех n [2]:

$$P(n,t) = P\{N_t = n\} = \sum_{i=na}^{na+a-1} \frac{(\lambda \cdot t)^i}{i!} e^{-\lambda \cdot t} \qquad (2.22)$$

где λ – интенсивность отказов одного устройства

Подставляя данное выражение в формулу (2.21) и учитывая, что

$\gamma^n = \gamma^{\frac{an+a-i}{a}} / \gamma^{\frac{a-i}{a}}$, где i=0,1,…,a–1, определяем правую часть выражения (2.21):

$$P_{\Pi}(t) \le e^{-\lambda \cdot t} \cdot \sum_{n=0}^{\infty} \sum_{i=1}^{a-1} \frac{1}{\sqrt[a]{\gamma^i}} \frac{(\lambda t \cdot \sqrt[a]{\gamma})^{an+i}}{(an+i)!}.$$

Выполним дальнейшие упрощения.

Пусть $\lambda t \cdot \sqrt[a]{\gamma} = \vartheta$.

Поскольку $\sum_{n=0}^{\infty} \frac{\vartheta^{an}}{(an)!} = \sum_{n=0}^{\infty} \frac{\vartheta^n}{n!} - \sum_{n=0}^{\infty} \sum_{i=1}^{a-1} \frac{\vartheta^{an+i}}{(an+i)!}$, то

$$P_{\Pi}(t) \le e^{-\lambda \cdot t(1-\sqrt[a]{\gamma})} + e^{-\lambda t} \cdot \sum_{n=0}^{\infty} \sum_{i=1}^{a-1} \frac{1-\sqrt[a]{\gamma^i}}{\sqrt[a]{\gamma^i}} \frac{(\lambda t \cdot \sqrt[a]{\gamma})^{an+i}}{(an+i)!}. \qquad (2.23)$$

Формула (2.23) представляет аналитическую оценку предельной вероятности безотказной работы резервированных объектов. Она имеет важное практическое значение, поскольку показывает, что ключевое влияние на уровень их безотказности оказывают характеристики средств обнаружения отказов составных устройств и своевременного переключения их на резерв (эти характеристики обобщены в формуле вероятностью γ).

Количество же резервных устройств, которое в приведенных рассуждениях принято сколь угодно большим, оказывают по отношению к вероятности γ второстепенное влияние. Проанализируем формулу (2.23). Если при неограниченном количестве резервных устройств можно было бы добиться сколь угодно близкой к 1 вероятности γ, то можно было полагать, что в этом случае безотказность резервированной группы будет сколь угодно высокой. С другой стороны, если, несмотря на неограниченное количество резервных устройств будет низкой данная вероятность ($\gamma \to 0$), то уровень безотказности резервированной группы будет на уровне безотказности одного составного модуля ($P_{\text{П}}(t) \to e^{-\lambda t}$). Это означает, что при отсутствии средств обнаружения отказов устройств и отсутствия возможности своевременного переключения их на резерв или же при низких характеристиках этих средств применение структурного резервирования практически не даёт эффекта.

Ранее было отмечено, что с помощью распределения Эрланга можно смоделировать практически любое распределение результирующего числа интервалов времени между отказами ВМ. Особый интерес представляет экспоненциальное распределение, которое характерно для электронной техники. Кроме того, с помощью экспоненциального распределения можно наглядно оценить влияние на надежность резервированной группы характеристик средств обнаружения отказов составных устройств, безотказность средств переключения и своевременность переключения на резерв. В этом случае параметр распределения Эрланга равен a=1, а выражение (2.23) преобразуется к виду

$$P_{\text{П}}(t) \le \exp(-\lambda t \overline{\gamma}) \qquad (2.24)$$

Формула (2.24) наглядно показывает ключевое влияние на уровень безотказности резервированных групп вероятности γ.

Она также свидетельствует о том, что в результате введения очень большого числа резервных устройств исходная интенсивность отказов резервированной группы снижается не более, чем на величину $\overline{\gamma} = 1 - \gamma$. Если значение вероятности γ невелико (например, $\gamma < 0{,}5 - 0{,}6$), то бессмысленно повышать уровень безотказности резервированной группы за счет наращивания резерва.

Рассмотрим теперь модель ИС, в которой система состоит из *m* резервированных групп. В каждой группе имеет место простой процесс отказов и восстановлений. Все они независимы. Отсюда следует возможность оценки предельной вероятности безотказной работы системы в виде:

$$P_{\text{СП}}(t) = \prod_{j=1}^{m} P_{\Pi j}(t) \le \exp(-\sum_{j=1}^{m} \lambda_j t(1-\gamma_j))$$

или при однотипных основных и резервных устройствах

$$P_{\text{СП}}(t) \le \exp[-m\lambda t(1-\gamma)].$$

Предельная максимальная наработка до отказа резервированного объекта при экспоненциальных распределениях времени между отказами составных устройств оценивается в следующем виде:

$$T_{\text{СП}} = \int_0^{\infty} P_{\text{СП}}(t)dt \le \frac{1}{m\lambda(1-\gamma)} \qquad (2.25)$$

Следует обратить внимание на то, что при бесконечном количестве резервных устройств не требуется ремонтировать отказавшее устройство – время его восстановления сводится только к его замене, после чего объект функционирует из исходного состояния. Поэтому можно принять, что максимальная предельная средняя наработка до отказа резервированного объекта равна предельной средней наработке между отказами

этого объекта, т.е. $T_{max}=T_{СП}=T_{ОП}$, где $T_{ОП}$ – среднее время между отказами объекта.

В п. III.2.3.1 приведен физический смысл вероятности успешного перехода на резерв и определена эта вероятность как $\gamma=\alpha\cdot\phi$. Для достижения вероятности α правильного обнаружения отказов цифрового устройства информационной системы на уровне 0,95–0,99 требуется объем средств контроля соизмеримый с объемом основного оборудования [1]. По этим причинам, а также по причинам технологического характера производства БИС и СБИС, средства их контроля не перенасыщены – находятся в разумных пределах по отношению к основным средствам устройства. Поэтому вероятность правильного обнаружения отказов в цифровых устройствах находится обычно в границах 0,7–0,99 (ниже в микропроцессорных системах, выше в высокопроизводительных ИС). Таким образом, даже при сколь угодно большом количестве резервных устройств в условиях ограниченных возможностей средств контроля (без учета реальных ограничений по допустимому времени переключения на резерв) следует ожидать увеличения наработки до отказа резервированных объектов в информационных системах до предельных значений, превышающих наработку до отказа исходного не резервированного объекта не более чем в 10–20 раз. Если же учесть реальную надежность средств коммутации, контроля переключения на резерв, то указанные граничные оценки будут существенно ниже.

В п. III.2.3.1 также отмечено, что если операции обнаружения и устранения отказа являются зависимыми, осуществляются комплексно с помощью средств адаптивной отказоустойчивости, то в формуле (2.25) следует учесть это обстоятельство и вероятность γ заменить на вероятность успешной адаптации к отказу β.

Рассмотренные ранее модели безотказности резервированных объектов с восстановлением развиты для кратнос-

ти *k/m* резервирования объектов, где количество резервных устройств находится в диапазоне k=2, 3, …, 10, а основных устройств рассмотрены в диапазоне от 2 до 10. При этом отношение интенсивности отказов к интенсивности восстановления устройства принято равным $\rho=\lambda/\mu=0{,}001$, что типично для цифровой техники.

Полученные результаты исследования отношения средней наработки до отказа резервированного объекта к средней наработке до отказа объекта без резервирования, содержащего m устройств ($T_{СР}/T(m)$) в зависимости от параметров α, k, m показаны на рис. III.2.14. На этом рисунке сплошными линиями показаны зависимости указанного отношения при кратности резервирования $1/m$, штриховыми линиями – зависимости этого отношения при кратности резервирования k/m, а штрихпунктирной линией – зависимость предельной средней наработки до отказа объекта от вероятности α. При низкой эффективности контроля отказов устройств ($\alpha<0{,}8$) сплошные и штриховые линии для соответствующих значений m совпадают.

Данные результаты позволяют сделать следующие выводы:

• При реальных значениях вероятности правильного обнаружения отказов устройств объекта $\alpha<0{,}95$ введение сколь угодно большого количества резервных устройств не позволяет повысить среднюю наработку до отказа исходного нерезервированного объекта более чем в 5…20 раз (см. штрихпунктирную линию на рис. III.2.14). Этот результат справедлив при любом количестве основных устройств и при различных соотношениях их интенсивностей отказов и восстановлений. Он свидетельствует о том, что реальные возможности структурного резервирования значительно скромнее тех, которые в теории надежности оценивались в предположении идеальной эффективности средств обнаружения отказов. Кроме того, за

счет повышения кратности резервирования, как это предполагалось ранее, нельзя добиться сколь угодно высокого уровня надежности технической системы.

• За счет структурного резервирования приемлемой кратности (k<10)можно ожидать повышения средней наработки до отказа резервированного объекта с восстановлением примерно в 2…10 раз. Так, например, при кратности резервирования 1/8, 2/8, 3/8 и при типичном для цифровой техники отношении интенсивностей отказов и восстановлений устройств данный показатель возрастает не более чем в 2 раза (см. рис. III.2.14 при α<0,99). По мере уменьшения количества основных устройств эффективность структурного резервирования увеличивается. Например, при кратностях 1/4, 2/4, и 3/4 средняя наработка объекта до отказа увеличивается уже не в два, а в 6 раз, а при 1/2, 2/2 и 3/2 – до 10 раз. Надо отметить, что приведенные оценки завышены, пос-

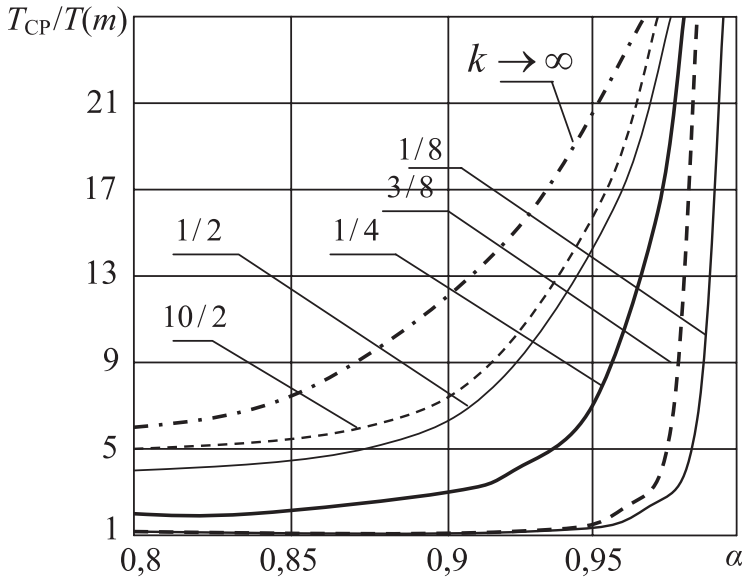


Рис. III.2.14. Зависимости нормированных значений средней наработки до отказа резервированных объектов с восстановлением кратности k/m (при k=1 –сплошные линии) от вероятности правильного обнаружения отказов составных устройств

кольку в расчетах предполагалась идеальная надежность средств контроля, коммутации и переключения на резерв;

• При достижении реальных значений эффективности обнаружения отказов можно ожидать ощутимый выигрыш в надежности в результате повышения кратности резервирования от $1/m$ до $2/m$. Однако и в этом диапазоне значений вероятности расчетные уровни средней наработки до отказа практически совпадают при кратности резервирования объекта 1/2 и 2/2 и даже 3/2 и 4/2;

• При ограниченных возможностях средств обнаружения отказов повышение кратности резервирования объекта выше двух нецелесообразно. Об этом свидетельствуют следующие результаты : при значениях вероятности $\alpha \leq 0{,}85$ используемый показатель безотказности резервированных объектов кратности 1/2, 2/2 , 3/2 и 4/2 находится примерно на одном уровне и только при кратности 10/2 он возрастает примерно на 10%.

III.2.6.2. Предельная готовность резервированных объектов

Оценку предельной максимальной готовности резервированных систем можно сделать, руководствуясь следующими соображениями:

• для цифровых устройств и объектов характерно функционирование в установившемся режиме. Переходный режим по оценкам проф. А.Д.Соловьева ограничен длительностью времени соизмеримой с суммой нескольких интервалов времени восстановления, что для цифровой техники соответствует нескольким часам непрерывной работы, тогда как установившийся режим оценивается годами также непрерывной работы. Поэтому в качестве показателей готовности резервированного объекта удобно использовать коэффициенты его готовности (неготовности, простоя);

• исходными данными для расчетов предельного коэффициента готовности естественно принять предельное максимальное значение средней наработки на отказ $T_{\text{ОП}}$ и предельного минимального среднего времени простоя объекта $\tau_{\text{ПРЕД}}$, которые связаны между собой соотношением

$$K_{\text{ПРЕД}} = \frac{T_{\text{ОП}}}{T_{\text{ОП}} + \tau_{\text{ПРЕД}}}.$$

• В приведенном выражении предельное максимальное значение средней наработки на отказ объекта определяется по формуле (2.25). В свою очередь, предельное минимальное значение времени простоя имеет место при нулевых затратах времени на операции доставки ремонтников, ЗИПа, диагностического оборудования, наконец, при нулевых затратах времени на обнаружение отказов составных устройств объекта. Следовательно, предельное минимальное значение времени простоя равно среднему времени восстановления объекта, т.е. $\tau_{\text{ПРЕД}} = T_{\text{ВО}}$. В свою очередь, среднее время восстановления объекта при однотипных основных и резервных устройствах равно среднему времени восстановления одного устройства. Действительно, среднее время восстановления резервированного объекта кратности *k/m* определяется следующим выражением

$$T_{\text{ВО}} = \frac{\sum_{i=1}^{k+m} \lambda_i T_{\text{В}i}}{\sum_{i=1}^{k+m} \lambda_i} \qquad (2.26)$$

При однотипности основных и резервных устройств времена восстановления всех устройств в среднем одинаковы ($T_{\text{В1}} = T_{\text{В2}} = \ldots = T_{\text{В}}$ Таким образом, с учетом формулы (2.26) минимальное предельное время простоя резервированного объекта равно

времени восстановления одного устройства, $\tau_{ПРЕД} = T_В$, а максимальный коэффициент готовности определяется в виде

$$K_{Гпред} = K_{Г\max} \le \frac{T_{ОП}}{T_{ОП} + \tau_{ПРЕД}} = \frac{\frac{T}{1-\gamma}}{\frac{T}{1-\gamma} + T_В} = \frac{1}{1 + m\rho\bar{\gamma}}$$

где $T = \frac{1}{m\lambda}$ и $\rho = \lambda/\mu$.

В свою очередь, минимальное значение коэффициента простоя выражается через коэффициент готовности объекта, т.е.

$$K_{П\min} \ge 1 - K_{Г\max} \ge \frac{m\rho\bar{\gamma}}{1 + m\rho\bar{\gamma}}$$

На рис. III.2.15 показаны графики зависимости минимального коэффициента простоя объекта от вероятности его успешной адаптации к отказам составных элементов. Из этих графиков следует:

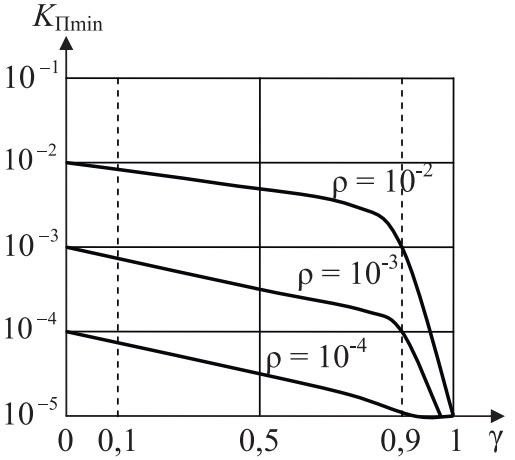


Рис. III.2.15. Зависимости минимального коэффициента простоя объекта с бесконечным количеством составных резервных элементов от их надежности и эффективности обнаружения отказов и своевременного перехода на резерв.

– существенное снижение коэффициента простоя объекта, а следовательно, существенное повышение его коэффициента готовности достигается при обеспечении вероятности успешной адаптации объекта к отказам на уровне γ>0,85–0,90;

– чем ниже надежность составных элементов, тем больше зависимость готовности объекта от эффективности средств обнаружения отказов и переключения на резерв.

III.2.7. Информационное резервирование

III.2.7.1. Технологии хранения информации

Системы хранения информации в информационных системах включают аппаратную и программную составляющие. Возможности построения накопителей данных (аппаратной составляющей системы хранения информации) весьма обширны. На мировом рынке сегодня представлены самые разные типы накопителей данных: RAID-массивы, оптические носители, роботизированные DVD-библиотеки и др.

RAID-массивы (Redundant Array of Independent Disks) представляют собой объединенный массив дисковых накопителей, управляемый интеллектуальной подсистемой. Программное обеспечение, используемое для управления массивами, интегрируется в общую информационную систему предприятия. Задачи, решение которых обеспечивает программное обеспечение RAID-массивов, это, прежде всего, устойчивость работы системы хранения, к примеру, выбор альтернативного пути передачи данных и принятие решений в случае отказа сервера или мэйнфрейм-системы.

Ленточные накопители привлекательны, прежде всего, невысокой ценой и при этом обладают значительной емкостью для хранения данных (2-8 ГБ и более). Кроме того, автономные ленточные накопители могут объединяться в своеобразные

“массивы”, называемые ленточными библиотеками, суммарная емкость которых оптимизирована для хранения значительных объемов корпоративной информации. Роботизированные ленточные библиотеки смогут со временем обладать практически теми же возможностями, что и дисковые массивы. Они уже сейчас отличаются сравнительно высокой надежностью, высоким быстродействием и возможностью расширения количества реализуемых функций.

Одной из наиболее перспективных технологий хранения данных является запись на оптические носители – DVD и CD диски. Роботизированная библиотека представляет собой массив DVD или CD-R дисков, размещенных в отдельном корпусе, суммарная емкость библиотеки измеряется десятками терабайт. Помимо дисковых массивов, в корпусе библиотеки расположены приводы, обеспечивающие запись и считывание информации. Число приводов может быть различным в зависимости от конкретной модели. Библиотеки имеют возможность подключения дополнительных дисков, хранящихся вне корпуса. Подключение дополнительных дисков осуществляется посредством mail-слота или специальных магазинов на несколько десятков дисков.

Роботизированная библиотека управляется специальным программным обеспечением, которое может быть представлено как базовым ПО для небольших массивов данных, так и программным обеспечением для иерархического управления хранилищами (Hierarchical Storage Management, HSM). Основными функциями программного обеспечения роботизированных библиотек являются: управление аппаратными средствами, в частности, для выполнения операций чтения и записи, организация хэширования данных, а также организация и управление виртуальной файловой системой библиотеки. Управляющее ПО позволяет представить библиотеку в составе системы в виде одного логического диска огромной емкости.

В рамках концепции иерархического управления хранилищами осуществляется перенос информации с RAID-массивов в DVD-библиотеки, программному обеспечению задаются определенные рамки, устанавливаемые администратором системы, где корпоративным данным присваивается определенный приоритет, определяется актуальность и частота используемой информации. При этом центр тяжести обеспечения надежности системы хранения информации приходится на дисковые массивы, в которых в реальном масштабе времени производится динамичное накопление данных.

III.2.7.2. Общие положения обеспечения надежности дисковых накопителей данных

Для достижения повышенного уровня надежности дисковых накопителей данных приходится жертвовать пропускной способностью ввода/вывода или емкостью памяти. Необходимо использовать дополнительные диски, содержащие избыточную информацию, которая позволяет восстановить исходные данные при отказе основного диска. Отсюда получают акроним для избыточных матриц недорогих дисков RAID (redundant array of inexpensive disks). Существует несколько способов объединения дисков RAID. Каждый уровень представляет свой компромисс между пропускной способностью ввода/вывода, емкостью диска, предназначенной для хранения избыточной информации, и надежностью. Когда какой-либо диск отказывает, предполагается, что в течение короткого интервала времени он будет заменен и информация будет восстановлена на новом диске с использованием избыточной информации. Это время до восстановления можно уменьшить, если в систему ввести дополнительные диски в качестве нагруженного резерва – при отказе основного диска резервный диск подключается аппаратно-программными средствами. Периодически оператор вручную заменяет все от-

казавшие диски. Четыре основных этапа этого процесса состоят в следующем:

• определение отказавшего диска,

• устранение отказа без останова обработки;

• восстановление потерянных данных на резервном диске;

• периодическая замена отказавших дисков на новые.

Для обеспечения надежности матриц дисков, обозначаемых как RAID – массивы, применяются следующие основные способы:

• дублирование дисков (создание так называемых зеркальных дисков) – RAID 1;

• поразрядное расслоение матрицы дисков – RAID 2;

• формирование типовых логических дисков с контролем по четности – RAID 3;

• расслоение матрицы дисков на уровне секторов – RAID 4;

• распараллеливание записей данных и соответствующих им блоков четности – RAID 5;

• распараллеливание записей данных и соответствующих им блоков двумерной четности – RAID 6.

Рассмотрим кратко каждый из перечисленных способов.

Зеркальные диски (RAID 1)

Зеркальные диски представляют традиционный способ повышения надежности магнитных дисков. Это наиболее дорогостоящий из рассматриваемых способов, так как все диски дублируются и каждый раз информация записывается как на основной, так и на резервный диск. Таким образом, приходится идти на некоторые жертвы в пропускной способности ввода/вывода и емкости памяти ради получения более высокой надежности. Зеркальные диски широко применяются многими фирмами. Некоторые компании кроме применения зеркальных дисков также дублируют контроллеры и магистрали ввода/вывода. Эта версия зеркальных дисков поддерживает параллельное считывание.

Дублирование всех дисков может означать удвоение стоимости всей системы или, иначе, использование лишь 50% емкости диска для хранения данных. Повышение емкости, на которое приходится идти, составляет 100%. Такая низкая экономичность привела к появлению следующего уровня RAID-массивов.

Поразрядное расслоение матрицы дисковых накопителей (RAID 2)

Один из путей достижения надежности при снижении потерь емкости памяти может быть подсказан организацией основной памяти, в которой для исправления одиночных и обнаружения двойных ошибок используются избыточные контрольные разряды. Это решение можно реализовать путем поразрядного расслоения данных и записи их на диски блоками основных данных. На контрольных дисках записываются контрольные данные, обеспечивающие обнаружение двойных и исправление одиночных ошибок с помощью кода Хемминга. Такая организация позволяет сформировать только один поток ввода/вывода для каждой группы независимо от ее размера. При этом группы большого размера приводят к снижению избыточной емкости, идущей на обеспечение отказоустойчивости, тогда как при организации групп меньшего размера наблюдается снижение количества операций ввода/вывода, которые могут выполняться матрицей параллельно.

При записи больших массивов данных системы с архитектурой надежности RAID 2 имеют такую же производительность, что и системы RAID 1, хотя в них используется меньше контрольных дисков и, таким образом, по этому показателю они превосходят системы RAID 1. При передаче небольших порций данных производительность теряется, так как требуется записать либо сосчитать группу данных целиком, независимо от конкретных потребностей. Таким образом, системы RAID 2

предпочтительны для суперкомпьютеров, но не подходят для обработки транзакций. Компания Thinking Machine использовала RAID 2 в ЭВМ Connection Machine при 32 дисках данных и 10 контрольных дисках, включая 3 диска нагруженного резерва.

Формирование типовых логических дисков с контролем по четности (RAID 3)

Контрольные диски, используемые в RAID 2, нужны для определения положения неисправного разряда. Эти диски становятся полностью избыточными, так как большинство контроллеров в состоянии определить, когда диск отказал при помощи специальных сигналов, поддерживаемых дисковым интерфейсом, либо при помощи дополнительного кодирования информации, записанной на диск и используемой для исправления одиночных случайных сбойных ошибок. Уменьшение числа контрольных дисков до одного на группу снижает избыточность емкости до вполне разумных размеров. Часто количество физических дисков в логическом диске равно 5 (4 диска данных плюс 1 контрольный). Подобные устройства выпускаются, например, фирмами Maxtor и Micropolis. Каждое из таких устройств воспринимается машиной как отдельный логический диск с учетверенной пропускной способностью, учетверенной емкостью и средствами обнаружения двойных и устранения одиночных ошибок в пределах данных этого диска.

Расслоение матрицы дисковых накопителей на уровне секторов (RAID 4)

Достоинство поразрядного расслоения состоит в простоте вычисления кода Хэмминга, что необходимо для обнаружения и исправления ошибок в системах с архитектурой надежности RAID 2. В RAID 3 обнаружение ошибок данных на диске с точностью до сектора осуществляется дисковым контроллером.

Следовательно, если записывать отдельный блок передачи в отдельный сектор, то можно обнаружить ошибки отдельного считывания без доступа к дополнительным дискам. Главное отличие между архитектурами надежных систем RAID 3 и RAID 4 состоит в том, что в последних расслоение выполняется на уровне сектора данных, а не на уровне битов или байтов. В архитектуре RAID 4 обновление контрольной информации может быть реализовано достаточно просто. Для вычисления нового значения четности требуются лишь старый блок данных, старый блок четности и новый блок данных. Архитектура надежной системы RAID 4 позволяет также повысить производительность передачи небольших объемов данных за счет параллелизма, давая возможность выполнять более одного обращения по вводу/выводу к группе в единицу времени. Логические блоки передачи в данном случае не распределяются между отдельными дисками. Вместо этого каждый индивидуальный блок попадает на отдельный диск.

В архитектуре RAID 4 для записи небольших массивов данных предусматривается использование двух дисков, которые выполняют четыре выборки (чтение данных плюс четности, запись данных плюс четности). Производительность групповых операций записи и считывания остается прежней, но при небольших (на один диск) записях и считываниях производительность существенно улучшается. К сожалению, улучшение производительности оказывается недостаточным для того, чтобы этот метод построения архитектуры матрицы дисковых накопителей мог занять место архитектуры RAID 1.

Распараллеливание записей данных и соответствующих им блоков четности (RAID 5)

В архитектуре RAID 4 можно добиться параллелизма при считывании отдельных дисков, но запись по-прежнему ограни-

чена возможностью выполнения одной операции на группу, так как при каждой операции должны выполняться запись и чтение контрольного диска. Архитектура RAID 5 улучшает возможности архитектуры RAID 4 посредством распределения контрольной информации между всеми дисками группы.

Это небольшое изменение оказывает огромное влияние на производительность записи небольших массивов информации. Если операции записи могут быть спланированы так, чтобы обращаться за данными и соответствующими им блоками четности к разным дискам, появляется возможность параллельного выполнения $N/2$ записей, где N – число дисков в группе. Данная организация имеет одинаково высокую производительность при записи и при считывании как небольших, так и больших объемов информации, что делает ее наиболее привлекательной в случаях смешанных применений.

Распараллеливание записей данных и соответствующих им блоков двумерной четности(RAID 6)

В архитектуре RAID 5 предлагают, по существу, лишь одно измерение дисковой матрицы, вторым измерением которой являются секторы. Теперь рассмотрим объединение дисков в двумерный массив таким образом, чтобы секторы являлись третьим измерением. Мы можем иметь контроль четности по строкам, как в системах уровня 5, а также по столбцам, которые, в свою очередь, могут расслаиваться для обеспечения возможности параллельной записи. При такой организации можно преодолеть любые отказы двух дисков и многие отказы трех дисков. Однако при выполнении логической записи реально происходит шесть обращений к диску: за старыми данными, за четностью по строкам и по столбцам, а также для записи новых данных и новых значений четности. Для некоторых применений с очень высокими требованиями к отказоустойчивости такая избыточ-

ность может оказаться приемлемой, однако для традиционных информационных систем и для обработки транзакций данный метод не подойдет.

В общем случае, если доминируют короткие записи и считывания и стоимость емкости памяти не является определяющей, наилучшую производительность демонстрируют системы с архитектурой надежности RAID 1. Однако, если стоимость емкости памяти существенна, либо если можно снизить вероятность появления коротких записей (например, при высоком коэффициенте отношения числа считываний к числу записей, или при эффективной буферизации последовательностей считывания-модификации-записи, либо при приведении коротких записей к длинным с использованием стратегии кэширования файлов), архитектура рассматриваемых RAID 6 матриц дисковых накопителей данных позволит обеспечить также очень высокую производительность, особенно в терминах отношения стоимость/производительность.

III.2.8. Вопросы для самоконтроля

1. Приведите и поясните классификацию видов структурного резервирования.

2. Сформулируйте модель общего постоянного резервирования с идеальным контролем.

3. Сформулируйте модель общего однократного постоянного резервирования с неидеальным контролем.

4. Как изменяется преимущество раздельного резервирования перед общим резервированием в зависимости от характеристик встроенных средств контроля составных элементов объекта?

5. Какие типы резервирования замещением Вам известны?

6. Раскройте проблему скрытых отказов в структурном резервировании замещением.

7. Чем объясняется снижение структурной надежности объекта при мажоритарном резервировании?

8. Какое влияние оказывает функциональная надежность восстанавливающего органа на общую надежность объекта с мажоритарным резервированием?

9. Объясните причины, почему резервирование с восстановлением имеет значительные преимущества перед резервированием без восстановления объекта.

10. Приведите модель предельной оценки надежности объекта со структурным резервированием.

11. Какой предельный рост безотказности объекта следует ожидать за счет его структурного резервирования при реальной эффективности средств обнаружения отказов?

12. Перечислите способы обеспечения надежности дисковых накопителей данных.

13. Раскройте суть способов информационного резервирования объекта путем дублирования дисков и поразрядного расслоения матрицы дисков.

14. Раскройте суть способов информационного резервирования объекта путем формирование типовых логических дисков с контролем по четности и расслоения матрицы дисков на уровне секторов.

15. Раскройте суть способов информационного резервирования объекта путем распараллеливания записей данных и соответствующих им блоков одномерной и двумерной четности.

ГЛАВА III.3. АДАПТИВНАЯ ОТКАЗОУСТОЙЧИВОСТЬ В ИНФОРМАЦИОННЫХ СИСТЕМАХ РЕАЛЬНОГО ВРЕМЕНИ

III.3.1. Требования к отказоустойчивости информационных систем

Для обеспечения надежного решения задач в условиях отказов системы применяются два принципиально различающихся подхода – восстановление решения после отказа системы (или ее компонента) и предотвращение отказа системы (отказоустойчивость).

Отказоустойчивость информационной системы должна отвечать следующим основным требованиям:

– система должна быть построена так, чтобы в ней ***отсутствовал какой-либо (даже самый незначительный) единственный компонент (ресурс),*** выход из строя которого приводит к полному отказу всей системы. Такая система не может содержать какого-либо «слабого звена», размещенного в одном из процессорных элементов системы;

– ***отказоустойчивые информационные системы реального времени*** должны быть способны не только гарантировать решение поставленной задачи, но при этом ***должны соблюдать временные ограничения***. Отказоустойчивость систем реального времени является важным параметром при их проектировании. Если пренебречь отказоустойчивостью, можно попасть в ситуацию, при которой система перестает функционировать, при этом задачи, требующие решения в данный момент времени, простаивают или вовсе остаются нерешенными;

– в системе должны присутствовать ***функции наблюдаемости и управляемости*** с тем, чтобы обеспечить ***реакцию на любое событие, даже ошибку***. Таким образом, любой сбой приведет к заранее определенному поведению, что должно обеспечить работу системы фактически в любых условиях;

– ***защита от перерасхода ресурсов*** в системе – эта защита предназначена для систем, в которых пользовательские программы пишутся без контроля над использованием ими системных ресурсов или содержат "размножающиеся" потоки исполнения. Надежность системы вследствие исполнения таких программ может существенно снизиться;

– ***оперативное восстановление ресурсов*** – после завершения работы прикладной программы (независимо по какой причине), выделенные ей ресурсы должны быть возвращены в систему, если только они не объявлены как резервные. При отсутствии этой возможности система с частыми запусками/остановками приложений быстро приходит к состоянию неэффективной работы в связи с недостатком требуемых ресурсов и необходимости её перезагрузки.

– ***изоляция приложений*** для того, чтобы ни одна из программ не смогла разрушить ни другие приложения, ни базовую операционную систему.

III.3.2. Традиционные способы обеспечения отказоустойчивости

Известны следующие способы обеспечения отказоустойчивости:

– *резервирование ресурсов*, например, вычислительных модулей (ВМ), под которыми в распределенных модульных информационных системах подразумеваются как отдельные процессорные элементы с элементами памяти, так и многопроцессорные

кластеры. Обычно применяется дублирование ресурсов с их восстановлением в случае отказов [51, 52 и др.];

– *защита от перерасхода ресурсов*; ограничение уровней приоритета пользовательских задач, контроль переполнения пользовательского стека во избежание разрушения содержимого памяти других задач и т.д. Для защиты от перерасхода ресурсов применяется выдача предупреждающих сообщений, а также, при необходимости, выдача сигнала об исчерпании лимита пользования ресурсом, при нарушении которого запускаются соответствующие действия по восстановлению;

– *кластеризация*. С точки зрения оборудования кластер – это несколько компьютеров, соединенных коммуникационными каналами и разделяющих общие ресурсы (например, дисковые накопители). Кластер имеет общую файловую систему, а для пользователя он выглядит и управляется как единая система. Это достигается с помощью специального программного обеспечения, регулирующего скоординированное использование общекластерных ресурсов и осуществляющего взаимный контроль работоспособности и обмен специфической “кластерной” информацией между узлами кластера и т. д. При этом различные простые аппаратные способы повышения надежности (RAID, структурное резервирование, аппаратный контроль и др.) также используются. Отдельные компьютеры называются узлами кластера. Отличительной особенностью кластера является то, что каждый его узел выполняет полезную работу в режиме нормального функционирования и может переключить на себя нагрузку отказавшего узла. Кластерные решения разрабатываются многими известными фирмами, такими, как HP, Digital, IBM и др. Все известные кластерные решения в той или иной степени обеспечивают высокую готовность приложений и возможность наращивания производительности за счет закупки нового оборудования или замены старого на более мощное.

– *контроль состояния объектов, ответственных за поддержание уровней сервиса в процессе модернизации*. Например, серверная задача, извлекающая запросы из очереди сообщений, может быть заменена без какого-либо уведомления клиентских программ, поскольку очередь продолжает существовать независимо от замены задачи и может накапливать запросы, пока обновленный сервер не начнет работать;

– *средства переназначения прав владения* в составе операционной системы, для того чтобы постоянно существующие объекты не возвращались в систему во время восстановления ресурсов;

– *для изоляции приложений создаются блоки управления памятью* (диспетчеры памяти), для чего в операционной системе формируется набор пользовательских интерфейсов для назначения прав чтения/записи, либо, что встречается более часто, механизм создания областей памяти с ограниченным доступом. При попытке обращения в запрещенную область исполнение программы прекращается и запускается механизм восстановления после сбоя. И что важнее всего, обработка исключительных ситуаций может быть реализована в отдельном адресном пространстве приложения и никак не влиять на остальные части системы;

– *обеспечение отказоустойчивости предполагает парирование действия отказов и маскирование сбоев,* т.е. предотвращение распространения последствий сбоя на продолжение выполнения системой своих функций. Парирование действия отказов всегда связано с введением в систему того или иного вида избыточности;

– *фиксирование отказа или сбоя системы в целом или в ее отдельных частях с последующей реконфигурацией системы*. Такой способ связан с прерыванием функционирования системы, т.е. не обеспечивает отказоустойчивость в системах реального времени;

– *маскирование сбоев по способам мажоритарного резервирования* (см.п. III.2.6). Недостатком этих способов является большое количество избыточного оборудования и снижение производительности системы;

– *микроядерная архитектура операционной системы (ОС),* например QNX Neutrino [41] с полной изоляцией модулей в ОЗУ. Микроядро обеспечивает связь элементов системы между собой, но при этом не реализует большинство системных сервисов. Это позволяет разработчику самому решить, какие сервисы нужны для решения его прикладной задачи, позволяет самому написать необходимые системные приложения и подключить их в систему, создавая по возможности ОС весьма небольшого размера и узкой специализации. Микроядро обеспечивает: мгновенное вытеснение задачи с меньшим приоритетом; защиту от инверсии приоритетов на базе протокола наследования приоритетов; исключение непредвиденных расходов ресурсов; механизм обеспечения «горячей» замены сервисов; механизм формирования распределенной вычислительной среды; механизм поддержки резервирования физических каналов связи в кластере; технологию автоматического восстановления процессов; технологию автоматизации восстановления логических соединений и т.д.;

– *архитектура ОС, построенная по технологии "Protection Domain"* (операционная система VxWorks AE [42, 44 и др.]), позволяет изолировать ядро ОС от приложений и приложения друг от друга. Protection Domain (домены защиты) – принцип структурирования ядра ОС, позволяющий разработчику изолировать процессы и ресурсы при помощи специального контейнера ресурсов, в котором задаются параметры среды исполнения. В каждом домене защиты может исполняться любое количество задач. Контроль переполнения стека каждой задачи и возможность автоматически увеличивать размеры пула динамической

памяти (в определенных пределах) в случае переполнения представляет собой одну из мер защиты от перерасхода ресурсов. Кроме того, такая система позволяет ограничивать диапазон задачных приоритетов внутри доменов защиты. Таким образом, предотвращается "неуправляемое размножение" приложений и в любом случае устраняется их возможное влияние на исполнение других прикладных задач. В операционной системе VxWorks AE домен защиты определяет также и права владения системными ресурсами (задачи, очереди сообщений, семафоры, страницы памяти и т.д.). Домен защиты, которому выделены или в котором созданы эти ресурсы, становится для них "базовым" доменом (home domain). Независимо от способа завершения работы прикладной программы внутри домена защиты, все выделенные этому домену ресурсы восстанавливаются системой для их повторного использования;

– *архитектура системы высокой готовности* (архитектура Foundation HA, разработанная в компании Wind River [43, 45 и др.]), обладает способностью отличать базовые причины возникающих в информационной системе сбойных ситуаций (в том числе отказов аппаратуры) и признаки, появляющиеся как свидетельства этих причин. Единичный отказ (типа проявившейся ошибки какого-либо программного компонента или отказа аппаратного компонента) может стать источником множества признаков. В архитектуре Foundation HA имеется стандартный API-интерфейс оповещения, посредством которого усиленные программные компоненты передают признаки на уровень оповещения Foundation HA.

Подводя итоги, отметим, что у всех рассмотренных способов обеспечения отказоустойчивости информационных систем есть сходная черта – резервирование.

Процесс восстановления ресурсов в определенной степени является ручным процессом, выполняемым для каждого объек-

та отдельно (например, создание задачи, уничтожение задачи, создание очереди, уничтожение очереди). Общим недостатком рассмотренных способов представляется в большинстве своем автономная реализация механизмов обеспечения отказоустойчивости. Такой же недостаток имеет место в отношении средств обнаружения отказов и сбоев составных элементов и системы в целом. Обращает на себя внимание то, что все без исключения рассмотренные способы обеспечения отказоустойчивости информационных систем реального времени основываются на применении больших объемов искусственной структурной и информационной избыточности, т.е. практически базируются на экстенсивном пути обеспечения отказоустойчивости. Обеспечение отказоустойчивости в информационных системах реального времени является сложной задачей, основной путь решения которой – это построение адаптивных отказоустойчивых систем.

III.3.3. Концептуальные положения адаптивной отказоустойчивости информационных систем реального времени

III.3.3.1. Исходные предпосылки

Любую информационную систему реального времени можно представить в виде системы массового обслуживания с ограничением по времени пребывания заявок в очереди. Этот интервал времени равен циклу обработки информации. При проектировании информационной системы длительность цикла обработки информации выбирают из расчета обслуживания максимального количества возможных заявок. В каждом последующем цикле информация обновляется, но объем ее обработки, так же как и в предыдущем цикле, рассчитан

на обслуживание максимального количества заявок. Однако число поступающих заявок в текущем цикле обработки – случайная величина, находящаяся в диапазоне от нуля до предусмотренного максимального количества. Концентрация поступающих заявок в пределах цикла обработки зависит от закона распределения потока заявок. При любом законе распределения заявок время цикла обработки используется не полностью, часто на уровне 60% [46]. Следовательно, *в каждом цикле обработки информации имеет место* ***естественная*** *временная избыточность. Эта избыточность в настоящее время не используется.*

Любую информационную систему реального времени можно представить как многоканальную систему массового обслуживания, в которой в каждом цикле обработки используются не все имеющиеся *m* каналов обработки информации – все зависит от интенсивности поступления заявок. Здесь мы опять – таки *выделяем* ***естественный*** *структурный резерв.*

Информационные системы реального времени в большинстве своем это модульные системы, содержащие от нескольких ВМ (бортовые системы) до сотен и даже тысяч модулей (распределенные стационарные системы). Вычислительные модули могут представлять собой процессорные элементы в сочетании с элементами памяти или даже многопроцессорные кластеры в критически важных информационных системах. С помощью операционных систем реального времени осуществляется программное управление взаимодействием ВМ, информационный обмен, как между ними, так и с другими компонентами информационной системы. Все это означает *наличие в системе среды, позволяющей оперативно осуществлять реконфигурацию составных компонент (в первую очередь, ВМ) программным способом в целях реализации задач* ***наблюдаемости*** *и* ***управляемости*** *системы (см. п. III.1.5).*

III.3.3.2. Исходные положения адаптивной отказоустойчивости в информационных системах

Адаптивная отказоустойчивость возможна в информационных системах путем введения в их состав системы обеспечения отказоустойчивости (СОО). Эта новая составная система формируется при проектировании ИС из предусмотренных избыточных аппаратных и программных средств. Она предназначена для своевременной защиты или предотвращения неисправностей основной аппаратуры и программного обеспечения ИС.

Высокий уровень организации СОО возможен с помощью механизма адаптации. Рассмотрим вариант создания адаптивных структур СОО. Эта система содержит (рис. III.3.1):

– **преобразователь информации** (ПИ), который выполняет две группы задач: *первая группа* – это связь измеряемых состояний *X* системы, не измеряемых состояний *E* и адаптирующего воздействия *U*. Под измеряемыми состояниями понимаются данные о текущих состояниях основных аппаратных и программных средств и ресурса. Не измеряемые состояния – это

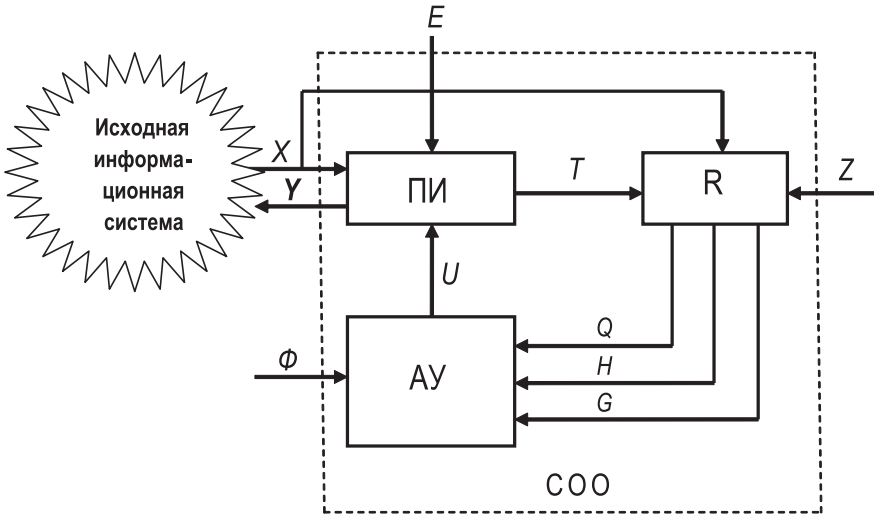


Рис. III.3.1. Система обеспечения адаптивной отказоустойчивости в информационной системе

потоки отказов, сбоев, программных ошибок. *Вторая группа* выполняемых ПИ задач – это формирование вектора *T* времен адаптации к отказам, сбоям, программным ошибкам и команд управления *Y* на текущее изменение ресурса, на реконфигурацию информационной системы, на корректировку текущих состояний системы;

– **аппаратный и программный ресурс *R*** системы. Он включает в себя как естественный, так и искусственный ресурс.

– **оператор адаптивного управления АУ,** предназначенный для формирования адаптирующего воздействия в соответствие с определенным алгоритмом *Ф*. Задача адаптации состоит в нахождении такого управляющего воздействия *U*, чтобы состояния вектора *T* системы СОО в среде измеряемых состояний *X* и в среде не измеряемых состояний *E* удовлетворяли поставленным целям *Z*

$$Z:\begin{cases} q_j(X,T) \to \text{extr}, j=1,...,k_1; \\ h_i(X,T) \le \tau_{\text{Д}}\ ; i=1,...,k_2; \\ g_i(X,T) \ge \beta\ \ , i=1,...,k_2 \end{cases},$$

где параметры целевой функции q_i для каждого *j*-го управления связаны с достижением при адаптации некоторых экстремальных значений, например, минимальных потерь производительности ИС за счет действий СОО, минимальных дополнительных расходов памяти, минимальных задержек в решении задач наблюдаемости и управляемости, минимального структурного резерва, максимального уровня показателей эффективности ИС в среде *X* и т.д. Эти минимизируемые (максимизируемые) параметры системы выступают как цель адаптации. Органу адаптивного управления кроме цели адаптации необходимо сообщить еще сведения о ресурсе R, в рамках которого возможна адаптация ($U \subset R$), и алгоритме Ф адаптации,

обеспечивающем синтез адаптирующего воздействия по имеющейся информации:

$$U = \Phi(X, T, Z, R).$$

Алгоритм Ф решает задачу оптимизации. Для этого задачу достижения целей Z сводят к известной задаче многокритериальной оптимизации

$$q_j(X,T) \to \text{extr}_{U \subset S}, j = 1, \ldots, k_1,$$

где множество S определяется условием $U \subset R$ и ограничениями h_i, g_i.

Ограничение h_i – это по существу требование к случайному времени адаптации $v_i = t_i - x_i$, заключающееся в том, что от момента возникновения неисправности x_i до момента t_i завершения защиты от ее воздействия на систему должно быть затрачено не более допустимого времени $\tau_Д$ перерыва в работе. Ограничение g_i состоит в выполнении первого требования в части ограничения h_i при условии выполнения заданного уровня гарантии – заданной вероятности успешной адаптации СОО к неисправности системы.

Для решения данной задачи многокритериальной оптимизации необходимо выявить зависимость целевой функции q_i от управления U путем прямого вычисления значений ограничений h_i, g_i и целевой функции. На рис. III.3.1 показана схема адаптации СОО, где Q, H, G – векторы с компонентами q_i и h_i, g_i. Для ее реализации требуется дополнительная информация об объекте. Процесс поиска этой информации может осуществляться эволюционным путем в ходе решения задач наблюдаемости и управляемости.

III.3.3.3. Идеи организации адаптивной отказоустойчивости информационных систем

Основная идея адаптивной отказоустойчивости состоит в активном использовании естественной временной и структурной избыточности и в активном переназначении имеющихся

вычислительных ресурсов не только для оперативной обработки информации, но и для реализации наблюдаемости системы при ограниченных средствах контроля. В дальнейшем понятие «адаптивная отказоустойчивость информационных систем реального времени» обозначается как *«активная защита (АЗ)»*

Активная защита предназначена для достижения требуемых уровней отказоустойчивости информационных систем реального времени в условиях незначительного резерва времени, ограниченной эффективности средств обнаружения неисправностей составных вычислительных модулей, а также при условии, что объем резервного оборудования не должен превышать объема основного оборудования. Она также предназначена для решения проблемы обеспечения заданного уровня вероятности успешной адаптации ИС к сбоям и отказам составных элементов и программ без существенного увеличения объема предусмотренных средств контроля и диагностики.

Активная защита основывается на следующих идеях [47]:

1. Длительности всех циклов обработки информации **разделяются на** определенные интервалы времени, которые в дальнейшем будем называть **тактами АЗ** или просто тактами. Такты вводятся с целью дискретизации непрерывного времени обработки информации. Моменты дискретизации предназначены для регистрации имеющихся в ИС в эти моменты исправных ВМ, а также для привязки к тактам АЗ операций наблюдаемости и управляемости ИС. *Каждый такт АЗ завершается формированием контрольной точки*, в которой сохраняются результаты работы ВМ в течение предыдущего такта. Операции наблюдаемости и управляемости, в частности механизмы контрольной точки, рестарта и др., строго согласуются с указанными моментами дискретизации. Так, контрольная точка, сформированная для вычислительного процесса какого-либо ВМ, обновляется через такты АЗ в моменты времени t_i, t_{i+1}, t_{i+2} и т.д.; рестарт осу-

ществляется на глубину, не превышающую времени $\tau_д$; сравнение результатов параллельной работы двух ВМ производится либо через один, либо через два и т. д., но не более, чем через m тактов АЗ.

Такты АЗ могут быть как **постоянной длительности** (например, в ИС с конвейерной обработкой информации), так и **случайной длительности**, что характерно для параллельных ИС с различной архитектурой. Как будет показано ниже, средняя длительность такта соответствует времени выполнения нескольких десятков или сотен программных модулей.

2. Все множество составных вычислительных модулей информационной системы подразделяется на два составных множества: **вычислительная среда** – множество, состоящее из $l \le m$ однотипных ВМ; **защитная среда** – множество из $k \le m$ однотипных ВМ. Если $l + k = m$, (где m – максимальное число основных ВМ), то считается, что в ИС отсутствует искусственная избыточность (отсутствуют дополнительные ВМ). Однако при таком соотношении сохраняется возможность использовать в интересах отказоустойчивости ИС $k < m$ естественно избыточных ВМ. Если $l + k > m$, то в системе есть $l + k – m$ дополнительных ВМ, а также $m – l$ естественно избыточных ВМ. В каждый момент дискретизации t_i для решения задач возможна потребность в использовании $l_i \le m$ основных ВМ, а также наличие в ИС $m–l_i+k_i$ избыточных ВМ. Если же работоспособно в этот момент времени всего l_i ВМ, то защитная среда себя исчерпала, но имеющихся основных ВМ достаточно для решения задач обработки информации. Таким образом, регистрация исправных элементов (ВМ) основной и защитной сред в каждый момент дискретизации необходима для определения принципиальной возможности продолжать в течение такта АЗ решать задачи обработки информации с достаточным количеством работоспособных ВМ. Она также необходима для рационального

обеспечения отказоустойчивости ИС в течение очередного такта с помощью избыточных работоспособных ВМ, если на момент времени t_i они имеются.

3. **Динамическая реконфигурация ИС производится через такты АЗ** для организации потактной параллельной работы требуемого количества основных ВМ и имеющихся исправных избыточных модулей. Это позволяет осуществить внешний контроль работоспособности ВМ. Так, если в момент времени t_i выполняется условие, что $m–l_i+k_i \geq l_i$, то возможно формирование l_i пар ВМ и, следовательно, возможен контроль по сбоям и отказам всех основных ВМ. Если же в этот момент времени исправен только один избыточный ВМ, то в предусмотренном порядке он подключается к очередному основному ВМ и в течение последующего такта АЗ параллельно работает одна пара ВМ, а остальные l_i+1 ВМ не контролируются в течение одного такта АЗ. Затем в следующем такте с помощью этого же исправного избыточного ВМ формируется другая пара ВМ и т. д. В результате за число тактов равное или меньше l_i обнаруживается факт неисправности в любом ВМ из l_i+1 оставшихся в конфигурации системы.

4. **Виртуальное резервирование всех *l* основных ВМ при наличии хотя бы одного исправного избыточного ВМ**. Достигается за счет того, что за очень короткое время, не превышающее *l* тактов, каждый из основных ВМ параллельно работает с однотипным ВМ. Следовательно, через эти же короткие отрезки времени работают пары модулей, в которых участвуют все основные ВМ. При наличии *k* исправных избыточных ВМ имеет место *k*-кратное виртуальное резервирование всех основных ВМ, поскольку каждый основной ВМ коммутируется с любым избыточным модулем.

5. Все этапы наблюдаемости ИС (обнаружение факта неисправности, локализация и классификация ее) **выполняются на**

реальных задачах без применения в процессе обработки информации диагностирующих средств. По этой причине для оценки отказоустойчивости ИС не имеет значения то, что при параллельной работе пары ВМ не обнаружен неисправный элемент модуля, который не использовался при решении данной задачи. Важно отметить, что чем выше интенсивность заявок, поступающих на обслуживание в ИС (т. е., чем более загружена система), тем чаще наблюдается ИС с помощью АЗ. И наоборот, чем менее загружена система, т. е. чем больше пауз между решаемыми задачами, тем реже наблюдается ИС с помощью АЗ. В больших паузах между задачами имеет смысл применять традиционные средства контроля и диагностики.

6. Для классификации неисправностей и обнаружения их местонахождения до уровня ВМ в системе должно быть **не менее *m = 2* основных и одного избыточного ВМ**. При простейшей организации активной защиты дополнительный ВМ работает параллельно с первым основным и в следующем такте (или через такт) – со вторым основным ВМ. В случае несовпадения результатов работы пары ВМ в предыдущем такте осуществляется повторный счет, что позволяет устранить ошибку или установить факт отказа одного из пары ВМ в случае повторного несовпадения результатов. Определение отказавшего ВМ производится в текущем такте по результатам работы дополнительного ВМ со вторым основным. В случае совпадения результатов принимается решение об отказе первого основного ВМ, в случае несовпадения результатов – об отказе дополнительного ВМ;

7**. Возможности активной защиты** в значительной мере зависят от выбора **средней длительности такта τ активной защиты.** Значение τ следует выбирать таким, чтобы в течение допустимого времени обнаружения отказа (сбоя, ошибки), определенном по длительности цикла обработки информации $\tau^{\bullet}_{\text{д}} = \tau_{\text{д}} - t_{\text{у}}$ отказавший ВМ с заданным уровнем гарантии дол-

жен быть локализован и перекоммутирован с дополнительным исправным ВМ. Время $t_у$ необходимо для восстановления вычислительного процесса с последней контрольной точки при соответствующем способе реализации активной защиты. При определении значения т необходимо учитывать времена решения задач и пауз между ними, законы распределения этих времен и времени такта активной защиты, количество A тактов в цикле контроля и количество b тактов принятия решения об отказе ВМ (эти числа зависят от количества m основных ВМ, способа реализации активной защиты, от количества к резервных ВМ).

III.3.4. Способы организации активной защиты

Активная защита допускает множество вариантов организации, которые можно классифицировать по трем основным признакам:

– по способам назначения пар ВМ;

– по способам формирования множеств основных и избыточных ВМ;

– по уровню активной защиты.

III.3.4.1.Способы назначения пар ВМ

Разработаны три группы способов формирования пар ВМ:

• с фиксированными контролирующими модулями, когда избыточные ВМ последовательно подключаются в качестве контролирующих к основным ВМ;

• с переназначаемыми модулями, когда предусматривается автоматическое перераспределение функций контроля между основными и избыточными ВМ;

• с приоритетным контролем ВМ, когда выделенные по каким-либо критериям приоритетности модули чаще других ра-

ботают в составе пар ВМ. В качестве критерия приоритетности можно рассматривать важность решаемой модулем задачи или пониженную его надежность, или увеличение интенсивности заявок на выполнение задачи модуля, или какую-либо совокупность этих частных критериев.

Рассмотрим содержание перечисленных вариантов назначения пар ВМ.

При **фиксации контролирующих модулей** имеют место два типичных случая. В первом случае, все m основных ВМ используются для решения предусмотренных задач. Предположим, что $m = 4$, а в защитной среде есть один избыточный ВМ ($k = 1$). Присвоим основным модулям номера 1, 2, 3, 4, а избыточному – номер 5. Тогда в первом такте АЗ образуется пара 5–1, во втором такте – пара 5–2, затем 5–3 и 5–4. Цикл активной защиты завершается через четыре такта. На пятом такте вновь образуется пара 5–1 и т. д. Следовательно, каждый основной ВМ в данном примере контролируется через цикл АЗ, равный четырем тактам. Пример такой организации контроля приведен в таблице III.3.1. Длительность контроля равна длительности одного такта. Если учесть, что длительность такта АЗ составляет несколько десятков миллисекунд, а среднее быстродействие ВМ в микропроцессорных ИС исчисляется десятками млн. операций в секунду, то в течение такта АЗ реализуется больше 100 операций [48], что свидетельствует о достаточной представительности полученных в течение такта наблюдений о работоспособности контролируемого ВМ.

В том случае, если для решения задач используется $l < m$ модулей и опять-таки применяются фиксированные контролирующие ВМ, возможности АЗ существенно повышаются. Убедимся в этом, сохранив прежние исходные условия относительно $m = 4$ и $k = 1$. Предположим теперь, что $l = 3$. Тогда в вычислительной среде содержатся ВМ с номерами, например,

1, 2 и 3, а в защитной среде – модули с номерами 4 и 5. В первом такте АЗ образуются пары 4–1 и 5–2, а модуль 3 остается без контроля. Во втором такте – пара 4–3, а свободный избыточный модуль 5 подключается для параллельной работы с основным модулем 1 (пара 5–1), который в предыдущем такте работал в паре с модулем 4.

Таблица III.3.1

Номер такта	**Номера основных ВМ**			**Номер контролирующего ВМ**	**Пары контролируемых ВМ**
1	1 2	3	3 4	5	5–1
2	1 2	3	3 4	5	5–2
3	1 2	3	3 4	5	5–3
4	1 2	3	3 4	5	5–4

Цикл АЗ завершен за два такта, однако при этом основной модуль 1 и оба избыточных ВМ контролировались в обоих тактах. Через три такта АЗ все основные ВМ контролируются в двух тактах (см. табл. III.3.2). В последующих трех тактах АЗ образуются те же пары ВМ, которые имели место в соответствующих первых трех тактах. Следовательно, при наличии двух и более модулей в защитной среде формируются два и более цикла АЗ, каждый из которых может обеспечивать более высокий уровень защиты.

Переназначение ВМ необходимо для сокращения цикла АЗ в условиях, когда количество основных ВМ существенно больше числа избыточных модулей. Суть переназначения заключается в том, что в определенных тактах АЗ перераспределяются ВМ между вычислительной и защитной средами. За определенными модулями защитной среды на время такта АЗ закрепляются функции основных модулей и наоборот. В результате этого устраняется недостаток, присущий способам фикса-

ции контролирующих ВМ, когда модули вычислительной среды контролируются значительно реже, чем модуль защитной среды. Действительно, во всех случаях фиксации модули защитной среды в пределах цикла АЗ участвуют во всех парах контролируемых ВМ (см. табл. III.3.1 и III.3.2). Тогда как модули вычислительной среды – только в одной паре из четырех возможных (табл. III.3.1), либо несколько чаще, если в каждом такте АЗ образуются две и более пары ВМ (табл. III.3.2).

Таблица III.3.2

Номер такта	**Номера циклов**		**Номера основных ВМ**	**Номера контролирующих ВМ**	**Пары контролируемых ВМ**
	1	**2**			
1	1 2 3	1	1 2 3	4 5	4–1 5–2
2			1 2 3	4 5	4–3 5–1
3			1 2 3	4 5	4–2 5–3
4		2	1 2 3	4 5	4–1 5–2
5			1 2 3	4 5	4–3 5–1
6			1 2 3	4 5	4–2 5–3

При переназначении все модули равномерно участвуют в совместной попарной работе и за счет этого сокращается длительность цикла АЗ. Проиллюстрируем это положение на следующем примере. Пусть $l = m = 4$, $k=l$. Основные ВМ нумеруются с 1 до 4, исходный избыточный ВМ – 5. Предположим, что в первом такте переназначения модулей не было и модуль 5 контролировал основной модуль 1 (пара 5–1). Во втором такте уже реализовано переназначение модулей. При этом модуль 5 выполняет функции основного модуля 2, который теперь контролирует основной модуль 3 (пара 2–3 в табл. III.3.3). В результате такой операции удается уже за два такта АЗ проконтролировать работу четырех из пяти ВМ (1, 2, 3 и 5), тогда как при

фиксации контролирующих ВМ в условиях этого же примера (см. табл. III.3.1) возможно проверить только три из пяти ВМ (1, 2 и 5). При переназначении ВМ за пять тактов АЗ все модули контролируются два раза, тогда как при фиксации ВМ возможно четыре основных модуля проверить по одному разу за четыре такта.

Эффективность переназначения ВМ возрастает но мере увеличения количества m исходных основных модулей. Так, при $m = 6$ и $k = 2$ удается проконтролировать все 8 модулей за два такта АЗ (см. табл. III.3.4), тогда как при фиксации модулей за эти же два такта возможно проконтролировать работу только $m = 3$ и $k = 2$ модулей (табл. III.3.2).

Таблица III.3.3

Номер такта	**Номера основных ВМ**	**Номер контролирующего ВМ**	**Пары контролируемых ВМ**	**Переназначаемые ВМ**
1	1 2 3 4	5	5–1	-
2	1 5 3 4	2	2–3	5 2
3	1 2 3 5	4	4–5	5 4
4	5 2 3 4	1	1–2	5 1
5	1 2 5 4	3	3–4	5 3

Система из 8 переназначаемых ВМ организована следующим образом (табл. III.3.4). Цикл однократной проверки модулей содержит $A = 2$ такта. В первом такте модуль 7 защитной среды выполняет функции основного модуля 2, который, в свою очередь, контролирует основной модуль 1 (пара 2–1). Кроме того, в этом же такте модуль 8 защитной среды контролирует основной модуль 5 (пара 8–5). Во втором такте модуль 8 выполняет функции основного модуля 4, при этом образуются пары 4–3 и 7–6. Возможны и другие варианты организации системы из 8 переназначаемых ВМ, однако данный вариант имеет важное

преимущество – переназначаются только модули 7→2 и 8→4, что существенно упрощает необходимые средства коммутации и управления в системе.

В общем случае при наличии m исправных основных ВМ и k избыточных можно определить количество тактов в цикле АЗ при условии их однократной проверки, исходя из рассмотренной логики назначения пар ВМ:

- при фиксации контролирующих модулей

$$A = \text{INT}\,[m/k], \tag{3.1}$$

Таблица III.3.4

Номер такта	Номера циклов		Номера основных ВМ	Номера контролирующих ВМ	Пары контролируемых ВМ	Переназначаемые ВМ
	1	2				
1			1 7 3 4 5 6	2 8	2–1 8–5	7 2
2	1		1 2 3 8 5 6	4 7	4–3 7–6	8 4
3	2	1	1 7 3 4 5 6	2 8	2–1 8–5	7 2
4			12 3 8 5 6	4 7	4–3 7–6	8 4

- при переназначении модулей

$$A = \text{INT}[(m + k)/2k]. \tag{3.2}$$

В формулах (3.1) и (3.2) имеется в виду, что операция INT – есть операция округления результата до ближайшего сверху целого числа. Если, например, $m = 5$, $k = 2$, то при фиксированных контролирующих ВМ A=INT[2,5] = 3, а при переназначении ВМ A=INT[1,75] = 2. Такие же значения A будут иметь место и при $m = 6$, $k = 2$. Поэтому при организации АЗ целесообразно при известном значении k избыточных ВМ охватывать защитой такое количество m основных модулей, чтобы выполнялись соотношения.

$$\text{INT}\,(m/k) = m/k \text{ и } \text{INT}[(m + k)/2k] = (m + k)/2k.$$

Приоритетный контроль организуется на основе переназначения ВМ. Однако при этом преследуется другая цель. Если при переназначении модулей решалась задача уравнять частоту контролей основных и избыточных ВМ, то при приоритетном контроле решается задача увеличения частоты контролей отмеченных модулей.

Проиллюстрируем возможности построения систем с одним и двумя отмеченными в качестве приоритетных модулями. В первом случае предполагается, что все ВМ, кроме отмеченного, контролируются с одинаковой частотой, а отмеченный ВМ с большей частотой.

Во втором случае предполагается, что первый отмеченный модуль (нулевой приоритет) контролируется в цикле АЗ с заданной наибольшей частотой, второй (первый приоритет) – с повышенной частотой, но меньшей, чем модуль нулевого приоритета. Остальные ВМ в этой же системе контролируются с одинаковой, но пониженной относительно приоритетных модулей частотой

Таблица III.3.5

Номер такта	**Номер цикла**	**Номера основных ВМ**	**Номер контролирующего ВМ**	**Пары контролируемых ВМ**	**Переназначаемые ВМ**
1		5 2 3 4	1	1–2	5–1
2	1	1 5 3 4	2	2–3	5–2
3		12 3 5	4	4–5	5–4
1		5 2 3 4	1	1–2	5–1
2	2	15 3 4	2	2–3	5–2
3		12 3 5	4	4–5	5–4

Предположим, что в системе содержатся $m = 4$ основных (номера с 1 по 4) и один (номер 5) избыточный ВМ. В качестве

приоритетного отмечен модуль 2 (табл. III.3.5) Требуется обеспечить вдвое большей частоту контролей модуля 2 по сравнению с другими четырьмя ВМ. Решение этой задачи приведено в табл. III.4.5. За три такта реализуется цикл АЗ, при этом модуль 2 контролируется дважды в течение цикла, а модуль 1, 3, 4, 5 – по разу.

Рассмотрим случай организации системы с двумя отмеченными ВМ в качестве приоритетных. Пусть $m = 7$, $k = 1$, нулевой приоритет отводится, как и в предыдущем случае, модулю 2, а первый приоритет – модулю 5.

Поставим условие, чтобы в цикле АЗ модуль 2 контролировался в четырех тактах, модуль 5 – в двух тактах, а остальные модули 1, 3, 4, 6, 7 и 8 – в одном такте. Решение этой задачи приведено в табл. III.3.6. Получены следующие результаты. Цикл АЗ равен $A = 6$ тактам, четыре раза переназначаются ВМ, модуль 2 контролируется в двух тактах из трех соседних, а модуль 5 контролируется через два такта.

Таблица III.3.6.

Номер такта	**Номера основных ВМ**	**Номер контролирующего ВМ**	**Пары контролируемых ВМ**	**Переназначаемые ВМ**	**Частота контролей ВМ**		
					2	**5**	**1, 3, 4, 6, 7, 8**
1	1 8 3 4 5 6 7	2	2–7	8–2	**4**	**2**	**1**
2	1 8 3 4 5 6 7	2	2–3	5–2			
3	1 2 3 4 5 6 7	8	8–5	–			
4	8 2 3 4 5 6 7	1	1–2	8–1			
5	1 2 3 8 5 6 7	4	4–2	8–4			
6	1 2 3 4 8 6 7	5	5–6	8–5			

Длительность цикла АЗ возросла по сравнению с равномерным переназначением ВМ в 1,5 раза, поскольку в том вариан-

те длительность цикла равнялась бы $A = (m + k)/2k = 4$. Это естественно, так как сокращение интервалов времени между контролями одних ВМ возможно за счет увеличения времени между контролями неприоритетных ВМ. В решении таких задач активной защиты должен быть применен разумный компромисс. Это в полной мере относится и к выбору способа фиксации или переназначения ВМ. В первом случае проще управление АЗ, во втором случае короче цикл управления. При очень малых величинах допустимого времени перерыва в работе предпочтительнее переназначение ВМ, хотя несколько сложнее управление АЗ. При менее жестких временах ограничениях следует стремиться к реализации АЗ на основе фиксации контролирующих ВМ.

III.3.4.2.Способы формирования множеств основных и избыточных ВМ

Системы с АЗ можно классифицировать следующим образом:

1) **с общей АЗ**, когда не применяется дифференцированный подход и к элементам защитной среды, и к особенностям архитектуры информационной системы. При организации общей АЗ формируется единый блок управления взаимодействием вычислительной и защитной сред и единое коммутационное поле. Основное достоинство общей АЗ состоит в наличии только одного блока управления. Однако при этом имеют место следующие недостатки:

– при больших количествах основных и избыточных ВМ резко возрастает сложность блока управления и, особенно, коммутационного поля. Вследствие этого снижается и производительность и отказоустойчивость системы, недопустимо возрастает количество элементов коммутации;

– существенно сокращаются возможности АЗ в адаптации к решаемым группами основных ВМ функциональным задачам и

к установленным для каждой задачи временным ограничениям по времени перерыва в работе соответствующих вычислительных структур;

– из-за сложности управления затруднено применение более совершенных способов АЗ (в частности, приоритетного контроля или переназначения модулей). Таким образом, применение общей АЗ целесообразно в отдельных подсистемах ИС, например, в интерфейсных каналах, содержащих небольшое количество основных и избыточных ВМ и выполняющих однотипные задачи обмена или передачи данных;

2) **с раздельной АЗ**, когда каждое из двух множеств ВМ разделяется на l подмножеств таким образом, чтобы соответствующие множества R_i вычислительной и S_i защитной среды (i = 1, 2, ..., r) были взаимосвязаны. При этом должны выполняться следующие условия: $S_i \geq r$, $\sum_{i=1}^{r} S_i = k$ и $\sum_{i=1}^{r} R_i = m$. Раздельная АЗ формируется на уровне каждой пары подмножеств R_i и S_i с отдельным управляющим органом и коммутационным полем. Таким образом, в ИС образуется r автономных подсистем с АЗ, которые управляются общей операционной системой ИС.

В параллельных ИС каждое подмножество R_i представляет собой вычислительную структуру, назначенную для решения i-ой функциональной задачи. Свободные ВМ вычислительной среды включаются в состав защитной среды и вместе с назначенными ВМ защитной среды разделяются на указанные r множеств. В ИС множество ВМ вычислительной среды разделяется без привязки к решаемым задачам на r – 1 одинаковых подмножеств R и оставшееся подмножество $s_m \leq R$. Следовательно, $m=(r-1)R+s_m$. Аналогично $k=(r-1)S+s_k$, где $s_k<S$.

Структурная схема ИС с раздельной АЗ без средств управления и коммутации показана на рис. III.3.2 а.

Отказоустойчивость ИС с раздельной АЗ обеспечивается на уровне каждой пары взаимосвязанных множеств R_i и S_i. Такая организация системы способствует построению простого и надежного управляющего органа каждой пары подмножеств и надежного малогабаритного коммутационного поля. Кроме того, в параллельных ИС отказы на уровне автономных пар подмножеств носят в большинстве своем частичный характер и не приводят к срыву функционирования ИС в целом.

Вместе с тем, организация раздельной АЗ существенно ограничивает возможности активной защиты вследствие отсутствия связей между различными парами подмножеств. Поясним это на примере. Пусть в течение времени управления в ИС используется для решения функциональных задач m = 200 ВМ и для обеспечения отказоустойчивости $к$ = 42 ВМ. Предположим, что

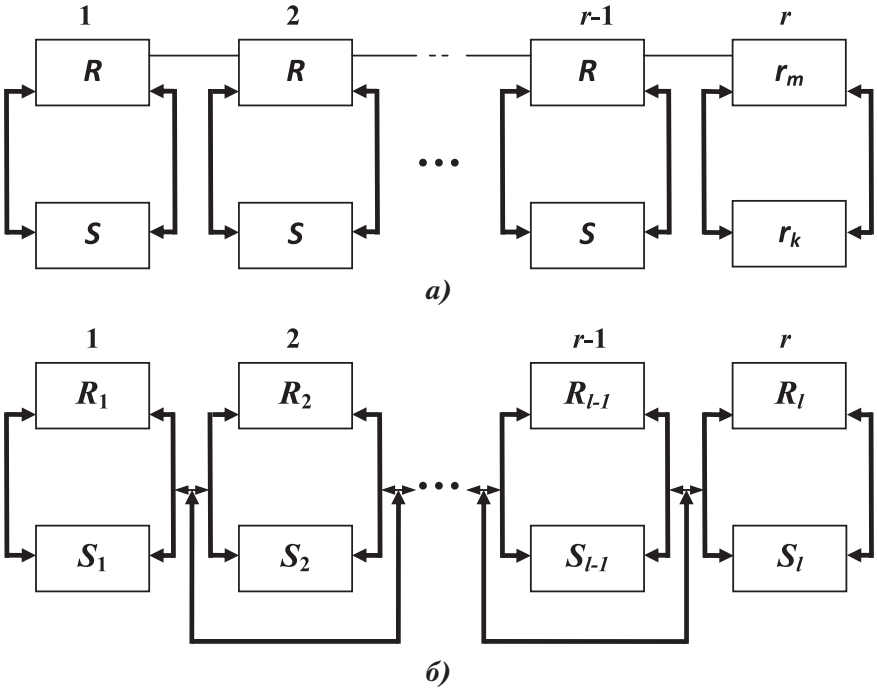


Рис. III.3.2. Принцип организации активной защиты: а) раздельной; б) смешанной

и вычислительная, и защитная среда разделены на 19 равных подмножеств с числом ВМ в каждой паре подмножеств $R = 10$, $S = 2$ и одно подмножество с числом ВМ $R = 10$, $S = 3$. В случае отказа более двух ВМ в одной паре подмножеств наступает отказ всей системы, хотя остаются исправными более 20 избыточных ВМ. Таким образом, за простоту организации АЗ приходится расплачиваться снижением ее эффективности.

3) **со смешанной АЗ**, когда пары подмножеств связаны с другими парами информационными и адресными шинами. Структурная схема параллельной ИС со смешанной АЗ без средств управления и коммутации показана на рис. III.3.2 б. В такой системе в полной мере используются возможности защитной среды, – осуществляется перераспределение избыточных ВМ между парами подмножеств по запросам управляющих органов соответствующих пар. Однако при этом возможны конфликты в системе, поскольку каждая пара подмножеств «тащит одеяло на себя», оголяя подмножества избыточных ВМ других пар. Разрешение этих конфликтов в параллельных ИС осуществляется на основе введения приоритетов нар подмножеств в соответствии с приоритетами решаемых ими функциональных задач. Парам высших приоритетов доступны избыточные модули всех других пар. В результате без защиты могут оказаться низкоприоритетные вычислительные подмножества ВМ. Чтобы в наибольшей мере оградить ИС от таких ситуаций, предусматривается следующая логика перераспределения избыточных ВМ между парами подмножеств при запросах соответствующих управляющих органов (запросы делаются в случаях отказов ВМ):

– все пары подмножеств упорядочиваются по приоритетам и отдельно по убыванию отношений S_i/R_i;

– запросы всех приоритетных пар обслуживаются путем перераспределения избыточных модулей из пары подмножеств с наименьшим приоритетом до тех пор, пока в этой паре оста-

нется только один избыточный модуль. Эта пара исключается из списка приоритетов и корректируется соответствующее ей число в списке отношений. Очередные запросы направляются к оставшейся в списке приоритетов наименее приоритетной паре. Если запрос поступил от вычеркнутой из списка приоритетов пары подмножеств (а это имеет место в случае полного отсутствия в ней исправных избыточных ВМ – в случае образования в ней пустого подмножества защитных модулей), то прерывается обслуживание всех запросов и из пары с наибольшим отношением S_i/R_i выделяется избыточный ВМ в вычеркнутую из списка приоритетов пару подмножеств, от которой поступил запрос на возобновление активной защиты. Затем вновь продолжается обслуживание запросов приоритетных пар, исключение из списков приоритетов очередной неприоритетной пары с одним избыточным модулем и так далее до тех пор, пока во всех парах подмножества защитных ВМ будут содержать по одному избыточному ВМ. После этого по запросам приоритетных пар перераспределяются модули из неприоритетных пар, которые остаются без активной защиты до моментов восстановления отказавших ВМ и реинтеграции их в ИС.

III.3.4.3.Уровни активной защиты

По уровню активной защиты возможны следующие способы построения ИС:

- с **одноуровневой защитой**, когда каждое подмножество основных модулей или вся вычислительная среда защищены только одним избыточным ВМ. Такая активная защита чрезвычайно экономична, проста в управлении, однако далеко не всегда обеспечивает высокий уровень вероятности успешной адаптации системы к отказам ВМ и во многих случаях не удовлетворяет требованиям по отказоустойчивости, предъявляемым к ИС реального времени. Поэтому при построении ИС с активной

защитой целесообразно предварительно оценить возможности применения одноуровневой АЗ;

- с **многоуровневой защитой**, когда каждое подмножество защитной среды содержит не менее двух избыточных ВМ. Верхняя граница количества избыточных ВМ, взаимодействующих с основными, составляет m при организации общей АЗ или $S_i = R_i$ в каждой паре подмножеств в системах с раздельной или смешанной АЗ.

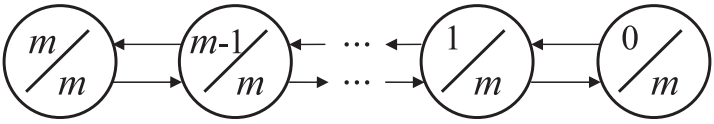


Рис. III.3.3. Граф состояний непополняемой активной защиты

Многоуровневая АЗ включает в себя одноуровневую. На рис. III.3.3 показан ориентированный граф укрупненных состояний непополняемой многоуровневой защиты в восстанавливаемых системах. Вершины помечены символами m/m, $(m–1)/m$, ..., $(m–j)/m$, ... $1/m$, $0/m$, означающими наличие в защитной среде соответственно m, ..., $m–j$, ..., 1 исправных модулей, либо отсутствие исправных избыточных ВМ ($0/m$). Правосторонние направления переходов в графе соответствуют событиям нарушений функционирования ВМ, левосторонние – их восстановлениям. Термин «непополняемой» АЗ введен, чтобы подчеркнуть отсутствие дополнительного источника избыточных ВМ.

В целях поддержания АЗ на исходном или допустимом уровне целесообразно применение дополнительного множества ненагруженных избыточных ВМ, объединенных общим коммутационным полем. Такая многоуровневая защита является пополняемой и содержит не более $2m$ избыточных ВМ. Аналогичный эффект в обеспечении отказоустойчивости ИС, но при меньшей избыточности может быть достигнут при реализации пополняемой многоуровневой смешанной АЗ.

Основные возможности многоуровневой активной защиты:

• замещение функций отказавшего основного ВМ любым исправным избыточным модулем без специальной реконфигурации ИС;

• обеспечение нескольких уровней вероятности успешной адаптации системы к отказам ВМ. Так, например, если пара подмножеств состоит из R_i=4 ВМ соответственно в основной и защитной группах, то при работе этой пары имеют место разные вероятности успешной адаптации: β_0 (все ВМ исправны), β_1 (отказал один любой из восьми ВМ), β_2 (отказали два любых ВМ) и β_3 (отказали три любых ВМ и активная защита обеспечивается с помощью только одного ВМ). Очевидно, что $\beta_0 \geq \beta_1 \geq \beta_2 \geq \beta_3$, поскольку с уменьшением количества исправных избыточных ВМ увеличивается случайное время адаптации ИС к отказам: $\beta_i = P(v_i < \tau_Д - t_у)$, где $t_у$ – время устранения нарушения ВМ;

• защита от программных ошибок и аппаратурных сбоев, поскольку во всех тактах АЗ осуществляется параллельная работа пар ВМ

На рис. III.3.4 приведено образное размещение вычислительной и защитной среды на двух дисках, посаженных на общую ось.

Основные модули обозначены номерами 1, 2, ..., 8, избыточные модули – номерами 11, 21, ..., 81. Оба диска проворачиваются в противоположные стороны. На рис.III. 3.3а показано исходное состояние дисков, при котором образуются пары ВМ 1 – 11, 2–21, ..., 8–81. На рис. III.3.3 б показано положение дисков после поворота их в противоположные стороны на шаг многоуровневой зашиты, равный 2. При этом образуются пары ВМ 1–71, 2–81, 3–11, … 8–61. Если зафиксировать один диск, то шаг защиты равен 1. Возможно осуществить шаги защиты 3 или 4 или 5 и т. д. Увеличение шага защиты позволяет избежать ошибок в работе соседних пар ВМ (в частности 1 – 11, 2–21, 3–31), которые могут иметь место под воздействием, например, поме-

хи по общей цепи питания. Изменение шага защиты повышает также гибкость управления на пониженных уровнях АЗ, когда количество избыточных ВМ меньше основных. В целом многоуровневая АЗ обладает большими потенциальными возможностями в обеспечении отказоустойчивости ИС.

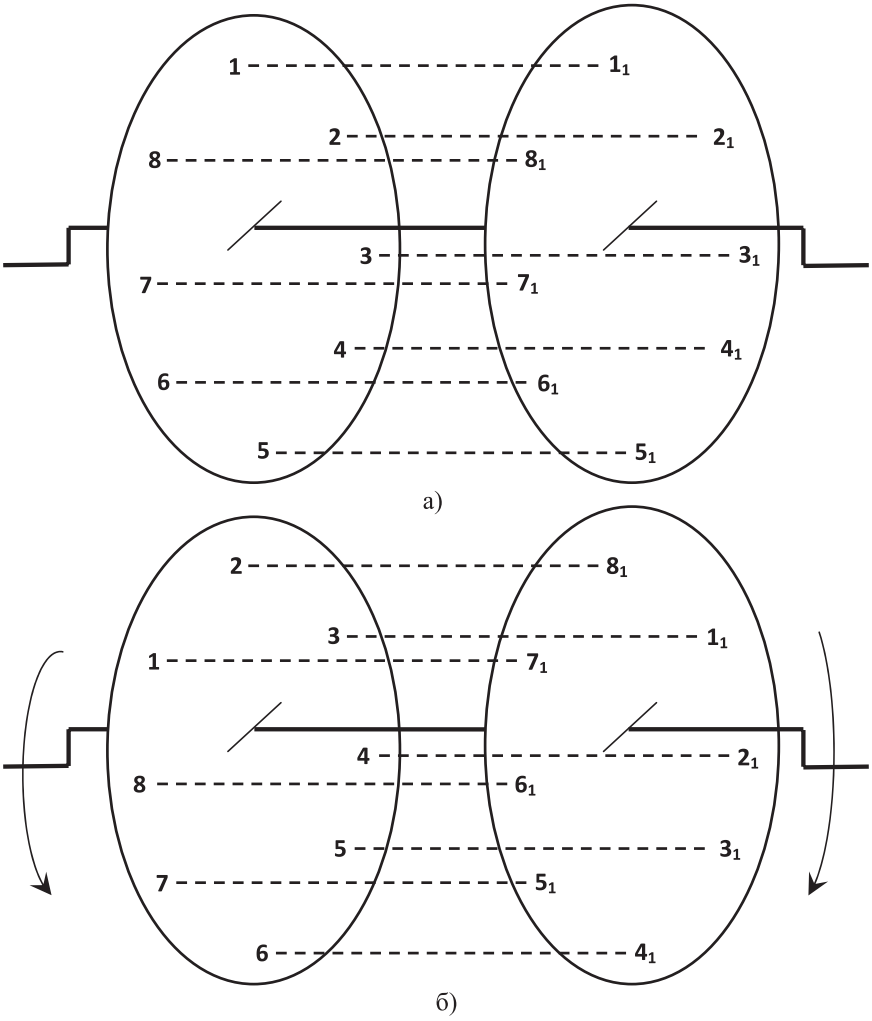


Рис. III.3.4. Формирование пар ВМ в ИС с многоуровневой АЗ типа m/m: а – исходные пары модулей; б – пары модулей, сформированные после реализации встречного шага защиты.

III.3.5. Способы обнаружения и устранения неисправностей в системах с активной защитой

Обнаружение неисправностей ВМ в системах с АЗ представляет собой логический процесс, основанный на результатах сравнения выходных данных пар параллельно работающих модулей, а также на логическом анализе изменений состояний вычислительного процесса в течение нескольких циклов и тактов АЗ, а также на различиях в характере проявления различных типов неисправностей.

Обнаружение неисправностей с помощью АЗ не исключает возможности применения в ВМ специальных аппаратных и программных средств наблюдаемости состояний модулей. Однако ставка делается на собственные силы, которые создаются в результате целенаправленных изменений взаимодействий между основными и избыточными ВМ и формируемых на этой основе цепочек логических условий, позволяющих в конечном итоге обнаруживать факты возникновения различных типов неисправностей, классифицировать их на отказы ВМ, собственные и привнесенные программные ошибки, локализовать и определять их местонахождение.

Возможно применение одного из следующих подходов к обнаружению неисправностей ИС с АЗ:

– продолжение вычислительного процесса после обнаружения факта неисправности;

– возврат вычислительного процесса на один такт АЗ или к предыдущей контрольной точке при наличии сигнала о неисправности.

Первый подход заключается в следующем. Неисправность одного из пары параллельно работающих ВМ обнаруживается в конце текущего такта АЗ по несовпадению выходных результатов работы этих модулей. Эта неисправность может быть сбоем

или отказом одного из двух ВМ данной пары. Классификация неисправности временно не выполняется, а избыточный модуль данной пары подключается к очередному основному ВМ и параллельно с ним реализует группу программных модулей, предусмотренных в работе очередного основного ВМ. Если выходные результаты работы этой пары ВМ также не совпали, то есть основания считать, что причина обоих несовпадений результатов заключается в отказе избыточного ВМ. Если же наблюдается совпадение результатов работы очередной пары модулей и при этом исключается неправдоподобная возможность случайного совпадения ошибочных результатов в работе обоих ВМ, то следует заключить об их исправности. Тогда относительно предыдущей пары справедливо одно из следующих суждений: отказал основной ВМ – сбойная ошибка в основном, либо в избыточном, либо в обоих ВМ.

Итак, обнаружен факт неисправности в паре ВМ. Тип неисправности и ее местонахождение неизвестно. Главное отличие первого подхода от второго заключается в продолжении работы системы без анализа исправности основного ВМ предыдущей пары. Ресурсы производительности ВС не расходуются на проверку подозреваемого основного ВМ (для обеспечения этого, естественно должны быть локализованы возможные ошибочные результаты в работе предыдущей пары модулей). Такое состояние системы сохраняется до образования новой пары с участием подозреваемого ВМ. Теперь уже возможно выполнение следующих операций наблюдаемости ранее подозреваемого ВМ, а именно, классификации неисправности на сбой или отказ и определения отказавшего модуля, если таковой имеется. Действительно, в случае совпадения результатов работы новой пары с участием подозреваемого ВМ есть основания принять, что этот модуль исправен, а зафиксированная ранее неисправность вызвана сбойной ошибкой, которая более не повторилась.

Таким образом, с помощью рассмотренного подхода есть возможность обнаруживать неисправный ВМ без остановки вычислительного процесса в системе. В этом его основное достоинство. Однако ему присущ существенный недостаток, – время обнаружения отказавшего ВМ превышает длительность цикла АЗ, что при малой избыточности системы и малой длительности допустимого времени перерыва в работе системы может привести к низкой эффективности АЗ. Кроме того, данный подход не позволяет обнаруживать и своевременно устранять программные ошибки.

Второй подход основывается на возврате вычислительного процесса, выполняемого парой ВМ, к последней контрольной точке, если в конце текущего такта обнаружено несовпадение выходных результатов работы этих модулей. Речь идет о выполнении легкого рестарта глубиной в один такт АЗ. При таком подходе совмещаются операции обнаружения факта неисправности, ее классификации и определения местонахождения до уровня модуля.

Рассмотрим некоторые способы АЗ, которые основываются на возврате вычислительного процесса и отличаются между собой возможностями обнаружения различных типов неисправностей.

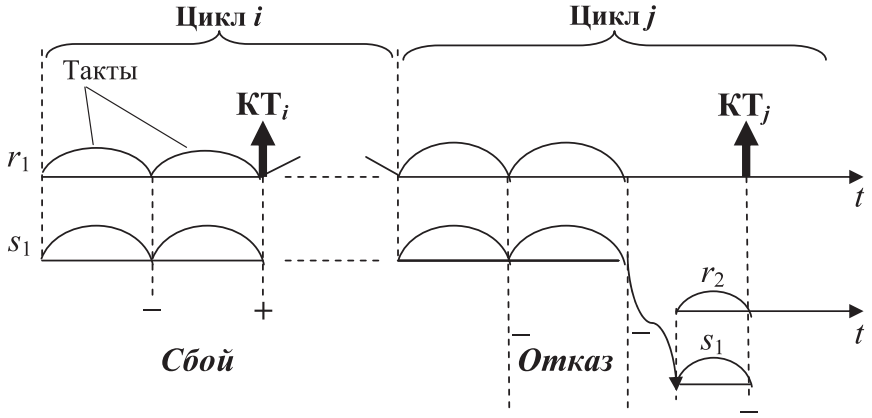
Суть способа обнаружения сбоев и отказов ВМ показана на рис. III.3.5 а. В *i*-ом цикле АЗ сформирована пара ВМ из основного $r_1 \in R$ и избыточного $s_1 \in S$. В течение такта АЗ параллельно работают оба ВМ, результаты сравниваются. Пусть зарегистрировано их несовпадение (на рис. III.3.5 а это событие помечено в конце такта АЗ знаком «–»). Поскольку способ ориентирован на обнаружение только неисправностей аппаратуры, то причина несовпадения либо в сбое модуля r_1 или s_1, либо в отказе одного из этих модулей. События одновременного возникновения сбоев или отказов обоих модулей в данном способе исключаются

из-за их чрезвычайно малой вероятности. Итак, возможно четыре исхода. Чтобы классифицировать неисправность, выполняется повторный счет по прежним исходным данным. Результаты повторной работы обоих модулей снова сравниваются. Предположим, что результаты совпали (это событие помечено на рис. III.3.5 а знаком « + »). Этот исход означает отсутствие отказов любого из модулей, а также отсутствие их сбоев в текущем такте АЗ. Причина несовпадения результатов в предыдущем такте обусловлена только сбоем одного из ВМ. Следовательно, по результату работы основного ВМ в текущем такте нужно сформировать контрольную точку i-го цикла ($КТ_i$), а также выдать полученный правильный результат пользователю.

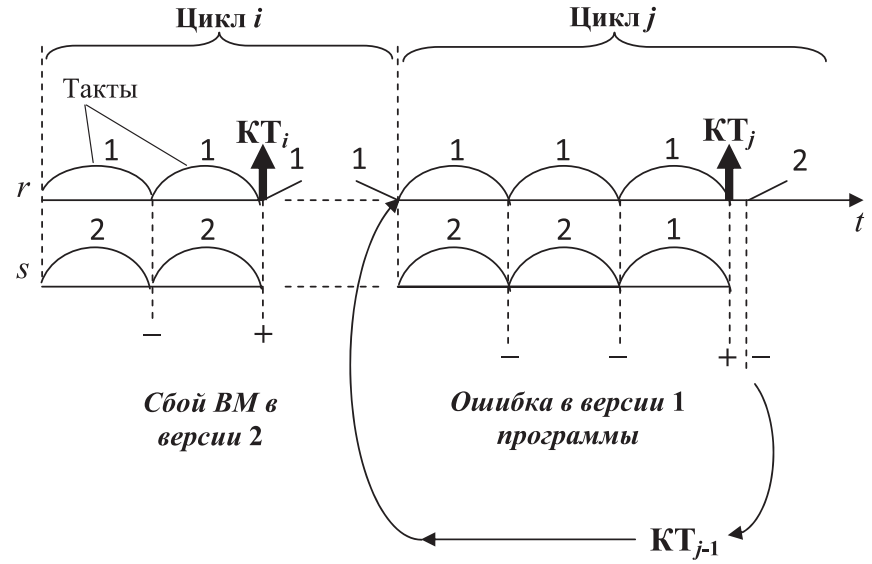
Предположим теперь, что в j-ом цикле АЗ зафиксировано несовпадение результатов работы ВМ r_1 и s_1, как в очередном, так и повторном такте АЗ. Есть основания полагать наличие отказа либо в модуле r_1 либо в модуле s_1, Классификация неисправности выполнена, а для определения отказавшего ВМ из этой пары формируется пара $s_1 - r_2$, где $r_2 \in R$. Работа этой пары осуществляется по программе модуля r_2 в течение такта АЗ. В случае их несовпадения (рис. III.3.5 а) принимается решение об отказе избыточного модуля s_1, а по выходным данным модуля r_2 корректируется контрольная точка в j-ом цикле ($КТ_j$) и полученный результат выдается пользователю. Если бы результаты работы пары $s_1 - r_2$ совпали, то следовало бы принять решение об отказе ВМ r_1 и содержимое $КТ_j$ нужно было бы сформировать по выходным данным ВМ s_1 в паре с модулем r_1.

Описанный вкратце способ повторных счетов позволяет существенно сократить время обнаружения отказавшего ВМ по сравнению с рассмотренным ранее способом. Вместе с тем, он не предусматривает обнаружение программных ошибок. Кроме того, за сокращение длительности существования отказа ВМ приходится расплачиваться некоторым снижением производи-

тельности ИС. Оценим эту величину. Введем следующие обозначения: t_y – среднее время управления в системе, t_C – среднее время между сбойными ошибками ВМ, τ – среднее время такта АЗ. Тогда среднее количество сбойных ошибок в одном рабо-



а)



б)

Рис. III. 3.5. Суть обнаружения неисправностей на основе АЗ: а) сбоев и отказов; б) сбоев, отказов и программных ошибок

тающем ВМ в течение времени управления определяется соотношением t_y/t_C, а среднее время непродуктивной работы ВМ вследствие однократных повторных счетов равно $(\tau \cdot t_y)/t_C$.

Таким образом, потери в производительности каждого ВМ из-за повторных счетов отнесенные к длительности такта АЗ определяются отношением $(\tau \cdot t_y)/\tau t_C = t_y/t_C$. На основе имеющихся опытно-теоретических данных можно принять, что частота сбойных ошибок ВМ на 3–4 порядка превышает частоту отказов [19, 47], тогда значение t_C относительно ВМ составляет примерно 1 –10 ч. В свою очередь, длительность такта АЗ составляет примерно 10–100 мс. Следовательно, дополнительные потери в производительности каждого ВМ в системе при реализации способа АЗ на основе повторных счетов незначительны.

Рассмотренный способ обнаружения неисправностей в ИС с АЗ может быть развит также и для обнаружения программных ошибок. Для этого необходимы хотя бы две версии функциональной программы, выполняемой соответствующим ВМ. Обозначим основную программу версией 1, а программу приближенного решения этой же задачи версией 2. Для обнаружения программных ошибок необходимо ввести граничные значения расхождений между результатами выполнения первой и второй версии каждой программы.

Предположим, что оба условия разделения обеих версий программ на группы программных модулей выполнимы. Тогда применим следующий способ обнаружения всех типов неисправностей ИС в процессе функционирования системы. В i-ом цикле АЗ образуется пара модулей с номерами $r \in R$ и $s \in S$, причем, в течение такта АЗ, модуль r реализует версию 1, а модуль s – версию 2 программы (рис. III.3.5 б). Задача версии 2, выполняемая модулем s, может быть завершена раньше задачи версии 1. Тогда результат хранится в выходных регистрах модуля s до завершения задачи версии 1, затем оба результата сравниваются.

Если их расхождения превышают граничные значения (знак «–» на рис. III.3.5 б), то принимается решение о наличии неисправности (при этом следует иметь ввиду, что оба ВМ работали при одинаковых исходных данных). Обнаружен факт неисправности. Причин неисправности много – это отказы или сбои модулей r или s, собственные программные ошибки в версии I или 2. Необходимо сузить число возможных причин. Для этого производится классификация неисправностей путем повторного счета на обоих модулях по тем же версиям программы. Предположим, теперь расхождение результатов в допустимых пределах (знак « + » по завершению второго такта АЗ на рис. III.3.5 б). Тогда очевидно отсутствие отказов в обоих модулях и отсутствие программных ошибок в обеих версиях. Причина предыдущей тревоги обусловлена сбоем ВМ, в который в текущем такте не повторился. Поэтому правильный результат работы основного модуля *r* используется для корректировки контрольной точки на данный момент времени. Следовательно, этот способ, подобно ранее рассмотренным, позволяет в процессе функционирования ВС за счет априорной реконфигурации и легкого рестарта обнаруживать отказы и сбойные ошибки.

Для иллюстрации возможности данного способа по обнаружению программных ошибок предположим, что в *j*-м цикле АЗ снова обнаружено недопустимое расхождение результатов работы модулей *r* и *s* по версиям 1 и 2 соответственно. При повторном счете это событие возобновилось (знак «–» по завершению второго такта АЗ в *j*-м цикле на рис. III.3.5 б). Теперь уже из перечня возможных неисправностей сбойные ошибки модулей исключаются вследствие чрезвычайно малой вероятности того, что в предыдущем такте АЗ имел место сбой, например, модуля *r*, а также сбой модуля *s* в текущем такте. Сохраняются предположения об отказе одного из модулей или о наличии программной ошибки в одной из двух версий программ. Это могло

внести ошибку в содержимое $КТ_{ij}$ и, следовательно, привести к работе модулей *r* и *s* с искаженными данными в *j*-м цикле. Чтобы разобраться с этими неясностями, в рассматриваемом способе предусматривается автоматическое выполнение следующих операций. После недопустимых расхождений результатов в двух смежных тактах АЗ модуль *s* в третьем смежном такте выполняет вместо версии 2 версию 1 программы. Для этого выделяется необходимое время на загрузку памяти модуля операндами и командами версии 1 программы. По данным [48] для микропроцессорных систем это время загрузки не превышает 10% от времени выполнения программных модулей. В случае очередного несовпадения результатов очевидно наличие отказа одного из двух ВМ. Операции нахождения конкретного отказавшего ВМ из пары аналогичны тем, которые декларировались в параграфе III.3.1 и пояснены на рис. III.3.5 а. Если же в третьем смежном такте АЗ результаты работы обоих модулей совпали, как это показано на рис. III.3.5 б, то исключаются отказы и сбои ВМ и принимается решение о наличии программной ошибки в версии 1 или 2. Остается невыясненным вопрос о том, на каком участке программы возникла ошибка. Поэтому сразу же за обнаружением факта программной ошибки анализируется непротиворечивость данных, которые были сформированы в результате работы модуля r в автономном режиме на интервале времени между тактом завершения предыдущей парной работы и тактом АЗ текущей парной работы, в котором обнаружена неисправность. Если данные непротиворечивы, то программная ошибка возникла в подпрограмме, выполненной в текущем такте парной работы. Ошибка локализована в пределах такта АЗ.

Более вероятна противоречивость данных, поскольку обычно алгоритмы задач строятся в рекуррентной форме, и поэтому скрытая ошибка в программе, не приведшая сразу к явным нарушениям вычислительного процесса, через определенное число шагов

вычислений обычно вызывает недопустимые искажения результатов. Если данные противоречивы (рис. III.3.5 б), то имела место программная ошибка в версии 1 после предыдущей контрольной точки, которую обозначим $КТ_{j1}$ Восстановление вычислительного процесса, в частности, возможно путем рестарта с точки $КТ_{j1}$ по версии 2 до текущего такта АЗ парной работы модулей *r* и *s*.

Способы устранения неисправностей в системах с АЗ зависят от типа обнаруженной неисправности. Сбойные ошибки устраняются в течение такта АЗ путем повторных счетов. Программные ошибки в одной версии изолируются, а вычислительный процесс восстанавливается путем возврата к предыдущей контрольной точке и повторения вычислений с помощью другой версии программы. Отказы основных ВМ устраняются путем автоматической замены их исправными избыточными модулями и повторения процесса опять-таки с предыдущей контрольной точки. Таким образом, время устранения неисправности равно $t_y = k\tau$, где $k = 1$ в случае сбойной ошибки. В случае отказов ВМ или программных ошибок значение t_y зависит от организации защиты и определяется количеством тактов АЗ, прошедших от момента образования предыдущей контрольной точки до момента определения типа и места неисправности.

III.3.6. Временные интервалы активной защиты

Возможности АЗ определяются следующими временными интервалами:

– длительностями решения функциональных задач и пауз между ними;

– допустимым временем перерыва в работе.

Значения этих длительностей зависят от внешней среды, при взаимодействии с которой функционирует ИС реального времени. При организации АЗ их следует рассматривать в качестве

исходных параметров системы обеспечения отказоустойчивости ИС, поскольку они не подлежат регулированию.

Обозначим длительности решения задач и пауз между ними символами υ_1 и υ_2 соответственно. Эти величины случайны. От их соотношения существенно зависит эффективность АЗ. В тех случаях, когда $\upsilon_1 >>> \upsilon_2$, можно полагать, что ИС работает без пауз с предельной информационной нагрузкой. Эти условия наиболее благоприятны для реализации АЗ, так как паузы не тормозят динамическую реконфигурацию системы и, следовательно, в полной мере используются возможности контроля работоспособности ВМ на системном уровне.

Наиболее типичны соотношения между случайными интервалами времени υ_2 и υ_1 такие, когда $\upsilon_2 << \upsilon_1$. Эти соотношения имеют место при решении функциональных задач. Времена их решения вычислительными модулями много больше длительностей пауз. Сравнительно большие длительности задач позволяют разделить их на такты АЗ. На рис. III.3.6 а показаны временные интервалы АЗ при указанных соотношениях между величинами υ_2 и υ_1, а также при постоянных длительностях тактов АЗ и при допустимом времени $\tau_Д$ перерыва в работе системы. Время $\tau_Д$ отсчитывается от момента возникновения неисправности, который на рис. III.3.6 а проиллюстрирован затемненным тактом АЗ. При построении системы с АЗ следует оперировать количеством тактов $x_Д$ защиты, которые размещаются в интервале времени $\tau_Д$. Каждый такт АЗ определяется суммой $\tau = \tau^* + \tau_1$, где τ^* – время работы пары ВМ с постоянными тактами, а τ_1 – время, затрачиваемое на перекоммутацию модулей, подготовку пары к работе и сравнение результатов. Эти временные интервалы выбираются таким образом, чтобы выполнялось неравенство $\tau_1 << \tau^*$.

Постоянные такты АЗ применяются в системах с конвейерной обработкой информации, в вычислителях быстрого преобразования Фурье и некоторых других. В большинстве управляю-

щих ИС вычислительные модули выполняют самостоятельные функции и поэтому активная защита реализуется по случайным тактам длительности υ со средним значением τ.

На рис. III.3.6 б показаны временные интервалы работы системы со случайными интервалами υ, υ_1, υ_2, а также со случайной непрерывной величиной υ_3 допустимого времени перерыва в работе. Естественно, что при этом допустимое количество тактов АЗ $x_Д$ также является случайной величиной, описываемой соответствующим дискретным распределением.

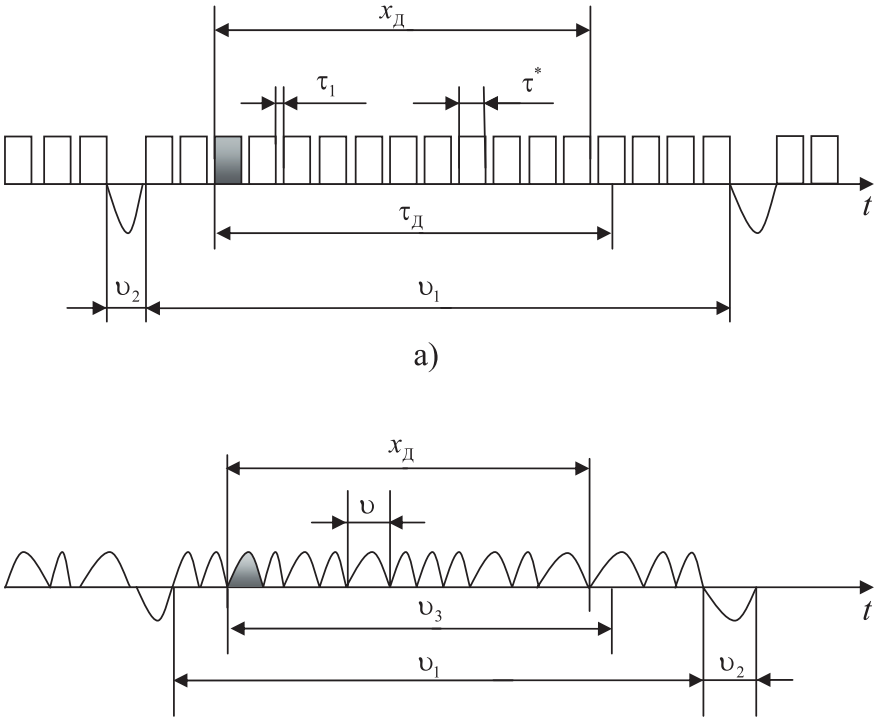
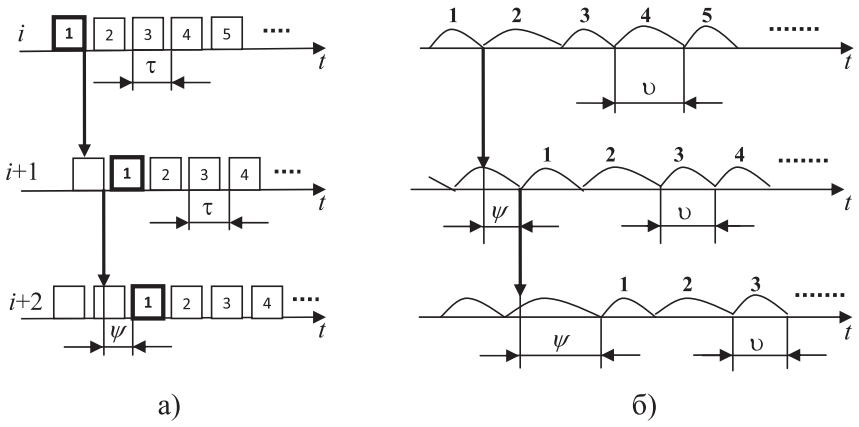


Рис. III.3.6. Временные интервалы активной защиты при больших длительностях решения задач и коротких пауз между ними: а – условия постоянного допустимого времени $\tau_Д$ перерыва в работе ИС и постоянных тактов АЗ τ; б – условия случайного допустимого времени υ_3 перерыва в работе ИС и случайных тактов АЗ.

В процессе функционирования ИС со случайными тактами АЗ имеет место время ожидания ψ начала параллельной работы контролирующего ВМ с очередным основным модулем. Это время обусловлено тем, что при функционировании различных основных ВМ такты АЗ не совпадают во времени. Поэтому после завершения параллельной работы с *i*-м основным ВМ контролирующий модуль подключается к (*i*+1)-му основному в интервале такта работы последнего и находится в режиме ожидания до завершения текущего такта и начала следующего. Таким образом, цикл АЗ состоит из случайных временных интервалов ψ+υ. Количество таких временных интервалов в цикле зависит от типа дисциплин АЗ, которые будут рассмотрены отдельно.

На рис. III.3.7 приведены реализации безошибочного функционирования модульной системы с постоянными (рис. III.3.7 а) или со случайными (рис. III.3.7 б) тактами АЗ.



а) б)

Рис. III.3.7. Временные диаграммы переключения контролирующих ВМ при больших длительностях решения задач: а – условия постоянных тактов АЗ; б – условия случайных тактов АЗ.

Следует отметить, что случайные величины υ и ψ, в отличие от исходных υ, υ_1, υ_2, υ_3 (или $\tau_Д$), являются управляемыми. Их характеристики рассчитываются при проектировании системы.

Поскольку времена ожидания ψ начала параллельной работы ВМ существенно увеличивают общее время адаптации ИС к ошибке, то необходимо изыскивать пути сокращения этих непроизводительных простоев системы обеспечения отказоустойчивости. Один из таких путей заключается в отыскании minψ на каждом шаге АЗ и в выборе на этой основе очередного основного ВМ для организации его параллельной работы с контролирующим модулем. Поясним суть такой организации системы на примере. Пусть в раздельной АЗ предусмотрено 5 основных и один контролирующий ВМ. Предположим, что завершен такт параллельной работы третьего основного и контролирующего ВМ. После этого сравниваются остаточные времена тактов АЗ модулей 1, 2, 4 и 5 (ψ_{11}, ψ_{21}, ψ_{41}, и ψ_{51}). Допустим, что $\min\psi=\psi_{41}$. Тогда контролирующий ВМ подключается к четвертому основному, в течение времени ψ_{41} подготавливается к параллельной работе с ним в течение последующего такта АЗ. По окончании этого такта вновь анализируются остаточные времена тактов АЗ теперь уже ВМ 1, 2, 3 и 5 (ψ_{12}, ψ_{22}, ψ_{32}, и ψ_{52}). Если $\min\psi=\psi_{52}$, то контролирующий ВМ подключается к пятому основному и т. д.

Таким образом, в системах с оптимизацией управления АЗ кроме рассмотренных ранее случайных величин фигурирует также и время minψ.

Изложенный вариант оптимизации управления АЗ будем называть управлением с постоянным составом объектов оптимизации. Ему присущ недостаток, который заключается в следующем. В течение ограниченного времени решения задач управления частота контроля различных основных ВМ неравномерна, – отдельные модули могут вообще не проверяться вследствие малого объема выборки.

Более предпочтителен вариант оптимизации управления с убывающим составом объектов. Он отличается от преды-

дущего варианта тем, что проверенный основной ВМ исключается из состава анализируемых объектов. Этот процесс продолжается до завершения однократной проверки состояния всех основных ВМ, после чего снова анализируется полный состав всех объектов оптимизации и т. д. Данный вариант оптимизации управления обеспечивает гарантированный системный контроль каждого основного ВМ в каждом цикле АЗ.

Выходной временной характеристикой ВС с АЗ следует считать случайное время v адаптации системы к отказам. Эта характеристика определяется рассмотренными временными интервалами, а также количеством m основных, k избыточных ВМ и принятой дисциплиной активной защиты, т.е.

$$v = f(\tau_Д, \upsilon, \upsilon_1, \upsilon_2, \upsilon_3 \min \psi, m, k, Д)$$

Дисциплина ***Д*** определяет способ организации АЗ, способы обнаружения и устранения неисправностей. В этой дисциплине учитывается архитектура системы. Естественно усложнение управления в системах с оптимизацией защиты. Определяющее влияние на выбор структуры АЗ оказывает допустимое время перерыва в работе системы.

III.3.7. Дисциплины активной защиты

Под дисциплиной АЗ подразумевается определенное сочетание способов организации и функционирования системы обеспечения отказоустойчивости, принятое исходя из временных интервалов ее работы и с учетом архитектуры ИС.

Разработанные способы АЗ позволяют создать широкий спектр различных дисциплин защиты. В данном параграфе ограничимся рассмотрением наиболее характерных из них: базовых и модульных. Базовые дисциплины обеспечивают активную защиту при минимальной избыточности системы.

Эти дисциплины служат основой для построения многоуровневых дисциплин защиты, поскольку в результате постепенной деградации ИС многоуровневые дисциплины трансформируются в базовые (одно и двухуровневые). При этом должен быть выдержан единый стиль построения многоуровневой дисциплины и ее базовой дисциплины. Каждая базовая дисциплина – это алгоритм взаимодействия одного или двух избыточных ВМ с множеством основных модулей (при общей АЗ) или с подмножеством основных ВМ (при раздельной или смешанной АЗ).

Модульные дисциплины в отличие от базовых дисциплин предназначены для организации взаимодействия строго ограниченного количества избыточных и основных ВМ. Это обеспечивает аппаратную реализацию управления модульной дисциплиной и позволяет сформировать модуль АЗ, а обслуживание множества всех основных ВМ возможно за несколько смежных тактов работы этого модуля.

III.3.7.1. Базовые дисциплины

Рассмотрим базовые дисциплины АЗ, которые обозначены символами ***Д1,..., Д7***.

Д1. Система с АЗ без возврата вычислительного процесса, один избыточный модуль на m основных, фиксированный контролирующий ВМ, контроль неприоритетный, считывание результата с контролирующего модуля. Принцип построения данной дисциплины для систем с конвейерной обработкой информации показан на рис. III.3.8. Одинаковыми прямоугольниками обозначены равные такту АЗ временные интервалы обработки информации отдельными ВМ. Буквой **o** обозначены временные интервалы функционирования основных (рабочих) модулей 1, 2, …, m. Буквой **k** обозначены интервалы подключе-

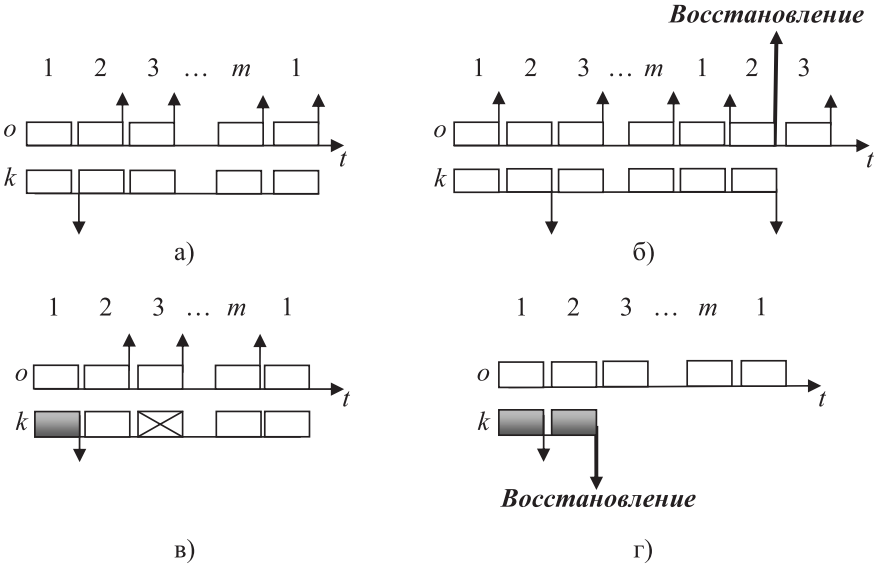


Рис. III.3.8. Принципы построения дисциплины Д1 при постоянных тактах А3: а – случай сбоя основного ВМ; б – случай отказа основного ВМ; в – случай сбоя контролирующего ВМ; г – случай отказа контролирующего ВМ.

ния контролирующего модуля к основным. Затемненный прямоугольник означает наличие в текущем такте работы соответствующего модуля ошибочного результата, который не выявлен встроенным контролем, однако обнаружен активной защитой. Перечеркнутый прямоугольник означает, что в текущем такте ошибочный результат обнаружен встроенным контролем; стрелка указывает номер ВМ, с которого считывается информация, две противоположные стрелки с одного выхода означают, во-первых, считывание информации и, во-вторых, замещение отказавшего модуля; жирная стрелка определяет перевод отказавшего модуля на восстановление.

На рис. III.3.8 а показана работа ИС при сбое в первом основном ВМ. Этот сбой обнаружен по результатам параллельной работы данного основного и контролирующего ВМ. Результаты не совпали, в системе зафиксирована ошибка. Информация

считывается с контролирующего ВМ, которому больше доверия, поскольку он сам контролируется основными ВМ в каждом такте АЗ. Первый основной модуль фиксируется в списке подозрительных в отношении состояния надежности ВМ. Через цикл АЗ, который в данной дисциплине равен $A = m$, вновь работает пара из первого основного и контролирующего ВМ. Результаты совпали. Принимается решение о сбое первого ВМ в предыдущем цикле АЗ. Этот модуль исключается из списка подозрительных модулей.

На рис. III.3.8 б показана работа ИС при отказе второго основного ВМ. Решение об отказе этого модуля принимается при наличии в ИС общей памяти в следующем цикле АЗ через m тактов. После этого контролирующий ВМ выполняет функции второго основного модуля и система работает в течение времени восстановления отказавшего модуля без защиты.

На рис. III.3.8 в показана работа ИС в условиях сбоя контролирующего ВМ. При несовпадении результатов работы пары, образованной первым основным и контролирующим модулями, больше доверия согласно алгоритму отводится контролирующему ВМ. С него считывается искаженная информация. Следовательно, от сбоев контролирующего ВМ, не обнаруженных его собственными средствами контроля, система не защищена. Этот недостаток дисциплины ***Д1*** имеет место и в случае отказа контролирующего ВМ. Хотя его отказ обнаруживается и устраняется за два такта АЗ (рис. III.3.8 г), в первом такте ошибочный результат запоминается в контрольной точке, сформированной по данным неисправного контролирующего ВМ (в рассматриваемом примере первого модуля). В результате этого возможно нарушение вычислительного процесса основного исправного ВМ.

Таким образом, данная дисциплина отличается простотой реализации, но имеет два недостатка: 1) система не защищена от сбоев и отказов контролирующего ВМ; 2) сравнительно

большое время адаптации ИС к отказам основных модулей: при наличии общей памяти в системе количество тактов в цикле АЗ $A = m$; количество тактов принятия решения об отказе основного ВМ $b = A = m$; количество тактов, затрачиваемых на восстановление вычислительного процесса с последней контрольной точки $x_y=m$.

Д2. Система с АЗ, содержащая m основных и 2 контролирующих ВМ. Переназначение модулей не предусмотрено, контроль неприоритетный, без повторного счета. Система с дисциплиной ***Д2*** построена следующим образом. Оба контролирующих ВМ подключаются к очередному основному модулю и в течение такта АЗ три ВМ решают одинаковую задачу. Правильный результат работы тройки ВМ определяется по мажоритарной логике. При этом также устанавливается ВМ, в выходном результате которого в данном такте защиты возникла ошибка. В следующем такте оба контролирующих ВМ подключаются к следующему основному модулю для совместной параллельной работы и т. д.

Таким образом, дисциплина ***Д2*** несколько сложнее ***Д1*** (содержит мажоритарную схему с восстанавливающим органом, дополнительный ВМ с входным и выходным коммутаторами). Однако эта дисциплина имеет существенные преимущества перед ***Д1***: повышена кратность резервирования; обеспечена защита от сбоев и отказов контролирующих ВМ; при отказе одного любого модуля дисциплина ***Д2*** трансформируется в дисциплину ***Д1.***

Вместе с тем, так же как и у предыдущей дисциплины, остается сравнительно большим время адаптации к отказам ($A = b = m$, $x_y=m$).

Д3. Система с АЗ, построенной также, как и в дисциплине ***Д1,*** но без выдачи результата при обнаружении несовпадения в текущем такте. Данная дисциплина предназначена для применения в специализированных ИС корреляционной обработки сигналов

с помощью быстрого преобразования Фурье (БПФ). Аналогичные допущения возможны в специализированных ИС, предназначенных для обработки информации с помощью метода максимального правдоподобия и т.д. Таким образом, дисциплина ***Д3*** имеет единственное отличие от ***Д1***, которое заключается в том, что при несовпадении результатов работы пары ВМ содержимое контрольной точки основного модуля этой пары не обновляется, а результат работы ВМ в данном такте блокируется. Этим достигается основное преимущество данной дисциплины перед ***Д1*** – ошибочный результат работы контролирующего ВМ не нарушает вычислительного процесса основного исправного модуля. Однако время адаптации к отказам ВМ немного увеличивается (при наличии в системе общей памяти $A = b = m$, x_y=2m).

Рассмотренные три дисциплины АЗ относятся к категории простейших с точки зрения организации управления ими. Эти дисциплины инерционны, особенно при отсутствии в ИС общей оперативной памяти. При автономности ВМ в системах с конвейерной обработкой информации невозможно обеспечить параллельную работу пары модулей сразу же после ее формирования. Для этого необходимо загрузить контролирующий ВМ операндами и программными модулями основного. Допустим, что время загрузки не превышает 10% от времени выполнения программных модулей [48]. Однако очередной такт работы проверяемого основного ВМ уже выполняется и поэтому контролирующий модуль 90% длительности этого такта находится в режиме ожидания до начала следующего такта АЗ, к которому он подготовлен в результате загрузки соответствующими программными модулями. Остается только дозагрузить его полученными в текущем такте АЗ операндами. Таким образом, в условиях автономности ВМ параметры A, b и x_y характеризуются следующими количествами тактов АЗ: ***Д1, Д2***: $A = b = m$, x_y=m; ***Д3*** : $A = b = m$, x_y=2m.

Д4. Система с АЗ и рестартом на один такт, содержащая m основных и один контролирующий ВМ, контроль неприоритетный, переназначение модулей не предусмотрено. В случае сбоя одного из пары ВМ выполняется повторный счет с прежними операндами. Совпадение результатов в очередном такте АЗ исключает возможность отказа модулей, сбой устранен, обновляется контрольная точка по задаче первого ВМ в i-ом цикле защиты. Если сбой ВМ обнаружен собственными средствами контроля, то естественно обновляется контрольная точка по данным первого основного ВМ. Случай отказа одного из пары ВМ обнаруживается с помощью рестарта на один такт АЗ. Если при решении одного и того же участка задачи в течение двух тактов результаты работы пары однотипных ВМ дважды не совпали, то контрольная точка не обновляется до выполнения совместной работы контролирующего ВМ с очередным основным (в данном примере третьим) ВМ. В случае совпадения результатов работы этой последней пары принимается решение об отказе предыдущего основного ВМ (в данном примере второго), обновляется контрольная точка для второго ВМ по данным контролирующего модуля, который теперь выполняет функции второго основного ВМ. Если же в трех смежных тактах АЗ результаты не совпали, то принимается решение об отказе контролирующего ВМ и система некоторое время в случае отсутствия исправного избыточного модуля может работать без АЗ.

Таким образом, относительно дисциплины ***Д4*** параметры A, b и x_y характеризуются следующими количествами тактов АЗ: $A = m$; $b = 2$; $x_y \le m+2$

Д5. Система с АЗ без рестарта, содержащая m основных и один контролирующий ВМ, считывание результатов при их несовпадении в паре не предусмотрено. В системе предусмотрено переназначение модулей, неприоритетный или приоритетный контроль.

Значение параметров: $A = b = \mathrm{INT}\,[(m+1)/2]$, $x_y \le m+1$

Д6. Система с АЗ с рестартом и переназначением ВМ, содержащая m основных и один контролирующий модуль. Организация обнаружения и устранения неисправностей такая же, как и в дисциплине ***Д4.***

Значение параметров: $A = \text{INT}[(m+1)/2]$; $b = 2$; $x_y \leq \text{INT}[(m+1)/2]+2$

III.3.7.2. Модульные дисциплины

Предназначены для применения в ИС с раздельной или гибридной АЗ и обеспечивают кратность резервирования *R/R* или *R/2R*, где *R* = 2, 3 или 4. Рассмотрим принцип построения типового несимметричного модуля АЗ типа 2/4 (дисциплина ***Д7***).

Д7. Система с АЗ с рестартом на один такт, переназначением модулей и неприоритетным контролем. Принцип построения дисциплины ***Д7*** для четырех основных и двух избыточных ВМ при условии, что все модули не обеспечены встроенным контролем, показан на рис. III.3.9. Обозначения на данном рисунке в основном соответствуют обозначениям рис. III.3.8. Пары цифр, проставленных в прямоугольниках, означают номера модулей, которые в текущем такте работают параллельно, причем первая цифра принадлежит контролирующему модулю, а вторая – контролируемому, который жестко закреплен за соответствующим каналом обработки информации. Контролируемые модули всегда четные (2 или 4), контролирующие – нечетные (1 или 3). Цифры у выходных стрелок означают номера каналов, с которых в текущем такте считывается выходной результат обработки информации.

На рис. III.3.9 а проиллюстрирована работа АЗ в условиях отсутствия сбоев и отказов модулей. Работа организована следующим образом. В текущем такте модули с нечетными номерами 1 и 3 контролируют модули с номерами 2 и 4 соответственно. При совладении выходных результатов работы модулей информация во второй и четвертый каналы считывается с контролирующих модулей 1 и 3 соответственно. Информацию первого и третье-

го каналов обрабатывают в текущем такте избыточные модули (ИМ) 1 и 2. В следующем такте контролируются ИМ с помощью контролирующих модулей 1 и 3.

Выходные результаты обработки информации считываются в первый и третий каналы с ИМ1 и ИМ2 соответственно. Модули 2 и 4 в этом такте работают без АЗ. На рис. III.3.9 б показана работа АЗ в условиях сбоя одного из модулей, в частности, модуля 1. Наличие сбоя приводит к несовпадению результатов работы модулей 1 и 2 в предыдущем такте. В этом случае со всех четырех каналов информация не считывается.

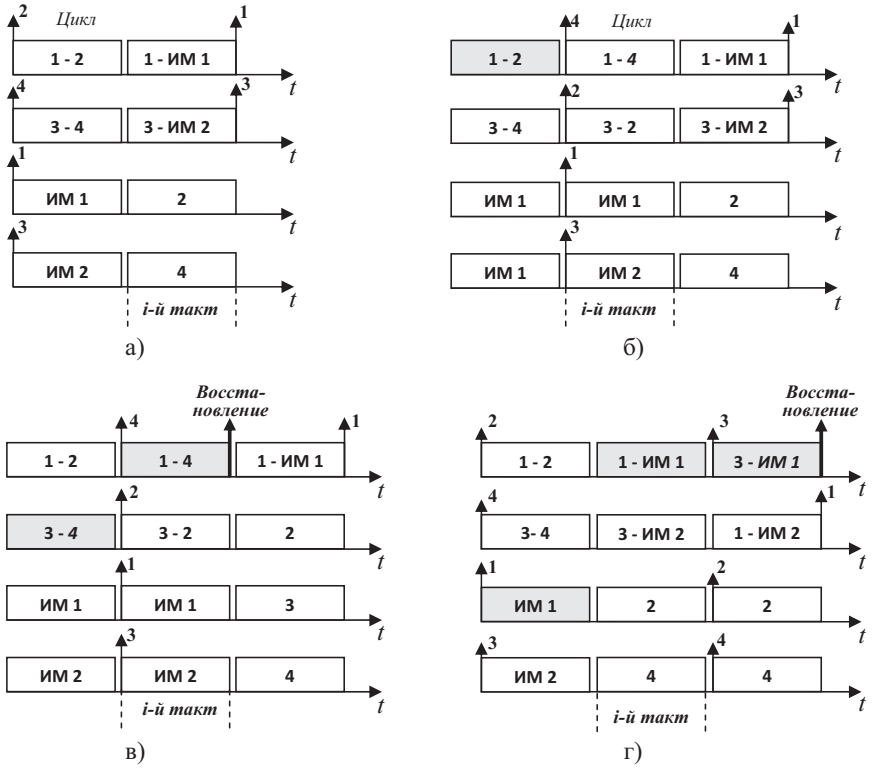


Рис. III.3.9. Принцип построения модуля АЗ типа 2/4 при постоянных тактах защиты: а – случай исправной работы; б – случай сбоя первого основного ВМ; в – случай отказа четвертого основного ВМ; г – случай отказа первого избыточного модуля.

В текущем такте осуществляется перекоммутация пар параллельно работающих модулей (вместо пары 1–2 коммутируется пара 1–4, а вместо пары 3–4 – пара 3–2). Информация в четвертый и второй каналы считывается соответственно с модулей 1 и 3. Дальнейшая работа АЗ осуществляется так же, как и в условиях отсутствия сбоев и отказов.

На рис. III.3.9 в показана работа АЗ в условиях отказа одного из четырех основных модулей, например, модуля 4. В этом случае отказавший модуль обнаруживается и замещается избыточным модулем в течение двух тактов. На третьем такте контролируется оставшийся ИМ1 с помощью модуля 1. Начиная с четвертого такта и до восстановления отказавшего модуля, АЗ работает с одним избыточным модулем.

На рис. III.3.9 г показана работа АЗ в условиях отказа одного из двух избыточных модулей, например, ИМ1. Отказ ИМ1 обнаруживается в течение трех тактов, затем АЗ работает с одним ИМ до восстановления отказавшего ИМ.

В рассмотренной дисциплине ***Д7*** время между соседними контролями модулей (цикл контроля) составляет 2 такта. На контроль избыточных модулей нерационально тратится один такт, т. е. половина длительности цикла контроля. Более экономичной организацией Д7 является построение АЗ в составе двух ИМ и двух групп но 4 основных модуля в группе. В этом случае цикл контроля составляет А7 = 3 такта, где 2 такта затрачивается на контроль модулей групп, а один такт – на контроль ИМ. Такая длительность цикла контроля может быть достигнута в ИС с общей оперативной памятью. Если же каждый модуль обработки информации содержит собственную память, то при тактах постоянной длительности потребуется за счет подготовки к параллельному счету удлинить цикл контроля до значения А7 = 5 тактов.

Таким образом, в данной дисциплине цикл контроля нечетных основных ВМ (1 и 3) составляет один или два (при двух

группах) такта АЗ, а четных (2 и 4)–два или четыре такта соответственно, т. е. $A_{1,3}(1)=1$; $A_{1,3}(2)=2$; $A_{2,4}(1)=2$; $A_{2,4}(2)=4$. В свою очередь, параметр $b = 1$, а количество тактов АЗ, затрачиваемых на восстановление вычислительного процесса: $x_{y1,3}(1)=2$; $x_{y1,3}(2)\leq 3$; $x_{y2,4}(1)\leq 3$; $x_{y2,4}(2)\leq 4$.

Следовательно, модульная дисциплина АЗ даже с малой избыточностью обеспечивает существенно меньшие затраты числа тактов АЗ, чем разветвленные базовые дисциплины. Вместе с тем, модульные дисциплины менее универсальны, чем базовые.

III.3.8. Эффективность применения системы адаптивной отказоустойчивости (активной защиты)

Эффективность активной защиты оценивается с помощью показателя вероятности успешной адаптации информационной системы к отказам, сбоям, программным ошибкам. Успешная адаптация будет в том случае, если в результате действий, предусмотренных алгоритмами АЗ, длительность существования указанных неисправностей меньше или равна допустимой величине. Здесь под допустимой величиной подразумевается допустимое время обнаружения и устранения неисправности в системе. Поскольку время устранения определяется для каждой дисциплины АЗ рестартом на заданное количество тактов, то достаточно ограничиться сравнением длительности обнаружения неисправности с допустимым временем обнаружения.

Оценка эффективности адаптивной отказоустойчивости зависит от следующих основных условий организации активной защиты:

1. Задачи обработки информации разделены на равные части (такты) τ, причем длительность такта много меньше длительности задачи.

2. Задачи обработки информации разделены на такты случайной длительности с равными средними значениями, причем длительность такта много меньше длительности задачи.

3.Длительность решения задачи υ_1 и длительность паузы между задачами υ_2 – соизмеримые случайные величины. Задачи кратковременные (например, задачи ввода – вывода информации, передачи данных и т.д.), поэтому их разделение на такты активной защиты нецелесообразно. Такой класс задач АЗ обозначим как активная защита интерфейсных модулей.

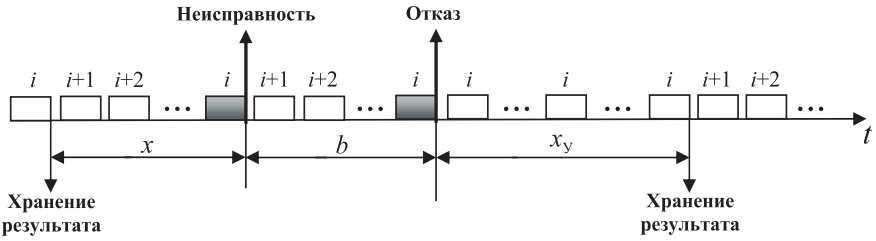
4. Задачи решаются через случайные интервалы времени υ_2, однако длительности решения задач υ_1 много больше длительности пауз, т.е. $\upsilon_1 >> \upsilon_2$. Это позволяет разделить задачу на такты АЗ (например, для общности – такты случайной длительности). Кроме того, учитывается, что допустимое время перерыва в работе системы – случайная величина υ_3, соизмеримая со временем решения задачи υ_1. Отсюда также следует, что $\upsilon_3 >> \upsilon_2$.

III.3.8.1. Задачи обработки информации разделены на равные части (такты)

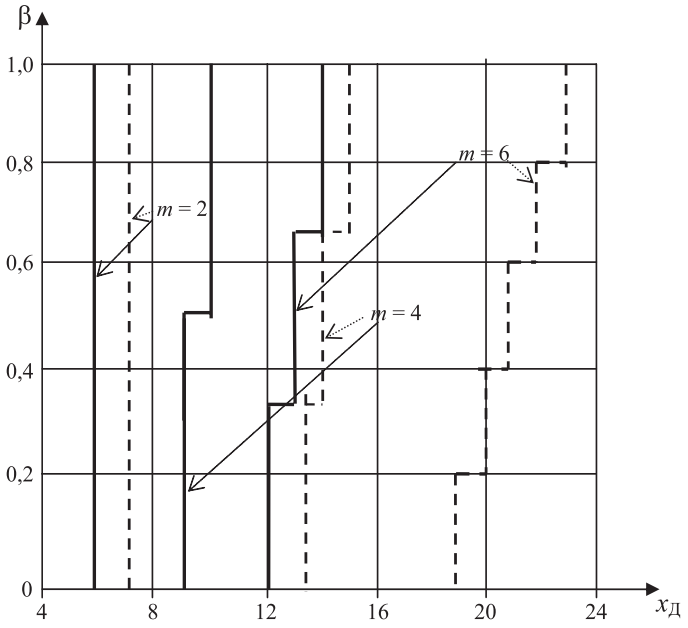
Задача, выполняемая ВМ, разделена на одинаковые такты постоянной длительности τ. Требуется определить вероятность успешной адаптации системы к отказам при условии, что отсутствуют одновременные отказы или сбои проверяемого в текущем такте рабочего и контролируемого ВМ.

Временная диаграмма работы системы при рассматриваемой организации АЗ показана на рис.III.3.10а

Случайное количество тактов существования скрытого отказа определяется в виде $v=x+b$ при $0<x\leq A-1$, где x – случайное количество тактов, прошедших от момента возникновения необнаруженного встроенным контролем отказа основного ВМ до подключения к нему контролирующего ВМ (на рис. III.3.10 а



а)



б)

Рис. III.3.10. Постоянные такты активной защиты ИС: а – временная диаграмма работы системы; б – зависимости вероятности успешной адаптации ИС к отказам модулей от допустимого количества тактов $x_д$ защиты и от числа m основных ВМ

показаны незатемненными прямоугольниками такты работы основного ВМ, в которых он либо исправен, либо неисправен, но неисправность не обнаружена). При подключении к основному ВМ контролирующего модуля обнаружена неисправность

(затемненный прямоугольник). Эта неисправность возникла в предыдущем цикле контроля и с равной вероятностью могла произойти на i+1, i+2, …, i+A-1 такте АЗ, где A – количество тактов в цикле. Затем в течение постоянного количества тактов b принимается решение по обнаружению скрытого отказа ВМ с помощью АЗ. Причем, b=m($2m$) при реализации дисциплин ***Д1, Д2, Д3***, b = INT[(m+1)/2] или b = INT[(2m+1)/2] при реализации ***Д5***, b=2 при реализации ***Д4*** или ***Д6.*** При реализации модульной дисциплины ***Д7*** b=1.

Вследствие однотипности ВМ может быть принята в течение A-1 тактов равновероятной возможность отказа любого из ВМ и одинаковыми законы распределения дискретной случайной величины x относительно каждого основного ВМ.

Величина x может быть выражена через дискретную случайную величину v и постоянное число b, т.е. x=v–b. Тогда функция распределения $F(x)$ случайного количества тактов до первого внешнего контроля отказавшего ВМ определяется вероятностью

$$F(x)=P\{x<X\}, \text{ где } X>x_{Д}–b–x_{y},$$

где: $x_{Д} = \tau_{Д}/\tau$ – допустимое количество тактов перерыва в работе системы, x_{y} – постоянное количество тактов, необходимых для устранения отказа основного ВМ с помощью исправного контролирующего модуля, который теперь замещает основной модуль. Устранение функционального отказа основного ВМ осуществляется путем решения исправным модулем участка задачи от момента последнего благоприятного внешнего контроля до принятия решения об отказе этого основного ВМ (рис. III.3.10 а).

Вследствие равновероятной возможности отказа любого из ВМ, которые не защищены в данном такте, можно полагать, что целочисленная случайная величина x равномерно распределена в диапазоне чисел 1, 2, …, A-1. Отсюда плотность распределе-

ния количества тактов существования скрытого отказа ИС определяется с помощью выражения:

$$f(x)=\sum_{i=1}^{A-1}\frac{\delta(x-i)}{A-1}, \qquad (3.3)$$

где $\delta(x)$ есть дельта функция от параметра x.

Таким образом, вероятность успешной адаптации к отказам ИС с АЗ при постоянной одинаковой длительности тактов равна

$$\beta=\begin{cases}0, & при\cdot x_{Д}\le b+x_{У},\\ \dfrac{x_{Д}-b-x_{У}}{A-1}, & при\cdot b+x_{У}<x_{Д}<b+x_{У}+A-1.\\ 1, & при\cdot x_{Д}\ge b+x_{У}+A-1\end{cases} \qquad (3.4)$$

Графики зависимостей вероятности β от допустимого количества $x_{Д}$ постоянных тактов АЗ показаны на рис. III.3.10 б для дисциплин ***Д1, Д2, Д3*** (пунктирные линии) и дисциплины ***Д5*** (сплошные линии). Из этих графиков следует, что для достижения заданной вероятности $\beta_{З}=1$ нужно рассчитать длительность такта по формуле $\tau\le\tau_{Д}/4m$. Например, при $\tau_{Д}=1$с и $m=10$ длительность такта должна быть меньше или равна 25мс.

Переназначение модулей согласно дисциплине ***Д5*** позволяет сократить время адаптации системы к отказу до $2m$-3 тактов. Это означает, что для достижения вероятности $\beta_{З}=1$ нужно меньше на указанное количество тактов для принятия решения, чем это должно быть при применении дисциплин ***Д1, Д2, Д3.***

Сочетание переназначения модулей с повторным счетом в случае несовпадения результатов (дисциплина ***Д6***) позволяет сократить время адаптации системы к отказу до m+4 . Так, при $\tau_{Д}=1$ с и $m=10$ длительность такта может составлять 70 мс.

Применение дисциплины ***Д7*** обеспечивает наименьшее число тактов адаптации системы к отказу. Наиболее развитые дис-

циплины ***Д7*** и ***Д6*** позволяют значительно оперативнее обнаруживать отказавший ВМ, чем простейшие дисциплины ***Д1, Д2, Д3.*** Однако при этом не следует забывать, что для реализации развитых дисциплин АЗ приходится усложнять средства управления ими.

III.3.8.2. Задачи обработки информации разделены на такты случайной длительности

Задача, выполняемая ВМ, разделена на такты случайной длительности со средним значением τ. Длительность задачи много больше длительности такта. Предполагается, что отсутствуют одновременные отказы или сбои проверяемого в текущем такте рабочего и контролируемого ВМ. Длительность такта определяется длительностью выполнения в пределах такта группы функционально законченных программных модулей. Поскольку все ВМ, на которых выполняются программные модули, однотипны, то порядок распределения программных модулей по тактам работы ВМ является общим для всех ВМ. Это позволяет принять одинаковыми распределения $F_{\upsilon}(t)$ длительности тактов для всех ВМ.

Требуется установить вероятность успешной адаптации системы к отказам:

$$\beta = P\{\nu \le \tau_{Д} - t_{у}\} = \int_{0}^{\tau_{Д} - t_{у}} f_{\nu}(t)dt, \qquad (3.5)$$

где $f_{\nu}(t)$ – функция плотности времени ν существования скрытого отказа в ИС.

Для нахождения функции плотности $f_{\nu}(t)$ и вероятности успешной адаптации к отказам β в целом используются следующие параметры:

• функции распределения и характеристики временных интервалов защиты – длительности тактов, допустимого времени

перерыва в работе системы и времени устранения обнаруженного отказа ($\tau_Д$ и $t_у$ соответственно);

• количество основных m и избыточных k модулей в системе;

• параметры принятой дисциплины АЗ: $A,b,t,x_у=t_у/\tau$.

Время подключения контролирующего ВМ к очередному основному слагается из случайной длительности такта υ и времени ожидания ψ от момента завершения параллельной работы с предыдущим ВМ до момента начала следующего такта работы очередного ВМ (рис. III.3.7 б). Причем, $\psi \le \upsilon$ и во время ожидания осуществляется загрузка памяти контролирующего ВМ командами и операндами очередного основного модуля.

Для нахождения функции плотности времени v предварительно установим функцию плотности суммарного времени $\psi+\upsilon$. В изображении Лапласа она имеет вид

$$f_\varsigma(s) = \phi_\psi(s) * f_\upsilon(s),$$

где $\varphi_\psi(S)$ – изображение плотности распределения времени ожидания ψ, а $f_\upsilon(s)$ – изображение плотности распределения длительности такта АЗ.

Пусть от момента возникновения скрытого отказа ВМ до момента подключения к нему контролирующего ВМ прошло x тактов. Тогда условная вероятность того, что $x<X$, где $X = 0, 1, \ldots, A, \ldots$, может быть найдена с помощью соответствующего преобразования Лапласа

$$f_x(s)=[f_\xi(s)]^x.$$

В рассматриваемых условиях организации АЗ x – дискретная случайная величина с функцией плотности $f(x)$, определяемой выражением (3.3).

Общая длительность существования отказа до его обнаружения представляет собой сумму времени $x(\upsilon+\psi)$ и времени $b(\upsilon+\psi)$ от момента обнаружения факта неисправности до локализации отказавшего ВМ в соответствии с выбранной дисциплиной АЗ.

Функция плотности случайной величины $x(\upsilon+\psi)=\theta$ в изображении Лапласа представляется следующим образом по формуле полной вероятности:

$$f_\theta(s) = \sum_{i=1}^{A-1} \frac{1}{A-1} (f_\xi(s))^i$$

Функция плотности случайной величины $(x+b)\cdot(\upsilon+\psi)$ в изображении Лапласа вычисляется в виде

$$f_\nu(s) = f_\theta(s) * (f_\upsilon(s))^b = \frac{1}{A-1} \sum_{i=1}^{A-1} (f_\xi(s))^{i+b} \qquad (3.6)$$

Следующий шаг в определении вероятности успешной адаптации к отказам системы с рассматриваемой организацией активной защиты состоит в том, чтобы в полученном выше выражении раскрыть функцию $f_\xi(s)$, которая в изображении Лапласа есть функция плотности суммы длительности такта и времени задержки в подключении контролирующего ВМ к основному в пределах такта ($\xi=\upsilon+\psi\le 2\upsilon$).

Ориентируясь на экспериментальные данные [28], примем распределения случайных длительностей тактов υ в виде распределения Эрланга a-го порядка с функцией плотности $f_\upsilon(t) = \frac{\rho(\rho\cdot t)^a}{a!} e^{-\rho\cdot t}$, которые в изображении Лапласа имеют следующий вид:

$$f_\upsilon(s) = \left(\frac{\rho}{\rho+s}\right)^{a+1},$$

где ρ – параметр распределения Эрланга (количество событий в единицу времени).

Согласно работе [50] функция плотности времени ожидания ψ (в нашем случае время подключения контролирующего к основному) в изображении Лапласа имеет следующий вид:

$$\phi_{\psi}(s) = \frac{\rho}{(a+1)s}[1-(\frac{\rho}{\rho+s})^{a+1}]$$

Следовательно, в формуле (3.6) функция плотности $f_{\xi}(s)$ равна

$$f_{\xi}(s) = f_{\upsilon}(s) * \phi_{\psi}(s) = (\frac{\rho}{\rho+s})^{a+1} * \frac{\rho}{(a+1)s}[1-(\frac{\rho}{\rho+s})^{a+1}]$$

Подставляя данное выражение в формулу (3.4), находим

$$f_{\nu}(s) = \frac{1}{A-1}\sum_{i=1}^{A-1}\{(\frac{\rho}{\rho+s})^{a+1}\frac{\rho}{(a+1)s}[1-(\frac{\rho}{\rho+s})^{a+1}]\}^{i+b} \qquad (3.7)$$

Переходя от изображения к оригиналу при постоянной величине допустимого времени перерыва в работе и воспользовавшись формулой (3.3), определяем вероятность успешной адаптации системы с АЗ к отказам

$$\beta = 1-\frac{e^{-(a+1)\cdot x_{Д}^{*}}}{A-1}\sum_{i=1}^{A-1}(\frac{1}{a+1})^{i+b}\sum_{|\bar{n}|=i+b}\frac{(i+b)!}{\bar{n}!}\cdot\sum_{k=0}^{\eta}\frac{((a+1)x_{Д}^{*})^{k}}{k!}, \quad (3.8)$$

где $\bar{n}! = n_0!\, n_1!\ldots n_a!$; $|\bar{n}| = n_0 + n_1 + \ldots + n_a$; $x_{Д}^{*} = \tau_{Д}^{*}/\tau$; $t_{Д}^{*} = \tau_{Д} - t_{у}$;

$$\eta = (a+2)(i+b) + \sum_{j=1}^{a} jn_j + 1$$

В частном случае a=0 (экспоненциальное распределение длительности такта) имеет место следующее выражение вероятности успешной адаптации системы к отказам

$$\beta = 1-\frac{e^{-\cdot x_{Д}^{*}}}{A-1}\sum_{i=1}^{A-1}\sum_{k=0}^{2(i+b)+1}\frac{(x_{Д}^{*})^{k}}{k!},$$

поскольку в этом случае $\bar{n}! = 1$, а $|\bar{n}| = 0$.

С помощью выражения (3.8) исследуем зависимости вероятности успешной адаптации системы с активной защитой от

допустимого количества тактов перерыва в работе, числа m основных модулей и от дисциплины АЗ. Исходные данные по основным дисциплинам АЗ приведены в табл. III.3.7.

Таблица III.3.7

Типы дисциплин	A	b	$X^*_д = x_д - x_у$
Д1…Д3	m	m	$t^*_д/\tau - 2m$
Д4	m	2	$\tau^*_д/\tau - (m+2)$
Д5	INT[(m+1)/2]	INT[(m+1)/2]	$\tau^*_д/\tau - (m+1)$
Д6	INT[(m+1)/2]	2	$\tau^*_д/\tau$ – INT((m+5)/2)
Д7	2 (при m=4)	1	2 (при m=4)

Исследования зависимостей вероятности от допустимого количества тактов АЗ при различных порядках распределения Эрланга длительности такта $\beta = f(x_д)$ показали, что начиная со второго порядка и выше результаты этих зависимостей практически совпадают. Они заметно отличаются от полученных расчетов при экспоненциальном распределении длительности такта. На рис. III.3.11 показаны зависимости $\beta = f(x_д)$ при $a \geq 2$ применительно к ключевым базовым дисциплинам ***Д4*** (сплошные линии) и ***Д6*** (пунктирные линии).

Эти графики подтверждают полученные ранее результаты для постоянных тактов работы системы и свидетельствуют о том, что наибольшей скоростью адаптации к отказу ВМ обладают дисциплины ***Д6*** и ***Д7***. Эти дисциплины примерно на 3-4 такта быстрее реагируют на событие возникновения отказа ВМ по сравнению с дисциплиной ***Д4***, на 4-5 тактов быстрее, чем ***Д5*** и в три раза быстрее, чем дисциплины ***Д1, Д2, Д3***. Преимущества отмеченных дисциплин повышаются по мере увеличения количества основных вычислительных модулей.

Вместе с тем, АЗ со случайной длительностью тактов значительно инерционнее, чем АЗ с тактами постоянной длительности. Так, даже при минимальном для АЗ количестве основных модулей m=2 время обнаружения и устранения отказа ВМ увеличивается в 1,5-2 раза. Поскольку для многих архитектур ИС и вычислительных процессов в них не представляется возможным организовать АЗ с постоянными тактами, то целесообразно изыскивать дополнительные возможности в повышении скорости адаптации ИС с АЗ к отказам составных ВМ.

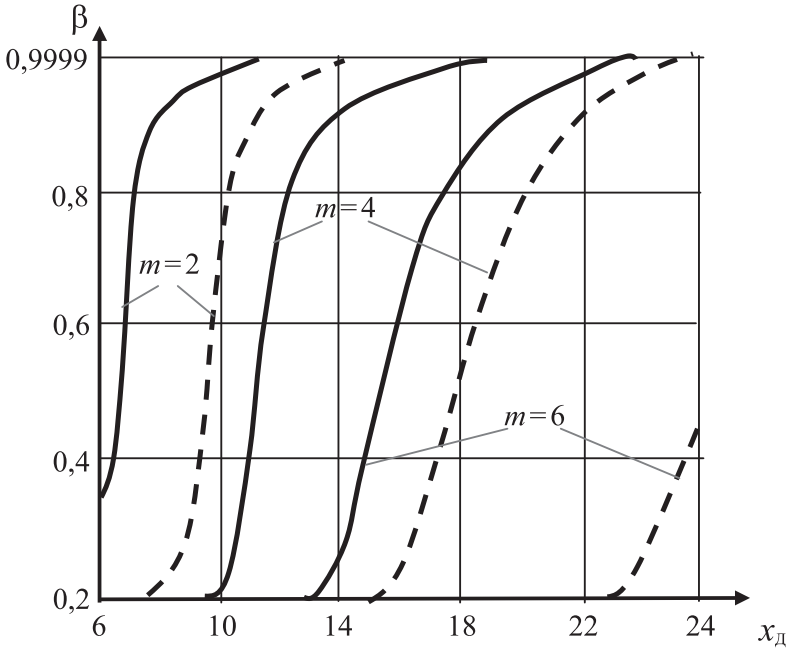


Рис. III.3.11. Зависимости вероятности успешной адаптации ИС к отказам со случайными тактами АЗ в зависимости от допустимого количества $x_д$ тактов защиты и от числа m основных ВМ

Такая возможность, например, имеет место при наличии встроенного контроля основных ВМ, с помощью которого в системах с АЗ возможно своевременно обнаружить и устра-

нить отказ ВМ с вероятностью $\gamma_{АЗ}=\gamma+\beta-\gamma\cdot\beta$, где γ – вероятность правильного и своевременного обнаружения и устранения отказа ВМ в системе без АЗ (см. формулу 2.7). При этих условиях реально снизить требования к эффективности АЗ, за счет чего ожидаемый эффект от нее наступит существенно быстрее (см. п. III.3.9).

III.3.8.3. Активная защита интерфейсных модулей

Рассматриваются ситуации, при которых длительности пауз больше длительностей решения задач. Эти соотношения между величинами υ_1 и υ_2 ($\upsilon_1 \le \upsilon_2$) характерны для интерфейсных каналов ИС (модули обмена и передачи информации). На рис. III.3.12 проиллюстрированы временные интервалы работы ИС с активной защитой при соизмеримых длительностях υ_1 и υ_2 и при условии, что интервалы времени υ_1 относительно длительности такта защиты невелики. Последнее обстоятельство означает, что возможно применение АЗ с естественными тактами защиты, которые равны времени решения коротких задач.

В течение времени υ_1 избыточный интерфейсный модуль работает в паре с i-м основным модулем. При благоприятных результатах сравнения избыточный модуль подключается к очередному (i+1)-му основному модулю. При этом возможно наступление одного из двух событий: избыточный модуль подключен к очередному в интервале времени решения им задачи (гипотеза H_1); избыточный модуль подключен к очередному в паузе между задачами (гипотеза H_2). Первое событие показано на рис. III.3.12 а. При этом избыточный интерфейсный модуль ожидает начала совместной работы с (i+1)-м основным модулем в течение случайного времени $\psi_1+\upsilon_2$, где $\psi_{1\le}\upsilon_1$ – случайный интервал времени от момента подключения избыточного модуля до завершения решения

задачи очередным основным модулем. Второе событие показано на рис. III.3.12 б. При этом время ожидания начала параллельной работы модулей равно случайной длительности $\psi_{2\leq}\upsilon_2$ от момента подключения избыточного модуля до завершения текущей паузы в работе (i+1)-го основного модуля.

Случайное время ζ реконфигурации системы включает в себя следующие составляющие (рис. III.3.12):

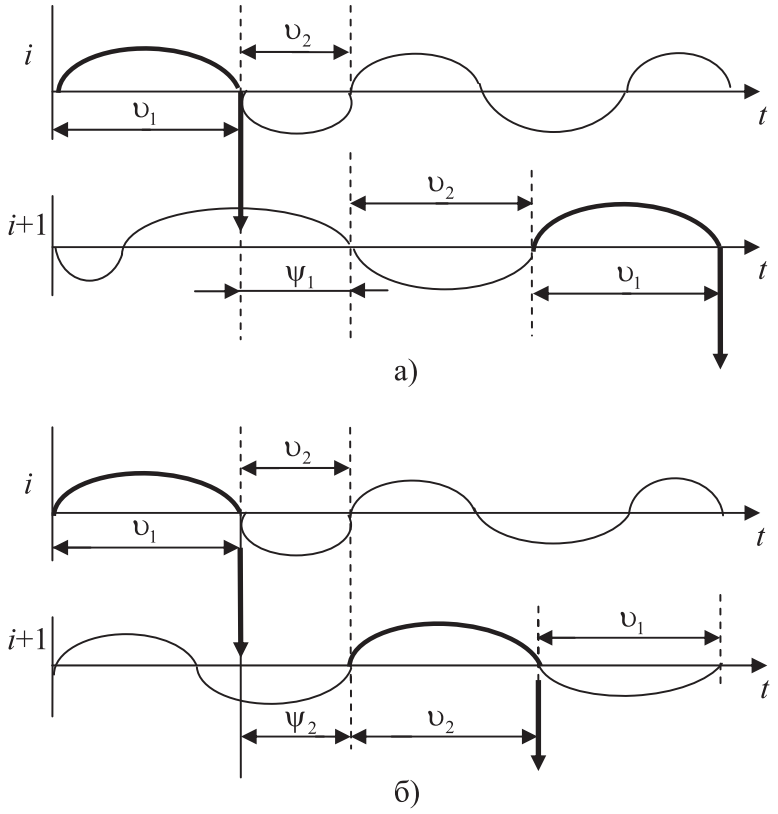


Рис. III.3.12. Временная диаграмма подключений контролирующего ВМ к очередному основному ВМ при соизмеримых и коротких интервалах решения задач и пауз между ними: а – случай подключения контролирующего ВМ в интервале решения задачи; б – случай подключения контролирующего ВМ в паузе между задачами.

$$\zeta = I_{H_1}(\psi_2 + \upsilon_1) + I_{H_2}(\psi_1 + \upsilon_1 + \upsilon_2),$$

где I_{H_1}, I_{H_2} – индикаторы гипотез H_1 и H_2, принимающие единичные значения при реализации этих гипотез или нулевые значения в противоположных случаях. Вероятности этих гипотез при условии распределений Эрланга случайных величин с параметрами соответственно ρ_1 и ρ_2 и порядками распределений a_1 и a_2 вычисляются следующим образом:

$$P_{H_1} = \frac{\rho_2(a_1+1)}{\rho_2(a_1+1)+\rho_1(a_2+1)};\ P_{H_2} = \frac{\rho_1(a_2+1)}{\rho_2(a_1+1)+\rho_1(a_2+1)}.$$

Функции плотности $\varphi_i(s)$ в изображении Лапласа случайных величин $\psi_i(i=1, 2)$ согласно [19] находятся в виде:

$$\phi_i(s) = \frac{\rho_i}{a_i+1}\sum_{j=0}^{a_i}\frac{\rho_i^j}{(\rho_i+s)^{j+1}},\ i=1,2.$$

Таким образом, функция плотности $f_\zeta(s)$ случайной величины ζ в изображении Лапласа имеет вид:

$$f_\zeta(s) = P_{H_1}\cdot\phi_1(s) * f_2(s) + P_{H_2}\cdot\phi_2(s) * f_1(s) * f_2(s),$$

где $f_i(s) = \left(\frac{\rho_i}{\rho_i+s}\right)^{a_i+1},\ i=1, 2.$

Подставляя данные выражения в формулу (3.6) и используя биноминальное разложение выражения $(f_\zeta(s))^{\gamma+b}$, находим изображение Лапласа функции плотности $f_v(s)$ времени скрытого отказа. После обратного преобразования этой функции и последующей подстановки ее в формулу (3.5) находим формульное выражение вероятности успешной адаптации системы к отказу интерфейсного модуля:

$$\beta = \frac{1}{A-1}\sum_{r=1}^{A-1}\sum_{|k|=r+b}\frac{(r+b)!}{\bar{k}!}c^{r+b}\rho_2^{\omega_1}\rho_1^{\omega_2}\cdot$$
$$\cdot\{\sum_{i=1}^{\omega_2}\frac{(\omega_4+i-2)!(\rho_1-\rho_2)^{-\omega_4}(\rho_2-\rho_1)^{-i+1}}{(\omega_4-1)!(i-1)!}\cdot$$
$$\cdot(\rho_2^{-\omega_4+i-1}--e^{-\rho_2\cdot\tau_{Д}^{*}}\sum_{z=0}^{\omega_2-i}\frac{\tau_{Д}^{*(\omega_2-i-z)}}{\rho_2^{z+1}(\omega_2-i-z)!})+$$
$$+\sum_{i=1}^{\omega_4}\frac{(\omega_4+i-2)!(\rho_2-\rho_1)^{-\omega_2}(\rho_1-\rho_2)^{-i+1}}{(\omega_2-1)!(i-1)!}\cdot \qquad (3.9)$$
$$\cdot(\rho_1^{-\omega_4+i-1}-e^{-\rho_2\cdot\tau_{L}^{*}}\sum_{z=0}^{\omega_4-i}\frac{\tau_{L}^{*(\omega_4-i-z)}}{\rho_1^{z+1}(\omega_4-i-z)!})\}$$

где $c=\dfrac{\rho_1\rho_2}{\rho_1(a_2+1)+\rho_2(a_1+1)}$; $\bar{k}!=k_0!k_1!...k_{a_1+a_2+1}!$

(k_j – целые положительные числа);

$$\omega_1=\sum_{j=0}^{a_1}jk_j+(a_2+1)(r+b)-\sum_{j=0}^{a_2}k_j(a_2+1);$$

$$\omega_2=\sum_{j=0}^{a_2}jk_j+(a_2+1)(r+b)-a_2\sum_{j=0}^{a_2}k_j;$$

$$\omega_3=(a_1-a_2)(r+b)+\sum_{j=a_2+1}^{a_1+a_2+1}jk_j+(a_2-1)\sum_{j=0}^{a_2}k_j;$$

$$\omega_4=(a_1-a_2+1)(r+b)+\sum_{j=a_2+1}^{a_1+a_2+1}jk_j+a_2\sum_{j=0}^{a_2}k_j.$$

Используя формулу (3.9) можно установить такое соотношение между допустимым временем перерыва в работе системы и длительностью такта АЗ, при котором активная защита ин-

терфейсных модулей обеспечивает вероятность успешной адаптации интерфейсных модулей на высоком желаемом уровне (например, $\beta \geq 0{,}9999$).

На рис. III.3.13 приведены графические зависимости требуемого количества тактов $x^*_{Д}=\tau^*_{Д}/\tau$ для обеспечения желаемого значения вероятности β от отношения средней длительности пауз между заявками на обмен или передачу информации $\bar{\tau}_{П} = 1/\rho_2$ (ρ_2 – параметр потока пауз между заявками) к средней длительности такта АЗ и от количества m интерфейсных модулей применительно к эффективной дисциплине ***Д6*** и для большинства распределений Эрланга ($a_1, a_2 \geq 2$).

Данные графики пронормированы относительно количества тактов АЗ при предельной загрузке (отсутствие пауз между заявками – $\tau_{П}=0$). Они свидетельствуют о линейном характере связи между требуемым количеством тактов АЗ и количеством тактов, которые можно разместить в интервале между заявками. Основной вывод, который следует из этих графических зависимостей состоит в том, что требуемая длительность такта для достижения желаемого уровня адаптации к отказам определяется главным образом длительностями пауз между заявками, и в меньшей мере зависит от количества избыточных интерфейсных модулей.

Проведенные теоретические исследования на основе формулы (3.9) позволяют сделать следующие выводы:

• При выборе необходимого количества тактов АЗ для достижения желаемого значения вероятности β успешной адаптации системы к отказам интерфейсных модулей можно руководствоваться следующим неравенством:

$$t^*_{Д} \geq \mathrm{INT}\left[\frac{(3m+23)(\bar{\tau}_{З}+\bar{\tau}_{П})}{2}\right], \qquad (3.10)$$

где $\bar{\tau}_{З}, \bar{\tau}_{П}$ – средние длительности обслуживания заявок и пауз между ними соответственно;

• Поскольку средняя длительность обслуживания заявок $\bar{\tau}_З$ обмена и/или передачи информации в современных ИС малая величина соизмеримая с длительностью такта, назначаемого при решении больших задач обработки информации, то невозможно разделять этот временной интервал на такты. Более того, в интерфейсных каналах ИС среднюю длительность обслуживания заявок следует рассматривать в качестве длительности такта АЗ, т.е. $\bar{\tau}_З = 1/\rho_1 = \tau$;

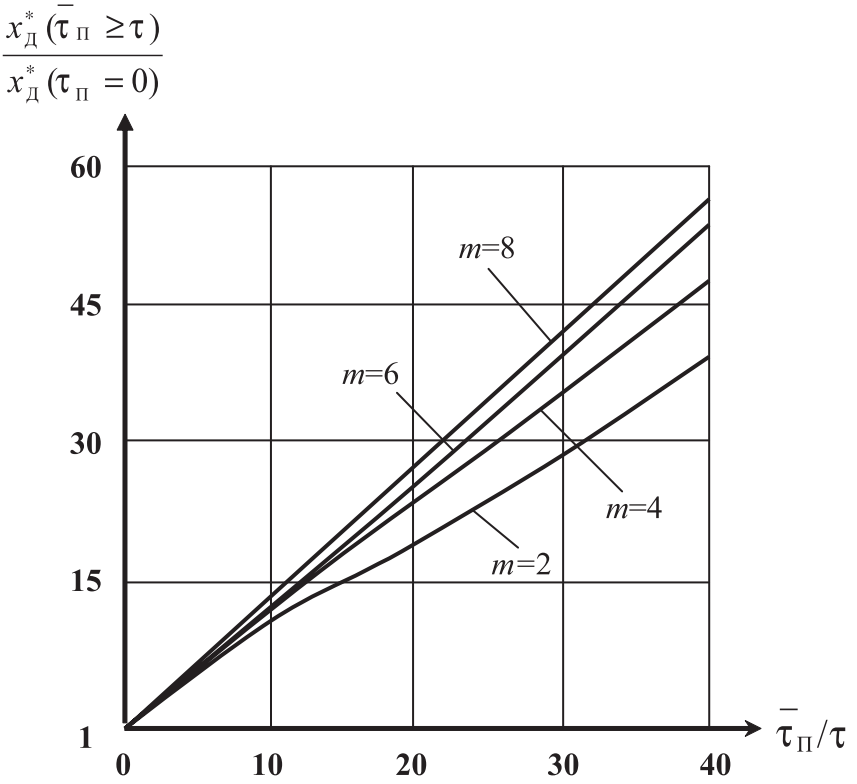


Рис. III.3.13. Зависимости требуемого количества тактов АЗ интерфейсных модулей от отношения средней длительности пауз между заявками к средней длительности такта и от количества *m* ***интерфейсных модулей***

• При значениях $\tau^*_{д}$, исчисляемых несколькими секундами (что характерно для информационно – управляющих ИС реального времени) неравенство (3.10) может быть выполнено и активная защита интерфейсных модулей даст желаемый эффект даже при сравнительно больших длительностях пауз ($\overline{\tau}_{П} > \overline{\tau}_{З}$);

• В ИС с малой информационной нагрузкой, когда интерфейсные модули длительное время простаивают в ожидании заявок на обслуживание, что характеризуется соотношением $\overline{\tau}_{П} / \overline{\tau}_{З} > 100$, *применение АЗ интерфейсных модулей не оправдано* по следующим причинам:

– не удается достичь желаемого уровня эффективности отказоустойчивости;

– большие интервалы времени ожидания заявок представляют значительный дополнительный естественный резерв времени, в течение которого можно гарантированно определить состояние работоспособности интерфейсных модулей с помощью известных способов технической диагностики и даже устранить отдельные отказы.

Таким образом, при длительных простоях вычислительных средств, обусловленных отсутствием заявок на обслуживание, применение активной защиты не всегда целесообразно.

III.3.8.4. Активная защита со случайным временем перерыва в работе информационной системы

В информационных системах реального времени ВМ работают в условиях большой информационной нагрузки, когда длительности решения задач υ_1 много больше длительности пауз υ_2 между ними, что позволяет разделить задачу на такты АЗ (например, постоянной длительности τ). Такая организация типична для многих информационных систем. Для общности примем случайным время υ_3 допустимого перерыва в работе системы.

При оценке эффективности АЗ в указанных условиях организации системы с активной защитой приняты следующие предпосылки:

Активная защита реализована с помощью любой из базовых дисциплин ***Д1, Д2, …, Д6***;

Длительности решения задач и пауз между ними распределены по закону Эрланга соответственно порядка a_1 или a_2 с параметрами ρ_1 или ρ_2;

Допустимое время перерыва в работе системы υ_3 описывается экспоненциальным распределением с интенсивностью ρ_3;

Вероятности одновременного возникновения отказов параллельно работающих ВМ ничтожно малы.

Задача заключается в определении вероятности β успешной адаптации системы к сбоям и отказам ВМ в зависимости от количества тактов $x_Д$ АЗ и количества основных ВМ. Временная диаграмма работы такой системы показана на рис. III.3.14.

Для решения задачи предварительно установим формульное выражение вероятности успешной адаптации при случайном допустимом времени перерыва в работе.

Если допустимое время υ_3 описывается экспоненциальным распределением с интенсивностью ρ_3, то по формуле полной вероятности

$$\beta = \int_{t_у}^{\infty} P\{\nu \le \tau_Д - t_у\}\rho_3 e^{-\rho_3 \cdot t} dt = e^{-\rho_3 \cdot t_у} \int_0^{\infty} f_\nu(t) e^{-\rho_3 \cdot t} dt,$$

где $t_у$ – неслучайное время устранения отказа ИС с АЗ.

При решении подобных задач удобно использовать преобразование Лапласа $f_\nu(s)$ функции $f_\nu(t)$. Тогда вероятность успешной адаптации системы к отказам вычисляется следующим образом:

$$\beta = e^{-\rho_3 \cdot t_у} [f_\nu(s)]_{s=\rho_3} \qquad (3.11)$$

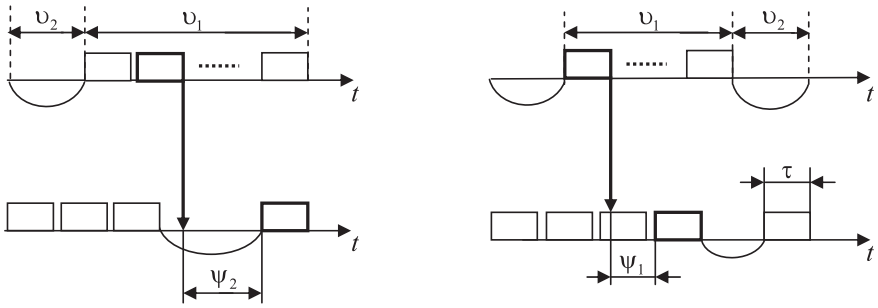


Рис. III.3.14.Времненная диаграмма реконфигурации ИС с АЗ с учетом случайных пауз между задачами: а – реконфигурация в интервале паузы между задачами; б – реконфигурация в интервале решения задачи

Чтобы определить функцию $f_v(s)$ следует предварительно установить распределение случайного времени ζ реконфигурации системы, которое включает в себя длительность такта АЗ и время ожидания начала параллельной работы контролирующего ВМ с очередным основным. В соответствие с рис. III.3.14 случайное время ζ представляет собой:

$$\zeta = I_{H_1}(\psi_1 + \tau) + I_{H_2}(\psi_2 + \tau),$$

где I_{H_1}, I_{H_2} – индикаторы гипотез H_1 и H_2, принимающие единичные значения при реализации этих гипотез или нулевые значения в противоположных случаях.

Гипотеза H_1 заключается в подключении контролирующего ВМ к очередному основному в интервале его работы, что вызывает ожидание начала параллельной работы в течение случайного времени $\psi_1 \le \tau$ (на рис. III.3.14 б последующая за этим интервалом времени ожидания параллельная работа показана прямоугольником с утолщенными краями). Гипотеза H_2 предполагает подключение контролирующего ВМ к основному в паузе между работами и ожидание начала параллельной работы в течение случайного времени $\psi_2 \le \upsilon_2$ (на рис. III.3.14 а последующая за этим интервалом времени ожидания параллельная работа показана прямоугольником с утол-

щенными краями). Вероятность этих гипотез при условии эргодичности случайного процесса вычисляется в следующем виде:

$$P_{H_1} = \frac{\rho_2(a_1+1)}{\rho_2(a_1+1)+\rho_1(a_2+1)};\ P_{H_2} = \frac{\rho_1(a_2+1)}{\rho_2(a_1+1)+\rho_1(a_2+1)}$$

Вероятности P_{H_1} и P_{H_2} образуют полную группу событий.

Случайный остаток ψ_1 такта постоянной длительности τ согласно работе [19] равномерно распределен на отрезке $[0, \tau]$ с интенсивностью $1/\tau$. Его функция плотности в изображении Лапласа имеет вид:

$$\varphi_1(s) = \frac{1}{\tau \cdot s}(1 - e^{-\tau \cdot s}).$$

Функция плотности остатка случайного времени ψ_2 паузы между заявками также на основании работы [19] определяется с помощью изображения Лапласа:

$$\varphi_2(s) = \frac{\rho_2}{(a_2+1)\cdot s}[1 - (\frac{\rho_2}{\rho_2 + s})^{a_2+1}].$$

Таким образом, имеет место следующая функция плотности случайного времени ζ в изображении Лапласа:

$$f_\zeta(s) = P_{H_1}\varphi_1(s) * e^{-\tau \cdot s} + P_{H_2}\varphi_2(s) * e^{-\tau \cdot s} =$$

$$= \frac{\rho_1\rho_2 e^{-\tau \cdot s}}{s[\rho_1(a_2+1)+\rho_2(a_1+1)]}[1 - (\frac{\rho_2}{\rho_2+s})^{a_2+1} + \frac{a_1+1}{\rho_1\tau}(1 - e^{-\tau \cdot s})]$$

Подставляя данное выражение в формулу (3.6), определяем изображение Лапласа $f_v(s)$ функции плотности времени существования скрытого отказа ИС, а затем с помощью формулы (3.11) для экспоненциального распределения с интенсивностью ρ_3 допустимого времени перерыва в работе системы можем найти конечное выражение вероятности успешной адаптации ИС

к отказам для рассматриваемой распространенной организации работы ИС с АЗ.

Вследствие громоздкости конечного выражения вначале ограничимся расчетной формулой вероятности β для типичного на практике условия малых значений отношения средних значений пауз и выполняемых задач $\bar{\tau}_{\text{П}} / \bar{\tau}_{\text{З}} << 1$, где $\bar{\tau}_{\text{П}} = \frac{a_2 + 1}{\rho_2}$, а $\bar{\tau}_{\text{З}} = \frac{a_1 + 1}{\rho_1}$. При этих условиях можно пренебречь малыми порядка $(\frac{1}{\rho_2 \bar{\tau}_{\text{Д}}})^{i+b}$ и записать расчетную формулу данной вероятности в следующем виде:

$$\beta \approx \frac{e^{-z}}{A-1} \frac{q^{b+1}(1-q^{A-1})}{1-q} \qquad (3.12)$$

где z – отношение времени $t_{\text{у}}$, необходимого для восстановления вычислительного процесса с последней контрольной точки, к среднему допустимому времени $\bar{\tau}_{\text{Д}}$ перерыва в работе системы. В общем случае $t_{\text{у}} = x_{\text{у}}(ДI) \cdot \bar{\zeta}$ (I=1, 2, 3, …, 7). Здесь $x_{\text{у}}(ДI)$ – количество тактов для устранения обнаруженных отказов при применении дисциплины АЗ ***ДI*** (см. табл. III.3.7), а $\bar{\zeta}$ – средний интервал времени между реконфигурациями ИС с АЗ. Таким образом, $z = x_{\text{у}}(ДI) \cdot \bar{\zeta} / \bar{\tau}_{\text{Д}}$. В частности, при применении дисциплины ***Д1*** $z = 2m \cdot \bar{\zeta} / \bar{\tau}_{\text{Д}}$, а при применении, например, дисциплины ***Д5*** находим по табл. III.3.7, что $z = 2m \cdot \bar{\zeta} / \bar{\tau}_{\text{Д}}$.

Параметр q в формуле (3.12) равен:

$$q = [f_{\zeta}(s)]_{s=\rho_3} = \frac{e^{-\rho_3 \tau}}{1 + \bar{\tau}_{\text{П}} / \bar{\tau}_{\text{З}}} \cdot [\frac{\bar{\tau}_{\text{Д}}}{\bar{\tau}_{\text{З}}} (1 - (1 + \frac{\rho_3}{\rho_2})^{-(a_1+1)}) - \frac{1 - e^{-\rho_3 \tau}}{\rho_3 \tau}].$$

Формула (3.12) обеспечивает высокую точность расчетов вероятности β в условиях большой информационной нагрузки ($\bar{\tau}_{\text{П}} << \bar{\tau}_{\text{З}}$). Графические зависимости вероятности β от ко-

личества m основных ВМ показаны на рис. III.3.15 для случая $\bar{\tau}_{\text{П}} / \bar{\tau}_{\text{З}} = 10^{-3}$.

Из графиков 1, 2 и 3 следует, что путем уменьшения отношения $\tau / \bar{\tau}_{\text{Д}}$ можно добиться высокого уровня вероятности β даже при большом количестве ($m > 10$) ВМ. Вместе с тем, при существенном повышении производительности ИС, когда уменьшается длительность решения задачи и величина $\bar{\tau}_{\text{З}}$ сверху приближается к $\bar{\tau}_{\text{Д}}$, нельзя добиться существенного повышения вероятности β, если при этом не отрегулировать должным образом длительность такта.

Исследования, проведенные в широком диапазоне исходных данных, показали приближенно линейную зависим ость вероятности β от количества m основных ВМ для малых параметров $\tau / \bar{\tau}_{\text{Д}} < 10^{-3}$ и $\bar{\tau}_{\text{П}} / \bar{\tau}_{\text{З}} < 10^{-3}$. Если учесть при этом наличие прямо пропорционального соотношения между длительностью цикла контроля A и количества основных модулей, то для решения практических задач можно вместо формулы (3.12) использовать следующее приближенное выражение:

$$\beta \approx (1 - b\eta - \frac{A}{2}\eta)^{-z} \qquad (3.13)$$

где $\eta = \dfrac{\bar{\tau}_{\text{П}} / \bar{\tau}_{\text{З}} + 1.5 \cdot \tau / \bar{\tau}_{\text{Д}}}{\bar{\tau}_{\text{П}} / \bar{\tau}_{\text{З}} + 1}$.

Погрешность вычислений по формуле (3.13) оценивается неравенством $\delta_{+} \le e^{-z} \dfrac{c}{A-1}\eta^2$, где

$$c = \frac{(b+A-1)(b+a)(2b+2A-1) - b(b+1)(2b+1) - 3 \cdot (2b+A)(A-1)}{6}.$$

Формула (3.13) может быть рекомендована при ограниченном цикле контроля A<10.

Следует отметить, что при одинаковых высоких порядках распределения Эрланга случайных длительностей задач и пауз между ними (υ_1 и υ_2) наблюдается незначительный рост расчетных значений вероятности β. Заметное колебание расчетных значений этой вероятности имеет место при больших взаимных отклонениях указанных порядков. Однако, при реализации этой организации ИС с АЗ во многих практических случаях можно ограничиться с приемлемой точностью экспоненциальными распределениями указанных случайных величин.

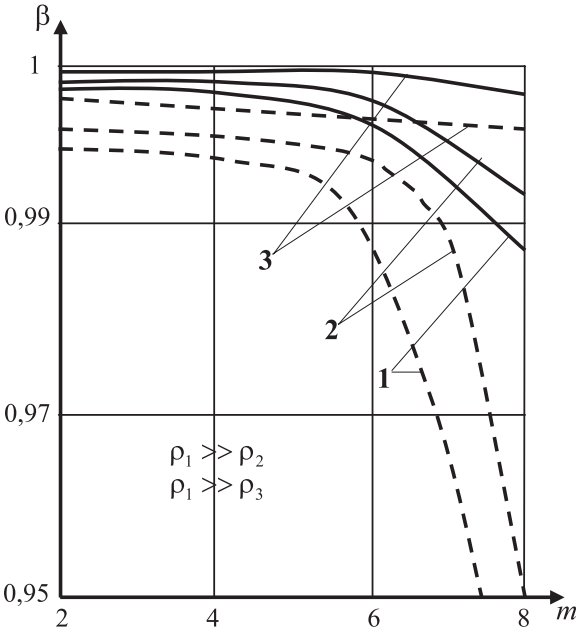


Рис. III.3.15. Зависимости вероятности успешной адаптации ИС к отказам в условиях большой информационной нагрузки и случайного времени допустимого перерыва в работе при:

$1 - \tau / \overline{\tau}_д = 10^{-2}$; $\overline{\tau}_д / \overline{\tau}_з = 10^{-2}$ (пунктирная линия);
$\overline{\tau}_д / \overline{\tau}_з = 10^{-2}$ (сплошная линия);
$2 - \tau / \overline{\tau}_д = 5 \cdot 10^{-3}$; $\overline{\tau}_д / \overline{\tau}_з = 10^{-2}$ (пунктирная линия);
$\overline{\tau}_д / \overline{\tau}_з = 10^{-3}$ (сплошная линия);
$3 - \tau / \overline{\tau}_д = 10^{-3}$; $\overline{\tau}_д / \overline{\tau}_з = 10^{-2}$ (пунктирная линия);
$\overline{\tau}_д / \overline{\tau}_з = 10^{-3}$ (сплошная линия);

III.3.9. Синтез активной защиты в информационных системах реального времени

Синтез активной защиты состоит в определении длительности такта АЗ. Эта задача обратная рассмотренным ранее задачам анализа вероятности успешной адаптации системы к неисправностям (отказам, сбоям, программным ошибкам). Она заключается в расчете средней длительности τ такта в зависимости от заданной вероятности β_3 и при известных всех других необходимых условиях:

- количество основных m и избыточных k ВМ;
- значение постоянного допустимого времени перерыва в работе системы $\tau_Д$;
- функции и параметры распределения случайной длительности такта $f_\upsilon(t)$ (если такты не постоянной длительности), времени решения задачи $f_{\upsilon_1}(t)$, времени паузы между решениями задач $f_{\upsilon_2}(t)$ и случайного допустимого времени перерыва в работе системы $f_{\upsilon_3}(t)$;
- условия организации активной защиты, связанные с классами решаемых задач (обработки информации или передачи или обмена данными) и привлекаемыми для этого вычислительными или интерфейсными модулями;
- параметры разработанных дисциплин активной защиты ***Д1, …, Д7***: A – количество тактов в цикле АЗ для данной дисциплины, b – количество тактов, затрачиваемых на принятие системой решения по обнаружению отказа ВМ, $t_У$ – постоянное время устранения отказа при реализации данной дисциплины АЗ.

Таким образом, требуется по исходному функционалу

$$\tau = \Phi\{\beta_3, f_\upsilon(t), f_{\upsilon_1}(t), f_{\upsilon_2}(t), \tau_Д(f_{\upsilon_3}(t)), A, b, t_У, m, k\}$$

аналитическим путем установить такое значение средней длительности такта АЗ, при котором выполняется условие $\beta \geq \beta_3$. Задача решается в следующем порядке:

1. Для конкретных условий организации работы системы с активной защитой и выбранном варианте дисциплины АЗ определяют аналитическое выражение функции плотности случайного времени v существования отказа ВМ до принятия решения по его обнаружению:

$$f_v(t) = f\{f_\upsilon(t), f_{\upsilon_1}(t), f_{\upsilon_2}(t), A, b, t_y, m, k\},$$

где все исходные условия и параметры известны, за исключением параметров функции плотности $f_v(t)$;

2. Определяют аналитическое выражение вероятности успешной адаптации системы к отказам

$$\beta = F\{f_v(t), \tau_Д(f_{\upsilon_3}(t), t_y\}$$

При постоянном допустимом времени перерыва в работе системы эта зависимость находится по формуле (3.5). Если же допустимое время перерыва в работе системы случайно и распределено, например, по экспоненциальному закону, то в этом случае вероятность успешной адаптации к отказам находят с помощью выражения (3.11);

2. В аналитическом выражении вероятности β подбираются такие значения неизвестных параметров функции плотности $f_v(t)$, чтобы при предварительно выбранной дисциплине АЗ выполнялось требуемое условие $\beta \geq \beta_3$. Если указанное условие выполняется, то решена предварительная задача синтеза – задача определения требуемого количества тактов АЗ, т.е задача нахождения значения $x_Д$;

3. По известному значению допустимого времени перерыва в работе системы $\tau_Д$ и установленному необходимому количеству тактов защиты решают окончательную задачу синтеза – находят среднюю (или постоянную) длительность такта АЗ. При этом учитывается то обстоятельство, что такты должны между собой

разнесены на отрезки времени $\Delta\tau<\tau$, необходимые для загрузки контролирующих ВМ командами и операндами задач, решаемых основными ВМ между соседними контрольными точками. Значение $\Delta\tau$ в современных ИС существенно меньше 50% от длительности такта АЗ и зависит от класса ИС – бортовые, стационарные, многопроцессорные, с конвейерной обработкой информации и т.д.), Средняя длительность такта вычисляется как

$$\tau^* = \frac{\tau_Д}{x_Д} - \Delta\tau = \tau - \Delta\tau = \tau(1 - k_\tau).$$

В общем случае необходимо найти максимальное значение τ, при котором еще обеспечивается заданный уровень вероятности β. С одной стороны, увеличение длительности такта АЗ позволяет уменьшить количество реконфигураций в системе и тем самым минимизировать потери в производительности в результате выполнения активной защиты. С другой стороны, с увеличением длительности такта уменьшается возможность разместить потребное количество тактов в интервале допустимого времени перерыва в работе ИС, что влечет за собой снижение эффективности системы обеспечения адаптивной отказоустойчивости. Изложенный порядок определения длительности такта обеспечивает решение этой противоречивой задачи.

Рассмотренная задача может быть расширена и в интересах синтеза архитектуры ИС с АЗ. Расширенный синтез отличается от ранее рассмотренного отсутствием исходной информации о параметрах *A, b* и $t_у$ дисциплины АЗ. Иными словами, разработчику дается возможность самостоятельно выбирать ту или иную организацию ИС с АЗ, руководствуясь тем, что должно выполняться условие $\beta\geq\beta_З$. Расширенный синтез активной защиты осуществляется путем подбора не только параметров функции плотности $f_v(t)$, но и параметров *A, b* и $t_у$, а, следовательно, дисциплины АЗ.

Пример 3.1. Требуется определить средние длительности тактов АЗ на примере функционирования микропроцессорной ИС реального времени в условиях случайной длительности тактов и постоянном допустимом времени перерыва в работе системы $\tau_Д$=1 с, когда длительности решения задач много больше длительности пауз между ними, что позволяет на большом интервале времени управления определять длительности тактов без учета пауз между задачами.

Решение.

На первом шаге находят аналитическое выражение функции плотности случайного времени v существования отказа ВМ до принятия решения по его обнаружению. С этой целью можно воспользоваться формулой (3.7), которая применительно к условиям данного примера получена в преобразованиях Лапласа для произвольного порядка распределения Эрланга длительности такта и может быть применена для различных дисциплин АЗ.

На втором шаге поставленная задача решается с помощью формулы (3.8), которая в явном виде позволяет в условиях рассматриваемого примера рассчитывать вероятность успешной адаптации к отказам для различных дисциплин АЗ и при произвольном порядке распределения Эрланга длительности такта.

На третьем шаге с помощью исходных данных табл. III.3.7 рассчитываются численные значения вероятности β в зависимости от вида дисциплины . Результаты расчетов для дисциплин ***Д4*** и ***Д6*** приведены в виде графических зависимостей $\beta=f(x_Д)$. По этим зависимостям, а также по результатам расчетов с помощью формулы (3.8) при условии, что все ВМ охвачены встроенным контролем с вероятностью правильного обнаружения α, определены значения допустимого количества тактов для различных дисциплин АЗ (табл. III.3.8), при которых обеспечивается вероятность $\gamma_{АЗ}\geq 0{,}9999$, где $\gamma_{АЗ}=\gamma+\beta-\gamma\cdot\beta$.

На четвертом шаге задаем поправку расчетной длительности такта равной $\Delta\tau=0{,}2\tau$ **[**47**].**

Значения средней длительности такта для 4 и 8 основных ВМ, рассчитанные с учетом поправки на время загрузки в контролирующие ВМ команд и операндов, приведены в правых частях клеток табл. III.3.8.

Данные табл. III.3.8 свидетельствуют о том, что уменьшение времени адаптации на один такт, приводит к такому же эффекту обнаружения отказов, что и повышение вероятности правильного обнаружения отказов аппаратным контролем от уровня $\alpha=0{,}4$ до $\alpha=0{,}8$ или от $\alpha=0{,}8$ до $\alpha=0{,}9$. Этот результат имеет место для всех дисциплин активной защиты и характеризует большие преимущества данной концепции по сравнению с традиционными методами обеспечения надежности информационных систем реального времени.

Этот результат также означает, что при применении АЗ можно ограничиться экономичными встроенными средствами обнаружения неисправностей или даже отказаться от их применения.

Соотношения, приведенные в табл. III.3.8, позволяют определить применительно к наиболее развитым дисциплинам ***Д6*** и ***Д7*** максимально допустимую длительность такта АЗ при условии, что случайная длительность такта распределена по закону Эрланга 2-го и выше порядков. С другой стороны, если для этих же дисциплин в предельном случае отсутствия последействия принять условие экспоненциального распределения длительности такта АЗ, то можно установить нижнюю границу длительности такта АЗ. Например, при $\alpha<0{,}4$ для рассматриваемых дисциплин справедливо соотношение $x_{\text{д}} \geq \text{INT}\left[\dfrac{3m+33}{2}\right]$, из чего следует, что нижняя граница средней длительности такта при $m=8$ и $\tau_{\text{д}}=1$ с составляет 23,2 мс, т.е. 23,2 мс $< \tau^{*} <$ 34,1 мс.

Таблица III.3.8

α	**0…0,39**	**0,40…0,79**	**0,80…0,90**
τ^* ***(m=4)*** / ***(m=8)*** / Д 1…3	23,5 мс / 9,6 мс / $8m + 2$	24,3 мс / 11,6 мс / $8m + 1$	25,0 мс / 12,5 мс / $8m$
τ^* ***(m=4)*** / ***(m=8)*** / Д4	36,4 мс / 23,5 мс / $3m + 10$	38,1 мс / 24,3 мс / $3m + 9$	40,0 мс / 25,1 мс / $3m + 8$
τ^* ***(m=4)*** / ***(m=8)*** / Д5	36,4 мс / 21,1 мс / $4m + 6$	38,1 мс / 21,6 мс / $4m + 5$	40,0 мс / 22,2 мс / $4m + 4$
τ^* ***(m=4)*** / ***(m=8)*** / Д6, Д7	44,5 мс / 34,1 мс / $\mathrm{INT}\left[\frac{3m+23}{2}\right]$	48,4 мс / 35,5 мс / $\mathrm{INT}\left[\frac{3m+21}{2}\right]$	51,6 мс / 37,2 мс / $\mathrm{INT}\left[\frac{3m+19}{2}\right]$

Поскольку быстродействие современных вычислительных средств превышает десятки миллионов операций в секунду, то в течение 20 мс могут выполняться задачи объемом в десятки и сотни тысяч операций. Поэтому практически возможно уменьшить длительность такта АЗ в десятки и сотни раз и, следовательно, примерно в такое же количество раз увеличить количество тактов, размещаемых в заданном интервале $\tau_Д$. Таким образом, концепция адаптивной отказоустойчивости открывает также новые возможности в применении временного резервирования.

III.3.10. Надежность отказоустойчивых информационных систем реального времени

Концепция адаптивной отказоустойчивости (активной защиты) обобщает известные методы обеспечения отказоустойчивости информационных систем реального времени, в том числе традиционные методы структурного резервирования. Поэтому оценка надежности систем с активной защитой возможна на основе сопоставления показателей надежности ИС с АЗ и ИС с традиционными методами резервирования (например, с дублированием ВМ или со скользящим резервированием этих модулей). Поскольку известны формульные выражения показателей безотказности и готовности резервированных систем, то естественно при сопоставительном анализе ИС с АЗ и ИС с указанными методами резервирования ограничиться этими показателями.

III.3.10.1. Безотказность восстанавливаемых систем с активной защитой

Безотказность восстанавливаемой информационной системы длительного действия оценивается с помощью среднего времени до первого отказа $\bar{t}_{\text{О}}$ системы (или составных объектов с АЗ) и вероятности безотказной работы $P_{\text{О}}(t)$. При этом показатель $\bar{t}_{\text{О}}$ может быть строго определен по формуле (2.16), а при определении второго показателя ограничимся его оценкой снизу по формуле $P_{\text{О}}(t) \geq \exp(-t / \bar{t}_{\text{О}})$.

Причины возникновения отказа объекта ИС с АЗ:

- из общего числа $m+k$ ВМ в состоянии отказа находятся $k+1$ ВМ;
- количество отказавших ВМ меньше числа резервных k, но либо отказ не обнаружен встроенным контролем ВМ и не выпол-

нено своевременное переключение на резерв (вероятность этого события $1–\gamma=1–\alpha\theta$, где α – вероятность правильного обнаружения отказа ВМ встроенными средствами контроля, $\theta=P\{t_{\text{пер}}\le\tau_{\text{Д}}\}$, $t_{\text{пер}}$ – случайное время переключения на резерв и восстановления вычислительного процесса), либо активная защита своевременно не отреагировала на отказ. Вероятность данного события равна $(1–\gamma)(1–\beta_i)$, где β_i – вероятность успешной адаптации ИС с АЗ к отказам ВМ при условии, что в реконфигурации участвуют k-i дополнительных ВМ (i=0, 1, …, k–1). Вероятность β_i определяется выражением

$$\beta_i = P\{\nu_{k-i} \le \tau_{\text{Д}} - t_{\text{у}}\},$$

где ν_{k-i} – случайная длительность существования скрытого отказа ИС с АЗ (k-i)-го уровня. $G_0(t)=1–[1–F(t)]^{m+k}$

Очевидно, что при $i = 0$ имеет место вероятность β_0 – вероятность успешной адаптации к отказам в условиях исправности m основных и k дополнительных ВМ.

Граф состояний безотказности ИС с АЗ представлен на рис. III.3.16.

В рассматриваемой модели безотказности системы принято, что ее поведение в пространстве показанных на рис. III.3.15 состояний описывается полумарковским случайным процессом при следующих предпосылках: в системе в течение времени существования скрытого отказа ВМ (это время при наличии АЗ не превышает $\tau_{\text{Д}}$) отказы или восстановления других ВМ отсутс-

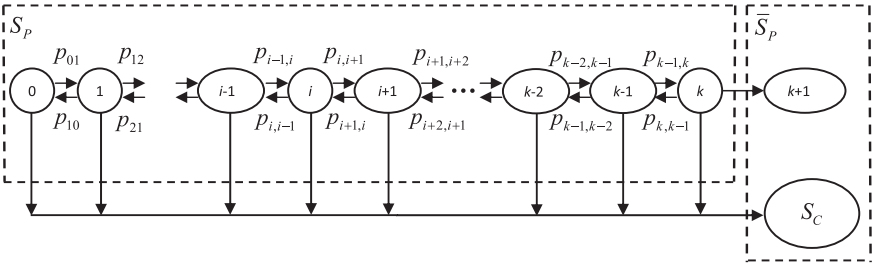


Рис. III.3.16. Граф состояний безотказности восстанавливаемой ИС с активной защитой

твуют, а при отсутствии АЗ возможен еще отказ не более одного ВМ; предполагается независимость интенсивности отказов ВМ от состояний их наблюдаемости.

Обозначим случайное время безотказной работы ВМ символом ξ, а время его восстановления – η. Это случайные величины с законами распределения $F(t)$ и $H(t)$.

Времена существования скрытых отказов ВМ при наличии АЗ (k-i)-го уровня $v_{k-i}<<\xi$ – случайные величины с функциями распределения $\Phi_{k-i}(t)$.

Множество работоспособных состояний системы $S_p=\{0, 1, \ldots, i, \ldots, k-1, k\}$, множество неработоспособных состояний $\overline{S}_P = \{S_c, k+1\}$, где S_c – подмножество состояний скрытых отказов ($0\le i\le k$) модулей длительностью больше допустимой.

Состояния графа: 0 – исправны $m+k$ ВМ;…; i – отказали $1\le i\le k$ ВМ, длительность этих отказов не превысила $\tau_Д$; …; $k+1$ – отказали $k+1$ или более ВМ.

Функции распределения безусловного времени пребывания полумарковского случайного процесса в состояниях $i\in S_p$ имеют следующий вид:

$$G_0(t) = 1-[1-F(t)]^{m+k};\ G_i(t) = 1-[1-F(t)]^{m+k-i}[1-H(t)],$$

где $i=1, 2, \ldots, k$

Переходные вероятности вложенной Марковской цепи определяются с помощью следующих выражений:

$$p_{01} = \Phi_0(t_Д);\ p_{i,i+1} = \Phi_{k-i}(t_Д)\int_0^\infty f_i(t)[1-H(t)]dt;$$

$$p_{i,i-1} = \int_0^\infty h(t)[1-(1-F(t))^{m+k-1}]dt,$$

где $f_i(t) = -\dfrac{d[1-F(t)]^{m+k-1}}{dt};\ h(t) = -\dfrac{d[1-H(t)]}{dt};\ i=1,2,\ldots,k$

При экспоненциальных распределениях отказов и восстановлений установлено, что ***среднее время до отказа*** ИС с ***одноуровневой*** АЗ (кратность *1/m*)

$$\bar{t}_{\text{о}}(1)=\frac{[(m+1)(\gamma+\beta_0-\gamma\beta_0)+m]\rho+1}{(m+1)\lambda[m\rho+(1-\gamma)(1-\beta_0)]}, \qquad (3.14)$$

где $\rho=\lambda/\mu$.

Заметим, что при предельном значении вероятности успешной адаптации к отказам $\beta_0=1$ формула (3.14) преобразуется в известную справочную формулу расчета средней наработки до отказа восстанавливаемой системы с нагруженным резервом кратности $1/m$

$$\bar{t}_{\text{о}}(1)\big|_{\beta_0=1}=\frac{(2m+1)\lambda+\mu}{(m+1)m\lambda^2}.$$

В частности, при одном основном устройстве имеет место известная справочная формула для дублированной восстанавливаемой системы с нагруженным резервом $\bar{t}_{\text{о}}(1)=\dfrac{3\lambda+\mu}{2\lambda^2}$.

Данные справочные формулы носят условный характер идеального контроля отказов составных элементов (абсолютно достоверного и абсолютно надежного) системы и идеальной системы адаптации к отказам. Установленная формула (3.14) лишена этих условностей и может быть рекомендована для практического применения.

Среднее время до отказа системы с ***двухуровневой*** защитой (аналог структурного резервирования кратности $2/m$) имеет следующий вид:

$$\bar{t}_{\text{о}}(2)=\frac{\left(\begin{array}{c}(m+1)[(1-\gamma)(1-\beta_1)+m\rho]\rho+(m+2)[(m+1)\cdot\\ \cdot(\gamma+\beta_1-\gamma\beta_1)\rho+m\rho+1]\rho(\gamma+\beta_0-\gamma\beta_0)+m\rho+1\end{array}\right)}{\left(\begin{array}{c}(m+2)\lambda[(m+1)m\rho^2+(m+1)\rho(1-\gamma)(1-\beta_1)+\\ +(m\rho+1)(1-\gamma)(1-\beta_0)]\end{array}\right)} \qquad (3.15)$$

При предельных значениях $\beta_0=\beta_1=1$ формула (3.15) преобразуется в известную справочную формулу расчета ***средней наработки до отказа*** восстанавливаемой системы с нагруженным резервом кратности $2/m$:

$$\bar{t}_{\text{о}}(2)\,|_{\beta_0=\beta_1=1}=\frac{2[(m+1)\rho+1](m+1)\rho+1}{(m+2)(m+1)m\rho^2\lambda}$$

Заметим, что в противоположном случае, когда $\gamma=\beta_0=\beta_1=0$, резервирование не реализуется и имеют место ожидаемые результаты

$$\bar{t}_{\text{о}}(1)\,|_{\gamma=\beta_0=0}=\frac{1}{(m+1)\lambda};\ \bar{t}_{\text{о}}(2)\,|_{\gamma=\beta_0=\beta_1=0}=\frac{1}{(m+2)\lambda}.$$

Уровни безотказности ИС с АЗ оценим применительно к характерным дисциплинам ***Д1, Д4, Д5*** и ***Д6*** при следующих типичных исходных данных***:***

• Интенсивности отказов системы $\lambda(1/\text{ч})=10^{-3}$; $5\cdot10^{-4}$; 10^{-4}; $5\cdot10^{-5}$ интенсивности восстановлений $\mu(1/\text{ч})=0{,}5$; 1; количество основных ВМ $m=2,4,6,8$; количество избыточных ВМ $k=1, 2, \ldots, 8$ (при $k=m$ – полная m-уровневая АЗ); вероятность обнаружения отказов собственными средствами и своевременного их устранения $\gamma=0{,}4$; 0,6; 0,8;

• Система работает с повышенной информационной нагрузкой, когда кратковременными паузами в решении функциональных задач можно пренебречь; допустимое время перерыва в работе постоянно, а длительности тактов случайны. При этих условиях вероятности $\beta_i = F\{f_{\nu_{k-i}}(t), \tau_{\text{Д}}, t_{\text{у}_i}\}$ можно определить по формуле (3.8) в зависимости от $x^*_{\text{Д}} = (\tau_{\text{Д}} - t_{\text{у}})/\bar{\tau}$ параметров дисциплины защиты $A_i=Д_i(m, k–i)$, $b_i=Д_i(m, k–i)$, $x_{\text{у}i}=Д_i(m, k–i)$.

Результаты сравнения безотказности ИС с развитыми дисциплинами одноуровневой АЗ (***Д4, Д5*** и ***Д6***) относительно ИС с

простейшей дисциплиной защиты ***Д1*** показаны на рис. III.3.17 в виде графиков отношений

$$\kappa_{Б} = \frac{\bar{t}_{\text{о}}(1)(Д4, Д5, Д6)}{\bar{t}_{\text{о}}(1)(Д1)}$$

от допустимого количества тактов $x^{*}_{\text{Д}}$ и количества m основных ВМ.

Из графиков, представленных на рис. III.3.17, следует:

• Наибольшими преимуществами в повышении безотказности ИС обладает дисциплина ***Д6*** (и последующие дисциплины АЗ ***Д7, Д8*** и т.д.), ее достоинства тем выше, чем больший случайный разброс длительности такта. Дисциплины ***Д4*** и ***Д5*** имеют примерно одинаковые достоинства в обеспечении безотказности ИС (рис. III.3.17 а) ;

• Преимущества дисциплин с развитыми возможностями адаптации по отношению к простейшим дисциплинам обусловлены их значительно большей скоростью реакции на отказ ВМ и, следовательно, существенно меньшим временем существования скрытого отказа (в среднем в 1,5 – 2 раза при $m \leq 8$). Эти преимущества играют решающую роль при выборе дисциплин в условиях малой длительности допустимого времени перерыва в работе системы $\tau_{\text{Д}}$. По мере увеличения этого времени возможности всех дисциплин активной защиты в обеспечении безотказности ИС становятся соизмеримыми.

Целесообразность повышения уровня АЗ следует из графиков $G_0(100)=f(k)$, представленных на рис. III.3.17 б. На этих графиках показана вероятность отказа системы с АЗ . которая определяется из выражения $G_0(100)=1–P_j(100)$. Графики, помеченные цифрами 1, 2, 3 и 4, относятся, соответственно к дисциплинам ***Д1, Д4, Д5*** и ***Д6*** и построены для условий явно недостаточного резерва тактов АЗ ($x^{*}_{\text{Д}} \leq 15$). Это сделано с целью изучения влия-

ния уровня защиты (k) и дисциплин АЗ на надежность системы в условиях жестких временных ограничений.

Полученные результаты свидетельствуют о том, что в ИС с простейшей дисциплиной ***Д1*** введение одного, двух и даже четырех дополнительных ВМ не приводит к снижению вероятности отказа системы, – более того, эта вероятность даже увеличивается, поскольку общее количество ВМ в системе возросло, а временные возможности своевременного выявления отказавших основных ВМ и замены их избыточными остались практически прежними. Однако по мере дальнейшего увеличения количества дополнительных ВМ сокращается время адаптации

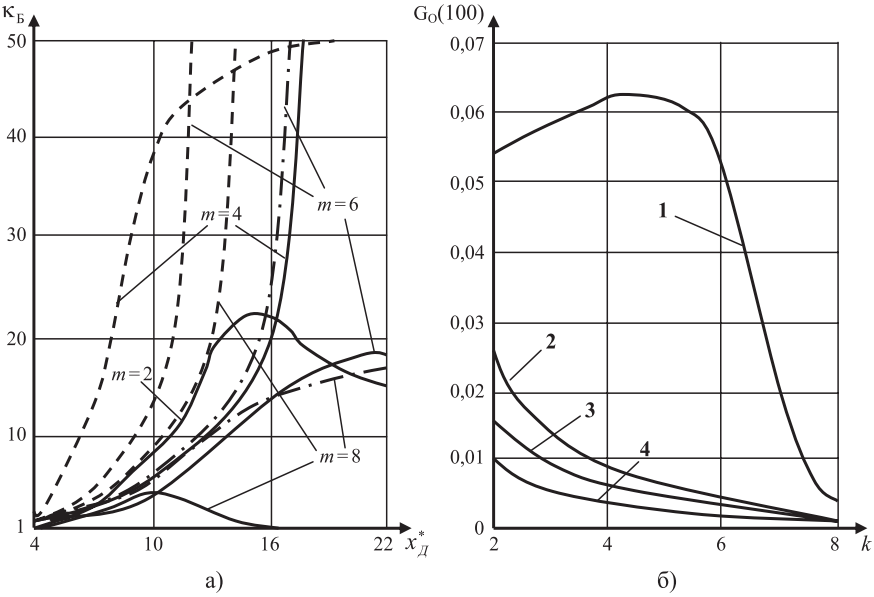


Рис. III.3.17. Зависимости показателей безотказности ИС с АЗ от составных параметров: а – зависимости нормированного среднего времени до отказа ИС с одноуровневыми дисциплинами Д4 (сплошные линии), Д5 (штрих – пунктирные линии), Д6 (штриховые линии) от допустимого количества тактов $x^*_Д$ перерыва в работе и от числа основных ВМ при: ρ=0,001, γ=0,8; б – зависимость вероятности отказа ИС от уровня k АЗ и от типа дисциплины Д1, Д4, Д5 и Д6 (кривые 1, 2, 3, 4 соответственно) при: t=100ч, $x^*_Д \le 15$, m=8, ρ=0,001, γ=0,8.

ИС к отказам и активная защита, построенная с помощью дисциплины ***Д1***, по своим возможностям приближается к другим дисциплинам АЗ;

В ИС с высокими уровнями активной защиты резерв времени $\tau_Д$, необходимый для достижения заданной вероятности $\beta_З$, может быть меньше, чем в ИС с одноуровневой защитой. Анализ показал, что при недостаточном естественном резерве времени безотказность ИС можно существенно повысить только при полной ($k = m$) или близкой к ней защите. И, наоборот, при значительном времени $\tau_Д$ вероятность безотказной работы системы с АЗ начинает существенно возрастать при введении более одного дополнительного ВМ. При этом интенсивность восстановления незначительно влияет на надежность ИС, если имеют место низкие уровни АЗ. Однако при полной АЗ интенсивность восстановления оказывает существенное влияние на надежность ИС.

Данные результаты исследования указывают на возможность выбора рациональных путей повышения безотказности ИС с АЗ. Они свидетельствуют о том, что определяющими факторами обеспечения адаптивной отказоустойчивости ИС являются допустимое количество тактов АЗ и уровень защиты.

III.3.10.2. Готовность восстанавливаемых систем с активной защитой

Готовность восстанавливаемых информационных систем длительного действия с активной защитой k-го уровня защиты оценивается с помощью следующего стационарного коэффициента готовности

$$K_Г^{(k)} = \frac{T_О^{(k)}}{T_О^{(k)} + T_В^{(k)}},$$

где $T_О^{(k)}$ – среднее время между отказами ИС с АЗ k-го уровня; $T_В^{(k)}$ – среднее время восстановления ИС с АЗ путем ремонта или

устранения скрытых отказов ВМ (длительностью большей $\tau_Д$) путем замещения отказавшего ВМ на резервный.

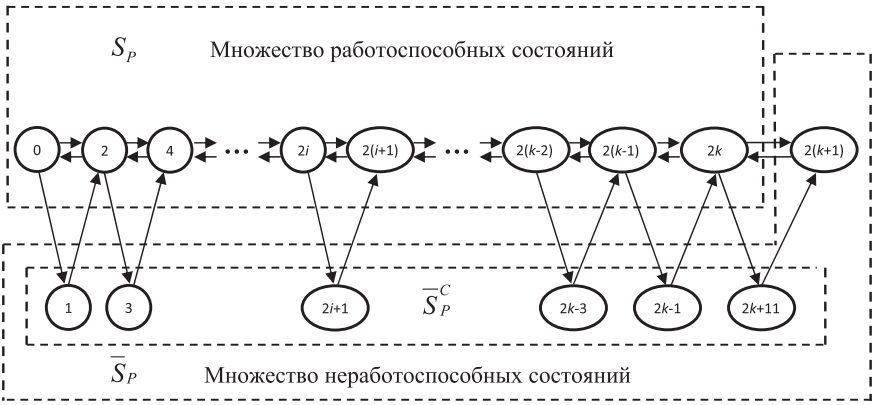


Рис. III.3.18. Граф состояний готовности восстанавливаемой ИС с активной защитой

Исходные предпосылки для определения показателей готовности системы изложены в предыдущем параграфе. Кроме того, предполагается, что в условиях активной защиты ИС в течение чрезвычайно короткого времени существования скрытого отказа ВМ практически исключены события отказов и восстановлений других ВМ.

Граф для определения готовности восстанавливаемой ИС с активной защитой показан на рис. III.3.18.

Множество работоспособных состояний

$$S_{\text{Р}} = \{0, 2, 4, ..., 2(i-2), 2i, 2(i+1), ..., 2(k-1), 2k\};$$

Множество неработоспособных состояний

$\overline{S}_{\text{Р}} = \{\overline{S}_{\text{Р}}^C, 2(k+1)\}$, где подмножество состояний скрытых отказов ВМ длительностью больше допустимого времени

$$\overline{S}_{\text{Р}}^C = \{1, 3, ..., 2i-1, 2i+1, ..., 2k-3, 2k-1, 2k+1\}.$$

Состояния графа: 0 – исправны $m+k$ ВМ; $2i$ – отказали $1 \le i \le k$ ВМ, длительность этих отказов не превысила $\tau_Д$; $(2i\text{-}1)$ – отка-

зали $1 \le i \le k$ ВМ, но длительность этих отказов превысила $\tau_Д$; …; $2(k+1)$ – отказали $k+1$ ВМ.

При экспоненциальных законах распределения отказов и восстановлений ***среднее время между отказами*** ИС ***с одноуровневой АЗ*** рассчитывается с помощью выражения

$$T_О(1) = \frac{(m+1)\rho + 1}{(m+1)\lambda[m\rho + (1-\gamma)(1-\beta_0)]}, \qquad (3.16)$$

где $\rho = \lambda/\mu$

В свою очередь, ***среднее время до восстановления ИС с одноуровневой АЗ*** определяется выражением

$$T_В(1) = \frac{T_1\mu(1-\gamma)(1-\beta_0) + m\rho}{\mu[(1-\gamma)(1-\beta_0) + m\rho]}, \qquad (3.17)$$

где $T_1 = \int_{\tau_д}^{\infty} t f_{v1}^{Д}(t)dt$, $f_{v1}^{Д}(t)$ – функция плотности времени скрытого отказа относительно конкретной дисциплины одноуровневой АЗ.

При $\beta_0 = 1$ (идеальная одноуровневая АЗ) выражения для средней наработки на отказ (3.16) и среднего времени до восстановления (3.17) преобразуются в известные формулы идеального резервирования кратности *1/m* при условии одной ремонтной бригады, т.е.

$$T_О(1) = \frac{(m+1)\rho + 1}{(m+1)m\lambda\rho};\ T_В(1) = \frac{1}{\mu}.$$

Рассмотрим двухуровневую активную защиту (*2/m*) ИС. При экспоненциальных законах распределения отказов и восстановлений ***среднее время между отказами*** ИС ***с двухуровневой АЗ*** определяется с помощью выражения

$$T_{\text{О}}(2)=\frac{1+(m+2)\rho[1+(m+1)\rho]}{(m+2)\lambda[(m+1)m\rho^2+(m+1)\rho(1-\gamma)(1-\beta_1)+(1-\gamma)(1-\beta_0)]} \quad (3.18)$$

а ***среднее время до восстановления*** системы ***с двухуровневой активной защитой***

$$T_{\text{В}}(2)=\frac{T_1(1-\gamma)(1-\beta_0)+T_3(1-\gamma)(1-\beta_1)(m+1)\rho+(m+1)m\rho^2 1/\mu}{(m+1)m\rho^2+(m+1)\rho(1-\gamma)(1-\beta_1)+(1-\lambda)(1-\beta_0)}, \quad (3.19)$$

где $T_3=\int\limits_{\tau_д}^{\infty} tf_{v2}^{д}(t)dt$, $f_{v2}^{д}(t)$ – функция плотности времени скрытого отказа относительно конкретной дисциплины двухуровневой АЗ.

В предельных случаях $\beta_0=\beta_1=1$ эти выражения преобразуются в известные формулы идеального резервирования с восстановлением кратности $2/m$ при условии одной ремонтной бригады

$$T_{\text{О}}(2)=\frac{1+(m+2)\rho[1+(m+1)\rho]}{(m+2)(m+1)m\rho^2\lambda}; \quad T_{\text{В}}(2)=\frac{1}{\mu}.$$

Результаты сравнительного исследования свойств восстанавливаемости ИС с одноуровневой АЗ показаны в виде графических зависимостей на рис. III.3.19, где

$$\eta_{\text{В}}=\frac{T_{\text{В}}(1)(Д4, Д5, Д6)}{T_{\text{В}}(1)(Д1)} \quad (3.20)$$

есть отношение средних времен восстановления работоспособности ИС с развитыми дисциплинами защиты к среднему времени до восстановления ИС с дисциплиной ***Д1.*** Из этих графиков следует, что среднее время до восстановления систем с простейшими дисциплинами в десятки и даже в сотни раз меньше, чем это время в системах с дисциплинами ***Д4, Д6, Д7*** и т.д., причем чем больше основных ВМ, тем более сильно выражено это соотношение. Объясняется данный эффект тем, что при малом до-

пустимом времени $\tau_Д$ перерыва в работе имеют место события, когда АЗ не успевает отреагировать на отказ основного ВМ и возникают отказы, соизмеримые по времени существования с $\tau_Д$. Эти отказы устраняются с помощью избыточного ВМ. Они существуют наряду с отказами системы, которые устраняются ремонтной бригадой. Поэтому средняя длительность восстановления ИС существенно меньше средней длительности ремонта. Чем меньше возможности адаптации ИС к отказам, чем замедленнее реакция активной защиты, тем больше кратковременных отказов в системе и, как следствие, тем меньше общая средняя расчетная длительность восстановления всех отказов. Время восстановления систем с простейшими дисциплинами в десятки и даже в сотни раз меньше, чем это время в системах с дисциплинами ***Д4, Д6, Д7*** и т.д., причем чем больше основных ВМ, тем более сильно выражено это соотношение. Объясняется данный эффект тем, что при малом $\tau_Д$ имеют место события, когда АЗ не успевает отреагировать на отказ основного ВМ и возникают отказы, соизмеримые по времени существования с $\tau_Д$. Эти отказы устраняются с помощью избыточного ВМ. Они существуют наряду с отказами системы, которые устраняются ремонтной бригадой. Поэтому средняя длительность восстановления ИС существенно меньше средней длительности ремонта. Чем меньше возможности адаптации ИС к отказам, чем замедленнее реакция активной защиты, тем больше кратковременных отказов в системе и, как следствие, тем меньше общая средняя расчетная длительность восстановления всех отказов.

Свойства готовности ИС с одноуровневой АЗ можно оценить с помощью графических зависимостей, показанных на рис. III.3.20, где

$$R_Г = \frac{(1 - K_Г^{(1)})(Д1)}{(1 - K_Г^{(1)})(Д4, Д5, Д6)}, \qquad (3.21)$$

где $K_{\Gamma}^{(1)}$ – коэффициент готовности ИС с одноуровневой АЗ, построенной на основе одной из дисциплин ***Д1, …, Д7*** и т.д.; $(1–K_{\Gamma}^{(1)})$ – коэффициент неготовности (простоя) системы с АЗ.

Данные графики свидетельствуют о том, что возможности дисциплин ***Д4*** и ***Д5*** в отношении обеспечения готовности ИС примерно одинаковы и при этом в десятки и даже в сотни раз выше, чем возможности простейших дисциплин, таких как ***Д1, Д2, Д3***. Наибольшими возможностями в обеспечении готовности, также как и безотказности, обладают дисциплины АЗ ***Д6, Д7*** и т.д.

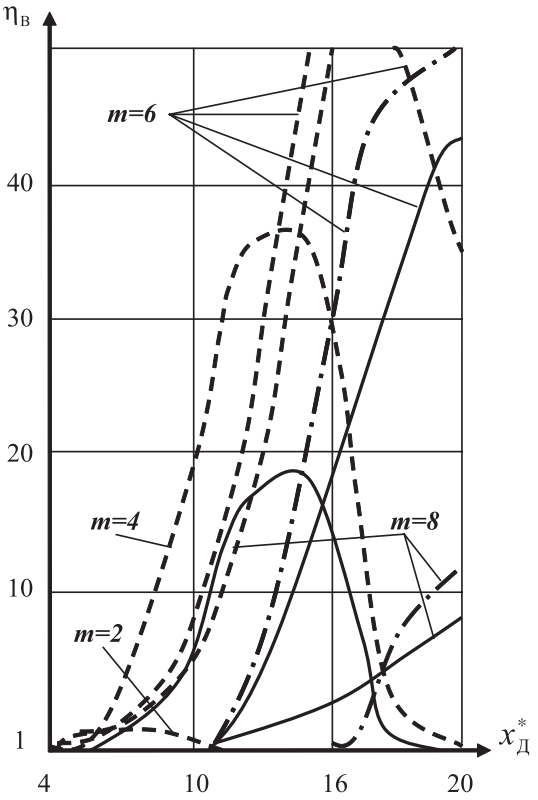


Рис. III.3.19. Зависимость нормированного по формуле (3.20) среднего времени до восстановления ИС с дисциплинами Д4 (сплошные линии), Д5 (штрих – пунктирные линии), Д6 (штриховые линии) от количества m основных ВМ и $x_{Д}^{*}$ тактов АЗ случайной длительности при: ρ=0,001; γ=0,8.

Графические зависимости нормированных коэффициентов простоя ИС с АЗ, приведенные на рис. III.3.20, построены для случайных тактов АЗ и, следовательно, случайных интервалов реконфигурации ИС. Проведенные исследования показывают, что применительно к ИС с постоянными тактами АЗ эти выводы еще более усиливаются.

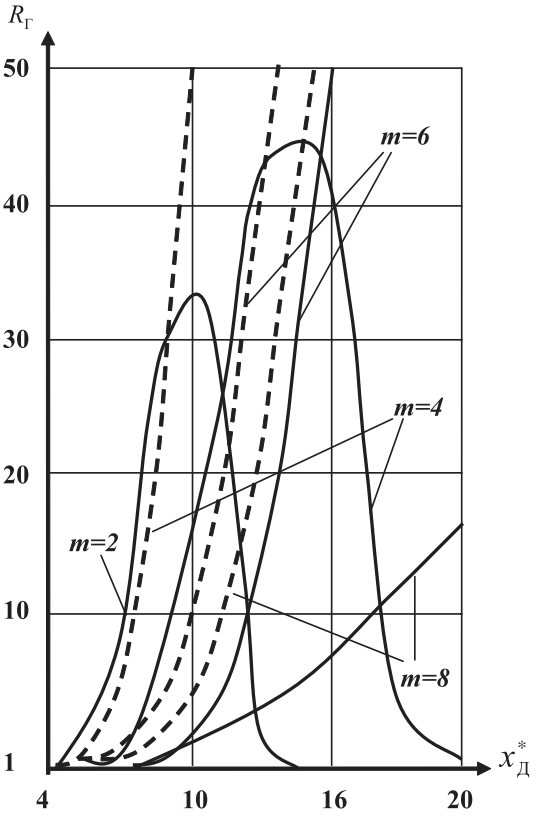


Рис. III.3.20. Зависимость нормированного по формуле (3.21) коэффициента простоя ИС с дисциплинами Д4 и Д5 (сплошные линии) и Д6 (штриховые линии) от количества m основных ВМ и $x^*_Д$ тактов АЗ случайной длительности при: ρ=0,001; γ=0,8.

Большая зависимость готовности ИС от допустимого количества тактов наблюдается также и при двухуровневой АЗ

(рис. III.3.21а). При малом количестве допустимых тактов для обнаружения отказа ($x^*_Д$<15) уровень готовности системы с дисциплиной АЗ ***Д6*** (кривая 4) значительно выше готовности системы с дисциплиной ***Д1*** (кривая 1). Так, при указанном количестве тактов коэффициент простоя системы ($K_{ПР}$=1–$K_Г$) с дисциплиной ***Д6*** более, чем на порядок меньше коэффициента простоя системы с дисциплиной ***Д1***. Вместе с тем, уже при практически достижимом количестве тактов $x^*_Д$≥25 все сравниваемые дисциплины АЗ обеспечивают примерно одинаковый высокий уровень готовности ИС.

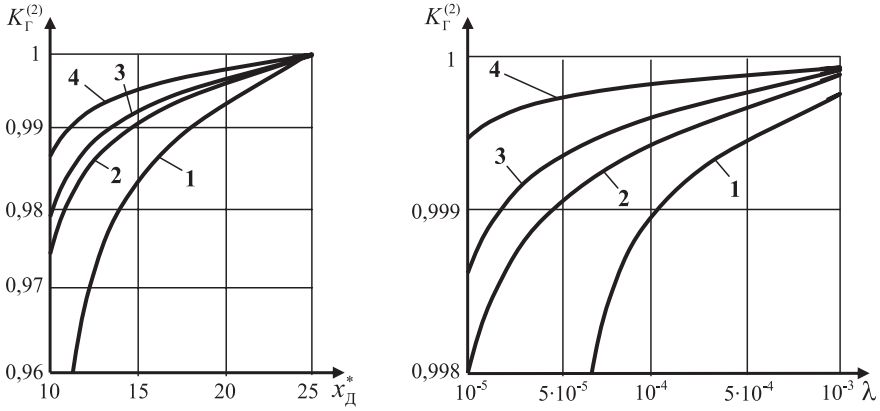


Рис. III.3.21. Зависимость коэффициента готовности системы с двухуровневой АЗ при m=8, ρ=0,001, γ=0,8 от: **а** ***– от допустимого количества тактов и от типа дисциплины (1-Д1,2-Д4,3-Д5,4-Д6);*** **б** ***– от интенсивности отказов ВМ (λ 1/ч) и от допустимого количества тактов обнаружения отказа (1 –*** **15,** ***2 –*** **18,** ***3 –*** **25,** ***4 –*** **33)**

Влияние надежности составных ВМ на готовность ИС заметно уступает влиянию времени адаптации системы к отказам. Так, при снижении интенсивности отказов ВМ на два порядка коэффициент простоя системы уменьшается только в 1,5-2 раза (рис. III.3.21 б), тогда как при увеличении резерва времени в 2 раза этот же показатель уменьшается на порядок

(рис. III.3.21 а). Заметим также, что чем выше уровень адаптации ИС к отказам, тем слабее зависимость системы от надежности составных ВМ (кривые 4, 3, 2 и 1 на рис. III.3.21 б).

Анализ показателей безотказности и готовности ИС с различными дисциплинами АЗ свидетельствует о больших преимуществах развитой многоуровневой дисциплины ***Д7*** перед простейшими дисциплинами ***Д1***, ***Д2*** или ***Д3***. Дисциплина АЗ ***Д7***, и тем более дисциплины с многоуровневой защитой типа ***m/m,*** обеспечивают наиболее эффективную адаптацию ИС к отказам составных ВМ. Однако, при небольшом количестве ВМ ($m = 2, 3, 4$), каждый из которых представляет сложное программно – аппаратное устройство, более рационально применение дисциплин ***Д6***, ***Д5*** или ***Д4***, что обеспечивает желаемый уровень надежности системы при меньших экономических затратах.

III.3.8.3. Сравнительная оценка эффективности активной защиты и структурного резервирования

В целях оценки эффективности концепции активной защиты выполнен сравнительный анализ безотказности и готовности ИС с АЗ, с одной стороны, и ИС с традиционными методами структурного резервирования соответствующей кратности, с другой стороны. Для большей уверенности в результатах сравнения принято, что при структурном резервировании обеспечивается мгновенное время переключения на резерв и идеально надежны контрольные и переключающие устройства. Однако учитывается реальный уровень вероятности правильного обнаружения отказов ($\alpha<1$), а при моделировании надежности объектов со структурным резервом и неидеальным контролем использовались материалы главы III.2.

Результаты сравнительного анализа средней наработки до отказа ИС с одноуровневой активной защитой со средней наработкой до отказа ИС со структурным резервированием кратнос-

ти 1/m показаны на рис. III.3.22. Эти результаты представлены графическими зависимостями, рассчитанными по формуле

$$\kappa_{Б}^{*} = \frac{\bar{t}_{о}(1)(Д4, Д5, Д6, Д1)}{\bar{t}_{о}(1/m)} \qquad (3.22)$$

Полученные результаты позволяют сделать следующие вывод: при одинаковой избыточности применение АЗ позволяет в сотни раз увеличить среднюю наработку до отказа системы с АЗ по сравнению со структурным резервированием. Расчеты показыва-

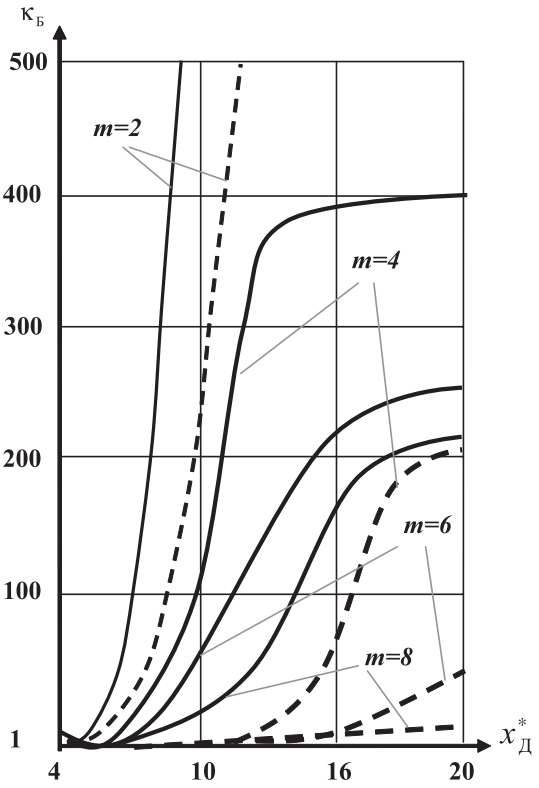


Рис. III.3.22. Зависимости выигрыша в отказоустойчивости восстанавливаемой ИС при использовании одноуровневой АЗ по отношению к скользящему резервированию кратности 1/m (построенные с помощью формулы (3.22)) от допустимого количества тактов и от типа дисциплины (Д4, Д5, Д6 – сплошные линии; Д1 – штриховые линии) при ρ=0,001, γ=0,8

ют, что этот вывод справедлив и в том случае, если в условиях применения развитых дисциплин АЗ составные ВМ не охвачены эффективным контролем, а в условиях применении структурного резервирования вероятность правильного обнаружения отказов составных ВМ высока, но не превышает уровня 0,99.

Из данных графиков следует:

– выигрыш в безотказности тем больше, чем меньше число основных ВМ. По мере увеличения их количества увеличивается

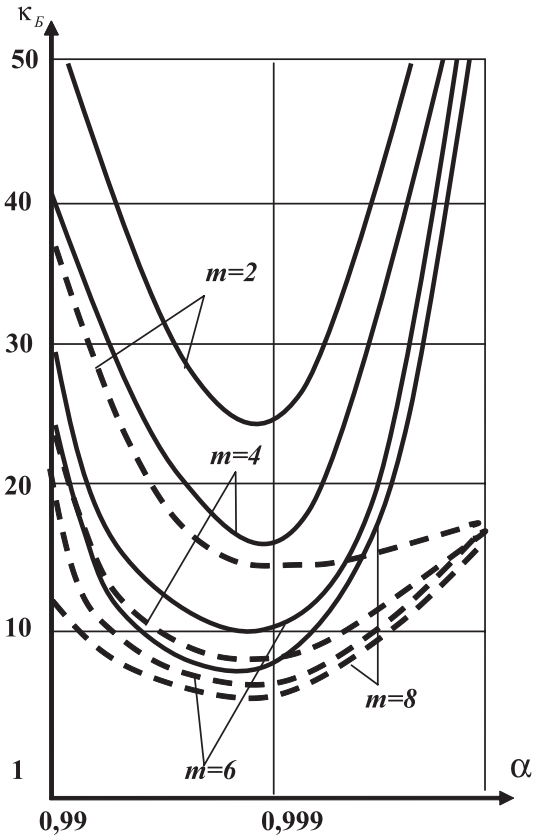


Рис. III.3.23. Графики (сплошные линии) отношения безотказности ИС с дисциплиной Д6 к безотказности ИС с резервированием кратности 1/m и с высоким уровнем встроенного аппаратного контроля (0,99≤α≤1) при ρ=0,001, γ=0,8; графики (пунктирные линии), построенные при тех же условиях, но при применении аппаратно-программного контроля ВМ

время адаптации ИС к отказам. Однако, в случае применения развитой дисциплины ***Д6*** выигрыш в безотказности сохраняется весьма внушительным и при большом количестве ВМ;

– если $x^*_Д$ не превышает 15…20 тактов АЗ и число основных ВМ порядка 10, то применение простейших дисциплин Д1 или Д3 не приводит к заметным преимуществам по сравнению с резервированием кратности $1/m$.

В проведенном анализе не затрагивался вопрос, что лучше: применение активной защиты или структурного резервирования в сочетании с высокоэффективной системой контроля работоспособности составных программно – аппаратных модулей. Ответ на этот вопрос в значительной мере может быть дан путем анализа двух семейств графических зависимостей, показанных на рис. III.3.23.

В первом случае контроль ВМ практически не влияет на потери производительности ИС, но требуется дополнительное контрольное оборудование объемом W_k, которое приближенно оценивается по формуле $\alpha=1–\exp(–\delta\Delta W)$ [1], где нормировочный коэффициент $\delta\approx5…10$, $\Delta W=W_k/(W_О+W_k)$, $W_О$ – объем основного оборудования.

Во втором случае периодически осуществляется тестовый контроль, имеются потери в производительности ИС, однако требуется существенно меньший объем контрольного оборудования. Действительно, $\alpha=\alpha_А+\alpha_П–\alpha_А\alpha_П$, где $\alpha_А$ и $\alpha_П$ – вероятности правильного обнаружения отказов аппаратным и программным средствами контроля соответственно.

Анализ графических зависимостей рис. III.3.23 позволяет сделать следующие выводы:

– применение одноуровневой активной защиты ИС с развитыми возможностями позволяет в десятки и даже в сотни раз повысить наработку системы до отказа по сравнению с традиционными решениями, основанными на структурном резервиро-

вании ВМ, которые кроме того обеспечены эффективным встроенным аппаратным контролем. Этот результат имеет место и тогда, когда при применении активной защиты вычислительные модули не обеспечены эффективным встроенным аппаратным контролем (объем контрольного оборудования не превышает (15-20)% от основного оборудования);

– активная защита ИС значительно эффективнее структурного резервирования ВМ, даже если каждый из них снабжен развитой системой аппаратно – программного контроля и при этом обеспечен достаточный резерв времени для реализации программного контроля (см. графики рис. III.3.23, показанные штриховыми линиями).

Экстремальный характер графических зависимостей рис. III.3.23 свидетельствует о наличии оптимального уровня вероятности правильного обнаружения отказов каждого ВМ в резервированной системе, при котором сбалансирован объем контрольного оборудования и эффект от его применения.

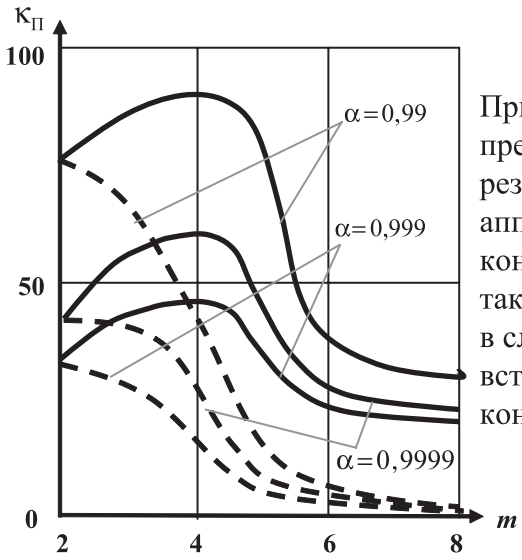
Сравнение по показателям безотказности эффективности применения в информационных системах активной защиты с эффективностью применения раздельного дублирования ВМ показало, что АЗ позволяет обеспечить от 2-3 раз до десятков раз больший уровень безотказности ИС.

Сравнительная оценка готовности системы с одноуровневой активной защитой, построенной с помощью дисциплины ***Д6***, и системы со структурным резервированием кратности $1/m$ проводилась с помощью следующей формулы

$$\kappa_{\text{П}} = \frac{(1 - K_{\text{Г}})(1/m)}{(1 - K_{\text{Г}})(Д6)} = \frac{K_{\text{П}}(1/m)}{K_{\text{П}}(Д6)}$$

Результаты сравнительного анализа готовности ИС показаны графическими зависимостями на рис. III.3.24. Эти графики (сплошные линии) показывают, что коэффициент простоя в ИС с

одноуровневой АЗ значительно меньше, чем в ИС с традиционной организацией резервирования. Этот результат имеет место даже при наличии в последних высокоэффективного встроенного аппаратного контроля. Причина этого в резком уменьшении количества отказов системы и их длительности. Применение в резервированных ИС программного контроля (если при этом приемлем уровень снижения производительности ИС) в сочетании с высокоэффективным встроенным аппаратным контролем (штриховые линии) при большом количестве ВМ обеспечивает соизмеримый с АЗ уровень готовности системы.



При малом количестве ВМ преимущества АЗ перед резервированием с аппаратно–программным контролем сохраняются на таком же уровне, как и в случае применения только встроенного аппаратного контроля

Рис. III.3.24. Графики (сплошные линии) отношения коэффициента простоя ИС с резервированием кратности 1/m и с высоким уровнем встроенного аппаратного контроля (0,99≤α≤1) к коэффициенту простоя ИС с дисциплиной Д6 при ρ=0,001, γ=0,8; графики (штриховые линии), построенные при тех же условиях, но при применении аппаратно – программного контроля ВМ

Технико-экономический эффект от применения в ИС активной защиты может быть определен с помощью коэффициента

$$K_{Э}(T) = \frac{C \cdot M \cdot T_{В} \cdot n(T)}{C \cdot M_{АЗ} \cdot T_{ПАЗ} \cdot n_{АЗ}(T)}, \qquad (3.23)$$

где C – штраф за простой одного ВМ в течение одной единицы времени, M, $M_{АЗ}$ – количество ВМ в исходной ИС и ИС с активной защитой соответственно, $T_{В}$, $T_{ПАЗ}$ – среднее время восстановления ИС без избыточности и среднее время простоя системы с АЗ соответственно, $n(T)$, $n_{АЗ}(T)$ – среднее количество отказов за большой календарный срок T исходной ИС без избыточности ВМ и ИС с АЗ соответственно.

В числителе выражения (3.23) произведение $C{\cdot}M$ определяет штраф за простой в единицу времени совокупности основных ВМ исходной ИС без избыточности, а произведение $T_{В}{\cdot}n(T)$ – общее время простоя ВМ исходной ИС. Следовательно, коэффициент $K_{Э}(T)$ определяет отношение штрафа за простой совокупности основных ВМ исходной ИС к штрафу за простой ИС с активной защитой в течение календарного времени.

При одном избыточном ВМ справедливо отношение $(M/M_{АЗ})=(m/(m+1))$ (здесь m – количество основных ВМ в исходной ИС), поскольку объем коммутаторов, узлов сравнения и устройства управления не превышает 5% от объема совокупности основных ВМ. Если учесть, что при применении дисциплины ***Д6*** более 20 тактов АЗ укладываются в допустимое время перерыва в работе, то это обстоятельство обеспечивает отсутствие в системе скрытых отказов. Таким образом, $T_{ПАЗ}=T_{В}$.

Отношение среднего количества отказов ВМ исходной ИС без избыточности к среднему количеству отказов вычислительных модулей ИС с АЗ за интервал наблюдения T можно оценить как $\frac{n(T)}{n_{АЗ}(T)} \approx \frac{T / T_{ИСХ}}{T / T_{О}(1)}$, где $T_{ИСХ}=1/m\lambda$ – среднее время до отказа ВМ исходной ИС без избыточности, λ – интенсивность отказов одного ВМ; $T_{О}(1)$ – средняя наработка на отказ ВМ ИС с АЗ

(определяется по формуле (3.16)). Следовательно, данное отношение при экспоненциальных законах распределения отказов и восстановлений ВМ оцениваем при $T>A$ (A – сколь угодно большое число) как

$$\frac{n(T)}{n_{\text{АЗ}}(T)} \approx \frac{T / T_{\text{ИСХ}}}{T / T_{\text{О}}(1)} = \frac{[1+(m+1)\rho]m\lambda}{(m+1)m\lambda\rho} = 1 + \frac{1}{(m+1)\rho},$$

где $\rho=\lambda/\mu$, $\mu=1/T_{\text{В}}$.

Таким образом, коэффициент технико – экономической эффективности применения АЗ определяем приближенно в виде $K_{\text{Э}} \approx \frac{m}{(m+1)} \cdot [1 + \frac{1}{(m+1)\rho}]$. При значениях $\rho=10^{-3}$ этот показатель составляет $K_{\text{Э}}>200$ для $m=2$ и $K_{\text{Э}}>100$ для $m=10$.

Следовательно, применение активной защиты позволяет в десятки раз уменьшить штрафы за простой рабочих ВМ нерезервированной ИС.

Применяя аналогичные рассуждения для структурного резервирования ИС при тех же исходных условиях и эффективной встроенной системе аппаратного контроля ВМ ($\alpha=0{,}999$), устанавливаем, что применение АЗ может обеспечить технико – экономический эффект в **15-20** раз выше резервирования кратности $1/m$ и почти на порядок выше, чем раздельное дублирование ВМ.

III.3.11.Вопросы для самоконтроля

1. Приведите и поясните граф состояний системы обеспечения отказоустойчивости и перечислите требования, предъявляемые к отказоустойчивости информационных систем (ИС).

2. Какие способы обеспечения отказоустойчивости ИС широко применяются на практике; их достоинства и недостатки?

3. Какие идеи положены в основу технологии адаптивной отказоустойчивости ИС?

4. Приведите и объясните способы назначения пар вычислительных модулей (ВМ) при реализации адаптивной отказоустойчивости (активной защиты) ИС.

5. Приведите и объясните способы формирования множеств основных и избыточных ВМ в активной защите.

6. Объясните содержание понятия «уровни активной защиты».

7. В чем состоит суть способов обнаружения и устранения неисправностей ВМ в системах с активной защитой?

8. Раскройте особенности организации временных интервалов активной защиты.

9. В чем суть базовых дисциплин активной защиты?

10. В чем суть модульных дисциплин активной защиты?

11. Оцените эффективность применения системы адаптивной отказоустойчивости при разделении решаемых задач на равные части (такты активной защиты).

12. Оцените эффективность применения системы адаптивной отказоустойчивости при разделении решаемых задач на такты случайной длительности.

13. Оцените эффективность активной защиты интерфейсных модулей.

14. Оцените эффективность активной защиты в условиях работы ИС со случайными перерывами.

15. Раскройте содержание синтеза активной защиты в ИС

16. Опишите модели расчетов безотказности и готовности ИС с активной защитой.

17. Какие выводы следуют из анализа безотказности и готовности ИС с различными дисциплинами активной защиты?

18. Объясните преимущества технологии адаптивной отказоустойчивости по сравнению с известными способами резервирования в информационных системах.

ГЛАВА III.4. ОБЕСПЕЧЕНИЕ НАДЕЖНОСТИ ПРОГРАММНЫХ СРЕДСТВ

III.4.1. Жизненный цикл функциональной надежности программных средств

При формировании требований к программному обеспечению (ПО) прежде всего должна быть выбрана модель жизненного цикла обеспечения его функциональной надежности. На рис. III.4.1 приведена широко распространенная в международной практике [55, 56, 57, 79 и т.д.] V-диаграмма жизненного цикла разработки функционально надежного ПО. По этой диаграмме по нисходящей ветви четко прослеживается последовательность формирования требований, архитектуры ПО, последовательность проектирования системы ПО, составных программных модулей и, наконец, кодирование программных модулей.

Анализ требований к системе подразумевает определение ее функциональных возможностей, пользовательских требований, требований к надежности и безопасности, требований к внешним интерфейсам и т. д. Требования к системе оцениваются исходя из критериев реализуемости и возможности проверки при тестировании на основе предварительной оценки рисков. *Анализ требований к программному обеспечению* предполагает определение следующих характеристик для каждого компонента ПО:

• функциональных возможностей, включая характеристики производительности и среды функционирования компонента;

• внешних интерфейсов;

• спецификаций функциональной надежности и безопасности;

• эргономических требований;

• требований к используемым данным;

• требований к установке и приемке;

• требований к пользовательской документации;

• требований к эксплуатации и сопровождению.

Требования к ПО оцениваются исходя из критериев соответствия ими требованиям к системе, реализуемости и возможности проверки при тестировании.

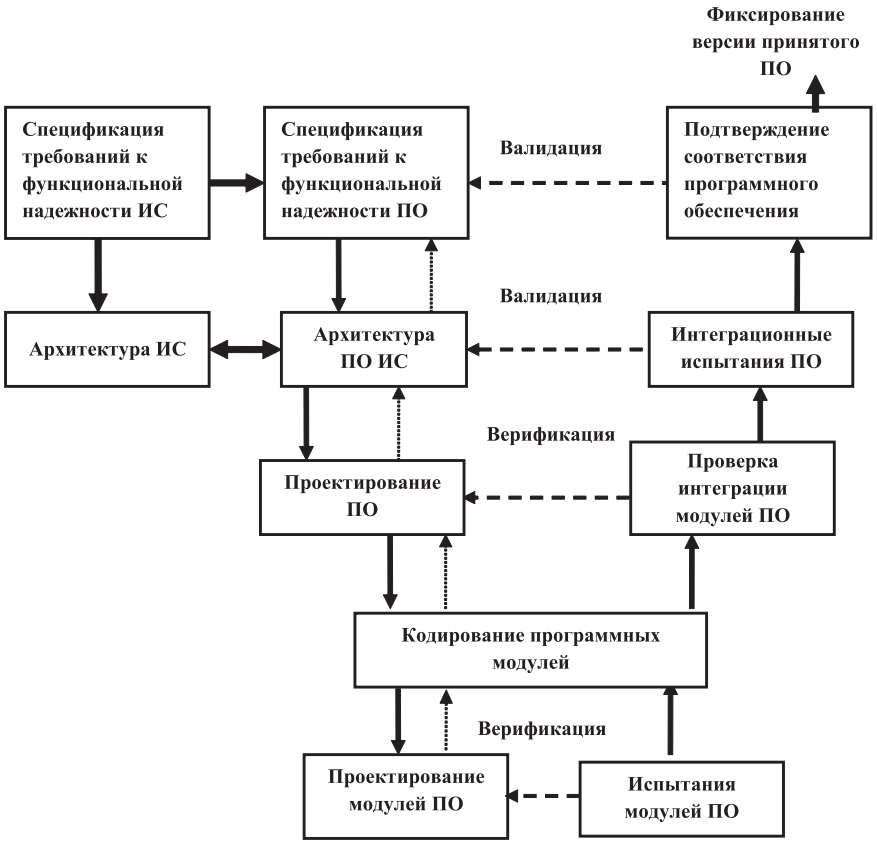


Рис. III.4.1. V-диаграмма жизненного цикла обеспечения функциональной надежности программного обеспечения

Проектирование архитектуры системы на высоком уровне заключается в определении компонентов ее оборудования, ПО и операций, выполняемых эксплуатирующим систему персоналом. Архитектура системы должна соответствовать требованиям, предъявляемым к системе, а также принятым проектным стандартам и методам [54].

Проектирование архитектуры ПО включает следующие задачи (для каждого компонента ПО):

- трансформацию требований к программным средствам в архитектуру, определяющую на высоком уровне структуру программных средств и состав его компонентов;
- разработку и документирование программных интерфейсов программных средств и баз данных;
- разработку предварительной версии пользовательской документации;
- разработку и документирование предварительных требований к тестам и плана интеграции программных средств.

Архитектура компонентов программных средств должна соответствовать требованиям, предъявляемым к ним, а также принятым проектным стандартам и методам.

Детальное проектирование программных средств включает следующие задачи:

- описание компонентов программных средств и интерфейсов между ними на более низком уровне, достаточном для их последующего самостоятельного кодирования и тестирования;
- разработку и документирование детального проекта базы данных;
- обновление (при необходимости) пользовательской документации;
- разработку и документирование требований к тестам и плана тестирования компонентов программных средств;
- обновление плана интеграции программных средств.

Кодирование и тестирование программных средств охватывают следующие задачи:

• разработку (кодирование) и документирование каждого компонента программных средств и базы данных, а также совокупности тестовых процедур и данных для их тестирования;

• тестирование каждого компонента программных средств и базы данных на соответствие предъявляемым к ним требованиям. Результаты тестирования компонентов должны быть документированы;

• обновление (при необходимости) пользовательской документации;

• обновление плана интеграции ПО.

Кодирование может производиться либо с помощью графической среды, библиотеки программных модулей, либо с помощью языков высокого уровня, либо с помощью ассемблера. Затем, следуя восходящей ветви V-диаграммы, необходимо провести испытания и связанную с ними верификацию всех разработанных программных модулей.

Под верификацией программного средства подразумеваются действия по определению того, что выходные данные каждой фазы жизненного цикла отвечают требованиям предыдущей фазы. Эти действия должны осуществляться посредством анализа и испытаний. Процесс верификации (verification process) состоит в определении того, что программные продукты, являющиеся результатами некоторого действия, полностью удовлетворяют требованиям или условиям, обусловленным предшествующими действиями *(верификация* в «узком» смысле означает формальное доказательство правильности программных средств). Верификации должны подвергаться как отдельные программные модули, так и группы модулей и система ПО в целом. Верификация может проводиться с различными степенями независимости. Степень независимости может ва-

рьироваться от выполнения верификации самим исполнителем или другим специалистом данной организации до ее выполнения специалистом другой организации с различными вариациями. Если процесс верификации осуществляется организацией, не зависящей от поставщика, разработчика, оператора или службы сопровождения, то он называется *процессом независимой верификации.*

В процессе верификации проверяются следующие условия:

• непротиворечивость требований к системе и степень учета потребностей пользователей;

• возможности поставщика выполнить заданные требования;

• соответствие выбранных процессов жизненного цикла программных средств условиям договора;

• адекватность стандартов, процедур и среды разработки процессам жизненного цикла программных средств;

• соответствие проектных спецификаций программных средств заданным требованиям;

• корректность описания в проектных спецификациях входных и выходных данных, последовательности событий, интерфейсов, логики и т.д.;

• соответствие кода проектным спецификациям и требованиям;

• тестируемость и корректность кода, его соответствие принятым стандартам кодирования;

• корректность интеграции компонентов программных средств в систему;

• адекватность, полнота и непротиворечивость документации.

На уровне групп модулей и системы должны проводиться так называемые интеграционные испытания, которые предназначены для комплексной проверки совместимости правильного взаимодействия программных и аппаратных средств системы.

Интеграция программных средств предусматривает сборку разработанных компонентов программных средств в соответствии с планом интеграции и тестирование агрегированных компонентов. Для каждого из агрегированных компонентов разрабатываются наборы тестов и тестовые процедуры, предназначенные для проверки каждого из квалификационных требований при последующем квалификационном тестировании. *Квалификационное требование* – это набор критериев или условий, которые необходимо выполнить, чтобы квалифицировать программный продукт как соответствующий своим спецификациям и готовый к использованию в условиях эксплуатации.

Интеграция системы заключается в сборке всех ее компонентов, включая программные средства и оборудование. После интеграции система, в свою очередь, подвергается *квалификационному тестированию* на соответствие совокупности требований к ней. При этом также производятся оформление и проверка полного комплекта документации на систему.

Завершающая стадия разработки – это ***аттестация (валидация) программного обеспечения***, под которой понимаются процедуры получения доказательственной базы для подтверждения соответствия программного обеспечения заданным требованиям по функциональной надежности. *Процесс валидации* (validation process) предусматривает определение полноты соответствия заданных требований и созданной системы или программного продукта их конкретному функциональному назначению. Под *аттестацией* обычно понимаются подтверждение и оценка достоверности проведенного тестирования ПО. Валидация должна гарантировать полное соответствие ПО спецификациям, требованиям и документации, а также возможность его безопасного и надежного применения пользователем. Валидацию рекомендуется выполнять путем тестирования во всех возможных ситуациях и использовать при этом независимых спе-

циалистов. Валидация может проводиться на начальных стадиях жизненного цикла программных средств или как часть работы по приемке ПО. Валидация, так же как и верификация, может осуществляться с различными степенями независимости. Если процесс валидации выполняется организацией, не зависящей от поставщика, разработчика, оператора или службы сопровождения, то он называется *процессом независимой валидации.*

Затем производится установка и приемка ПО.

Установка ПО осуществляется разработчиком в соответствии с планом в той среде и на том оборудовании, которые предусмотрены договором. В процессе установки проверяется работоспособность программных средств и баз данных. Если устанавливаемое ПО заменяет существующую систему, разработчик должен обеспечить их параллельное функционирование в соответствии с договором.

Приемка ПО предусматривает оценку результатов квалификационного тестирования ПС и системы и документирование результатов оценки, которые проводятся заказчиком с помощью разработчика. Разработчик выполняет окончательную передачу ПС заказчику в соответствии с договором, обеспечивая при этом необходимое обучение и поддержку.

Модель жизненного цикла разработки функционально надежного ПО должна быть подробно описана в плане обеспечения качества программного обеспечения. Процедуры обеспечения качества должны выполняться параллельно с другими работами в течение всего жизненного цикла программного обеспечения. Весь объем работ, осуществляемый в течение одной из фаз жизненного цикла, должен быть определен до начала этой фазы. Каждая фаза жизненного цикла программного обеспечения должна быть разделена на элементарные задачи с хорошо определенным входными и выходными данными и видом работ с ними. В плане обеспечения качества ПО должны быть

описаны и документированы все требуемые верификационные (проверочные) процедуры и отчеты.

Каждый документ программного обеспечения должен быть написан в соответствии со следующими правилами:

• содержать или выполнять все применимые условия и требования предшествующего документа, с которым он имеет иерархическую связь;

• не противоречить предшествующему документу;

• каждый термин, акроним и аббревиатура должны иметь одно и то же значение во всех документах;

• на каждый пункт или понятие нужно ссылаться в каждом документе по одному и тому же имени или описанию.

III.4.2. Правила и этапы построения надежных программных средств

III.4.2.1. Характерные недостатки программных средств

Как покупные, так и собственной разработки программные средства (ПС) имеют ряд характерных недостатков. К ним относятся следующие:

• во многих случаях отсутствует единая техническая политика в создании ПС – недостаточно проработана архитектура ПС, алгоритмы плохо сопряжены между собой;

• ПС часто формируется из модулей, созданных применительно к другим системам и написанных на разных языках программирования;

• при разработке нередко используются программные модули неопределенного происхождения, что является одним из существенных источников нарушения функциональной надежности ПС;

• технические проблемы исполнения программ показывают недостатки операционной среды – как правило, применяются собственные «кустарные» операционные системы;

• покупные программные составляющие отечественного производства, как правило, не сертифицированы и недостаточно качественны;

• для нештатных ситуаций программное средство имеет недостаточно средств диагностики и поддержки пользователя;

• в большинстве разработок нет технического задания на программное обеспечение;

• низкий технический уровень документов (часто не пригодные инструкции по инсталляции и по эксплуатации программ, в документации много противоречий, в том числе между документами в твердой копии и на диске и т.д.) и др., в результате чего эксплуатация и модификация ПС возможна только конкретным его разработчиком.

III.4.2.2. Маршрутная карта функциональной надежности программных средств

Для создания функционально надежных ПС, лишенных указанных выше недостатков, должны решаются следующие первоочередные задачи:

- формирование квалифицированных и организованных команд программистов;

- применение рекомендованных международным сообществом технологий проектирования и хорошо зарекомендовавших себя методов и технических приемов;

- применение сертификационных испытаний в целях как оценки соответствия, так и устранения выявленных недостатков.

Стандартами [54, 55, 56 и др.] определена маршрутная карта безопасности программного обеспечения. В ней исчерпывающим образом указаны все этапы жизненного цикла безопасности ПС. Применительно к функциональной надежности эта маршрутная карта интерпретируется следующим образом (рис. III.4.2).

Для обеспечения приемлемого уровня безошибочности ПС целесообразно руководствоваться следующими правилами:

- при разработке ПС применять методы нисходящего проектирования;
- обеспечивать модульность программ;
- осуществлять проверку каждой фазы жизненного цикла разработки программ;
- стремиться использовать проверенные модули и библиотеки модулей;
- разрабатывать понятную документацию;
- создавать документы, которые доступны аудиту;
- проводить аттестационные испытания.

В соответствии с маршрутной картой и указанными правилами должны быть выполнены следующие шаги в обеспечении функциональной надежности и безопасности ПС:

1. Разработка спецификации требований к программному обеспечению.
2. Разработка архитектуры программного обеспечения.
3. Создание проекта программного обеспечения и его реализация.
4. Верификация программного обеспечения.
5. Интеграция программного обеспечения/аппаратных средств.
6. Аттестация программного обеспечения.

III.4.3. Технология разработки надежных программных средств

III.4.3.1. Рекомендации по разработке спецификации требований

При разработке спецификации требований настоятельно рекомендуется применение структурной методологии, а также формальных и полуформальных методов. Существует ряд структурных

Рис. III.4.2. Маршрутная карта функциональной надежности программного обеспечения

методологий. Для информационно-управляющих систем наиболее интересны методологии MASCOT, JSD, Real-time Yourdon, которые ориентированы на управление производственным процессом и применение в реальном времени (что особенно важно для безопасности критически важных приложений).

Структурированные методы по существу – «думающие инструменты», целью которых является систематическое понимание проблемы и разбиение ее на составные части. Их главные особенности следующие:

- логический порядок мышления, разбивающий большую проблему на управляемые стадии;
- идентификация полной системы, включая окружающую среду, а также необходимую систему;
- разложение данных и функций в требуемой системе.

Формальные методы обеспечивают средства разработки описания системы на некоторой стадии разработки ее спецификации, проекта или кода. Получающееся описание принимает математическую форму и может быть подвергнуто математическому анализу, чтобы обнаружить различные классы противоречивости или некорректности. Кроме того, описание может в некоторых случаях быть проанализировано машиной с точностью, подобной проверке синтаксиса исходной программы посредством компилятора. Формальный метод в общем случае предлагает систему обозначений (обычно некоторую форму используемой дискретной математики), а также технический прием для получения описания в предлагаемой системе обозначений. Формальный метод предлагает различные формы анализа для того, чтобы проверить описание для различных свойств корректности. К формальным методам относятся: CSP, OBJ, LOTOS и др.

Например, с помощью метода CSP система может быть смоделирована посредством составляющих ее последовательных или параллельных независимых процессов. Процессы могут обме-

ниваться сообщениями (синхронизироваться или обмениваться данными) через каналы связи. Обмен данными имеет место только тогда, когда оба процесса готовы. Относительный выбор времени событий также может быть смоделирован. Метод CSP обеспечивает язык для спецификации систем процессов и доказательство того, что выполнение процессов удовлетворяет их спецификации (описанная как трассировка разрешенная последовательность событий). Метод OBJ, в свою очередь, является алгебраическим языком спецификации.

Пользователи определяют требования в терминах алгебраического уравнения. Спецификация OBJ и последующее ее пошаговое выполнение поддается одним и тем же формальным методам доказательства, как и другие формальные подходы. Кроме того, так как конструктивные аспекты спецификации OBJ являются выполнимыми на аппаратных средствах, можно аттестацию системы выполнять непосредственно из спецификации. Такая выполнимость позволяет конечным пользователям предусмотренной системы получать представление о возможностях системы на стадии разработки спецификации системы. Метод OBJ применим только к последовательным системам или к последовательным аспектам параллельных систем. OBJ широко используется для спецификации как малых так и крупномасштабных промышленных приложений.

III.4.3.2. Технология разработки архитектуры надежной программы

При разработке архитектуры применяется основной набор методов и технических приемов обеспечения функциональной надежности программ.

Все принципы и методы обеспечения надежности в соответствии с их целью можно разбить на четыре группы: ***предупреждение ошибок, обнаружение ошибок, исправление ошибок*** и

обеспечение устойчивости к ошибкам (рис. III.4.3). К первой группе относятся принципы и методы, позволяющие минимизировать или вообще исключить ошибки. Методы второй группы сосредоточивают внимание на функциях самого программного обеспечения, помогающих выявлять ошибки. К третьей группе относятся функции программного обеспечения, предназначенные для исправления ошибок или их последствий. Устойчивость к ошибкам (четвертая группа) – это мера способности системы программного обеспечения продолжать функционирование при наличии ошибок.

Предупреждение ошибок

К этой группе относятся принципы и методы, цель которых – не допустить появления ошибок в готовой программе. Должно быть очевидно, что предупреждение ошибок – оптимальный путь к достижению надежности программного обеспечения. Лучший способ обеспечить надежность – прежде всего не допустить возникновения ошибок. С этой целью широко применяют способы так называемого *защитного (безопасного) програм-*

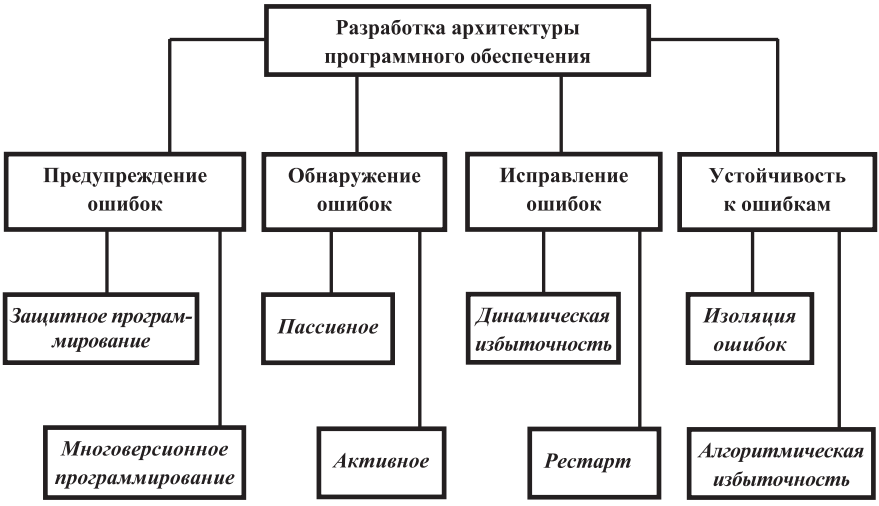


Рис. III.4.3. Методы и способы построения архитектуры надежного программного обеспечения

мирования, направленного на уменьшение вероятности ошибок в программах, а также *многоверсионное программирование.*

Защитное программирование

Защитное программирование опирается на две основные концепции: защиту и устойчивость к ошибкам.

Под защитой в рассматриваемом аспекте понимают ограничение неправильного использования программных объектов. Другими словами, выдвигается требование проектировать и программировать таким образом, чтобы не только гарантировать ожидаемое использование программы в строгом соответствии со спецификациями, но и сделать невозможным ее неправильное использование. Например, при проектировании системы, в которой взаимодействует много модулей, мы можем потребовать, чтобы какие-то взаимодействия между ними разрешались лишь в определенных ситуациях. Таким образом, вызов модулем *A* модуля *B* может быть разрешен всегда, в определенных ситуациях или же никогда, несмотря на то, что модули *A* и *B* находятся в таких отношениях, что вызов возможен.

Защитное программирование – это технология предупреждения потенциальных ошибок путем проверки в каждом модуле множества допустимых условий. Во время программирования может быть использовано много технических приемов, чтобы проверить аномалии в управлении или в данных. Ниже перечислены некоторые из методов безопасного программирования:

- анализируются граничные значения переменных;
- проверяются размеры, тип и диапазон параметров процедуры ввода данных;
- параметры «только чтение» и «чтение-запись» должны быть разделены, и должен быть проверен доступ к ним;
- символьные константы не должны быть доступны для записи;

- входные переменные и промежуточные (вспомогательные) переменные с физическим значением должны быть проверены на предмет достоверности;

- воздействие выходных переменных должно быть проверено, предпочтительно непосредственным наблюдением связанных с ними изменений состояния системы и т.д.

Простейший метод защитного программирования заключается в использовании специальных ловушек ошибок, рассчитанных на ошибки типа неправильного использования модулей. Разработчика программы не интересует, что будет делать пользователь после того, как получит сообщение о неправильном использовании модуля, но при этом он обязан спроектировать модуль так, чтобы ошибки пользователя не вызывали необратимых изменений в модуле. Таким образом, пользователь, пойманный на неправильном употреблении модуля, принимает корректирующие действия и снова вызывает модуль, не оставляя никаких следов ошибочных вызовов. Иначе говоря, речь идет о таком программировании, когда программный продукт очень трудно или невозможно использовать за пределами области действия его спецификации.

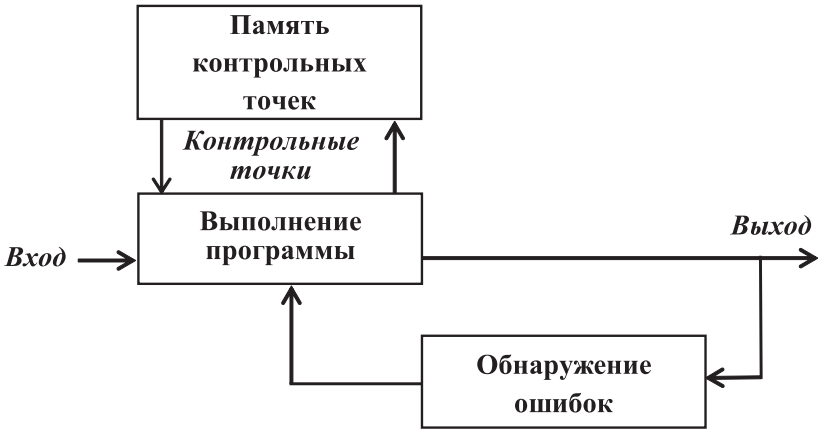


Рисунок III.4.4. Контрольная точка и перезапуск.

К технологии защитного программирования следует отнести и одноверсионное программирование с контрольными точками и перезапусками, а также с парными прогонами [19]. Одноверсионное программирование – это стиль программирования, при котором по спецификации создается ровно одна версия программы.

Рассмотрим **метод контрольной точки и перезапуска.** На вход программы подается некое входное значение, а далее, во время выполнения, программа через определенные интервалы времени создает контрольные точки и проверяет, корректны ли вычисления. Если же на каком-то этапе вычисления становятся некорректными, то программа возвращается к предыдущей контрольной точке и продолжает вычисления с нее (рис. III.4.4). Данный метод эффективен для устранения временных неисправностей, поскольку при ошибках, вызванных такими неисправностями, программа будет каждый раз возвращаться к последней контрольной точке до тех пор, пока неисправность не исчезнет, тем самым ошибка будет устранена.

Метод парных прогонов – усовершенствование метода контрольной точки и перезапуска. Пусть есть две идентичные

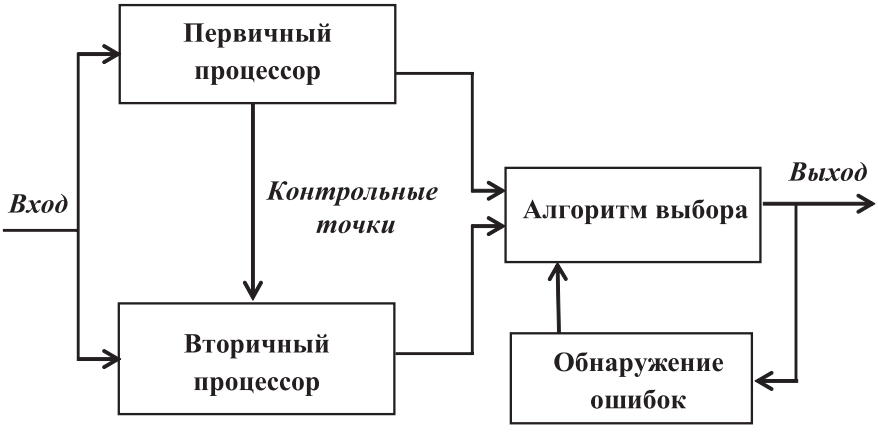


Рис. III.4.5. Парные прогоны

версии программы, запускаемой на двух разных процессорах. Тогда на основном процессоре программа выполняется, и там же создаются контрольные точки и передаются вторичному процессору до тех пор, пока не обнаруживается ошибка. После обнаружения ошибки вторичный процессор загружает последнюю контрольную точку и берет на себя роль первичного процессора (рис. III.4.5).

Данный метод эффективен для устранения временных неисправностей в ПО (по тем же причинам, что и в предыдущем методе) и постоянных – в аппаратуре, поскольку, в случае ошибки, вызванной постоянной неисправностью первичного процессора, программа будет выполняться с последней контрольной точки на вторичном процессоре, тем самым ошибка будет устранена.

Однако в силу специфики информационных систем реального времени, описанные методы устранения ошибок и одноверсионное программирование в целом недостаточно эффективны.

Многоверсионное программирование

Цель многоверсионного программирования состоит в обнаружении и маскировке остаточных ошибок проекта программного обеспечения в течение выполнения программ, чтобы предотвратить критически опасные отказы системы и продолжить работу с высокой надежностью. В многоверсионном программировании данная спецификация программы реализуется по-разному N раз. *Многоверсионное программирование – стиль программирования, при котором по единой спецификации независимо создаются несколько различных версий программы.*

При многоверсионном программировании применяются следующие основные методы устранения ошибок [18 – 21 и др.].

При ***N*-версионном программировании** значения входных данных задаются *N* версиям, и результаты, произведенные *N* версиями, сравниваются с помощью алгоритма выбора (рис. III.4.6).

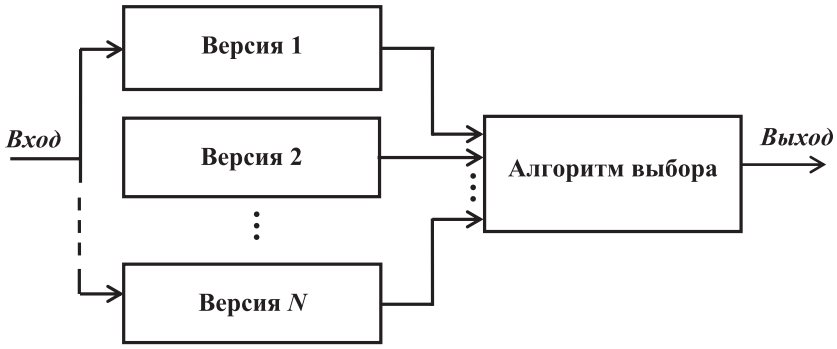


Рис. III.4.6. N-версионное программирование

Если результат считается достоверным, то он передается компьютеру в качестве выходных данных. *N* версий могут выполняться параллельно на отдельных компьютерах, альтернативно все версии могут выполняться на одном компьютере. Выходные результаты подвергаются внутреннему голосованию. Различные стратегии голосования могут быть использованы на *N* версиях в зависимости от требований применения.

При дуальном программировании (если разрабатываются две версии программы) в случае обнаружения расхождения в результатах необходимо определить по дополнительным критериям, какой результат правильный и отбросить другой результат. При *N*-версионном программировании правильный результат определяется по мажоритарному признаку, т.е. выбирается тот результат, который наблюдается в большинстве вариантов программы.

Практическая реализация многоверсионного программирования возможна следующими основными способами (рис. III.4.7).

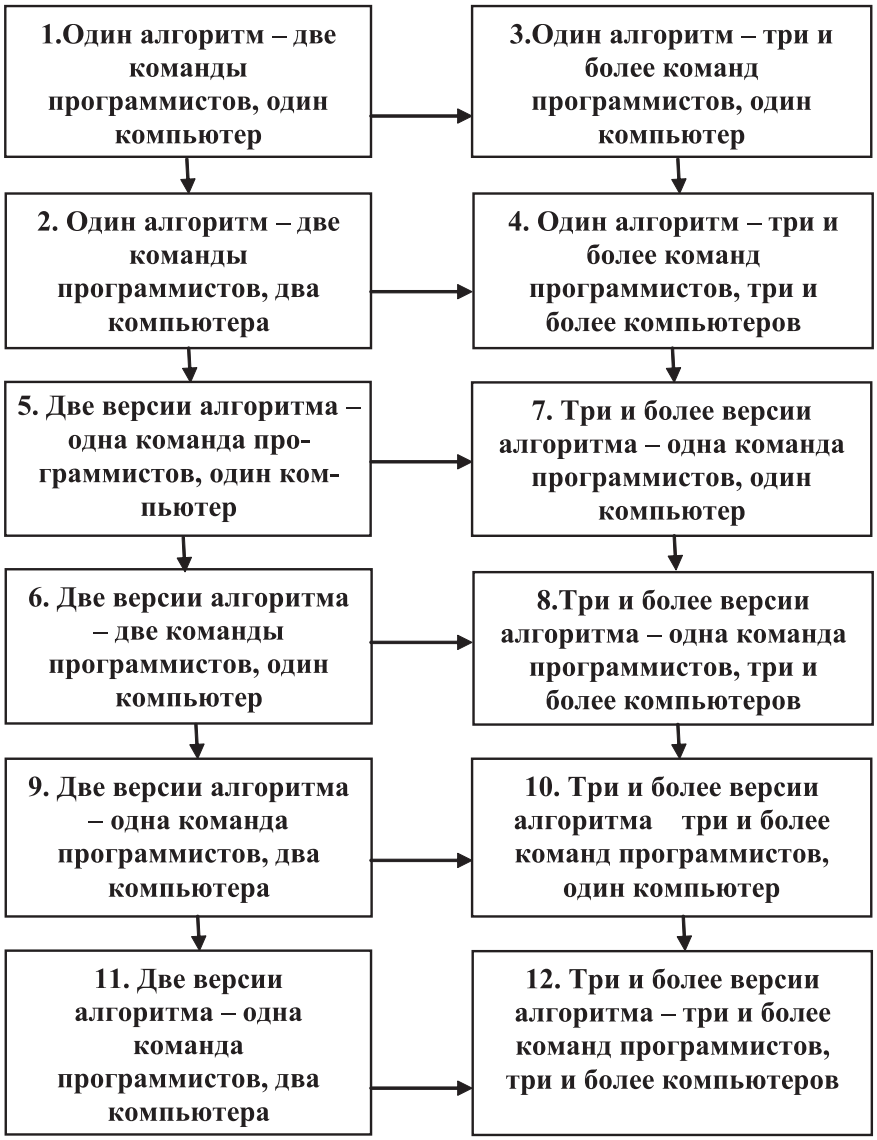


Рис. III.4.7. Способы реализации многоверсионного программирования

Для критически важных информационных систем необходимо полное согласие всех *N* версий. Для других информационных систем может быть использована стратегия голосования

путем простого большинства. Для случаев, где не имеется коллективного согласия, могут быть использованы вероятностные подходы, чтобы максимизировать шанс выбора правильного значения, например, взяв среднее значение, временно заморозив выходные данные до возвращения согласия и т.д.

Наименее затратным является способ № 1 (рис. III.4.7). Он основывается на реализации одного алгоритма задачи двумя группами программистов (или просто двумя программистами). Обе группы создают две версии программы, которые последовательно выполняются на одном вычислительном модуле. Результаты сравниваются с помощью безопасного компаратора или безопасной программы. Термин «безопасный» здесь применяются в том смысле, что компаратор или программа сравнения результатов сами не должны привносить ошибки. С этой целью разрабатываются специальные компараторы.

Реализация двух версий программы на двух вычислительных модулях обеспечивает более надежную работу программы (способ № 2). При реализации двух версий программы на одном модуле возможна ситуация, когда отказ модуля (особенно скрытый отказ) может привести к одинаковым ошибочным результатам при выполнении обеих версий программы. При сравнении результатов эта ошибка не будет обнаружена. Применение двух вычислительных модулей позволяет также практически исключить зависимость реализации программы от состояния надежности модуля, поскольку в случае отказа одного любого модуля на другом исправном модуле реализуется вторая версия программы.

Реально применение такого *N*-версионного программирования, при котором создаются две версии алгоритма. По этим версиям алгоритма три команды программистов строят от трех до шести версий программы. Это позволяет на основе метода мажоритарного резервирования не только обнаруживать, но и

«маскировать», т.е. изолировать ошибочные результаты и выдавать правильные результаты выполнения программы.

Рассмотренные способы резервирования требуют в 2 или *N* раз больше времени для вычислений и увеличение объема труда программистов во столько же раз. В связи с этим представляет интерес модифицированное дуальное программирование при котором, наряду с достаточно точной, но сложной основной программой, используется менее точная, но простая резервная программа. Если при одинаковых исходных данных результаты работы программ отличаются на величину, большую, чем допустимая погрешность, делается предположение, что отказала основная программа, как менее надежная, и в качестве правильного результата принимается результат работы резервной программы.

Гарантировать отсутствие ошибок, однако, невозможно никогда. Другие три группы методов опираются на предположение, что ошибки все-таки будут. Здесь основная нагрузка ложится на алгоритм выбора. В качестве такого алгоритма можно принять один из отказоустойчивых алгоритмов, описанных в [19]. Данный метод эффективен для устранения неисправностей в программном обеспечении, поскольку различные временные и постоянные неисправности, вызывающие ошибки в разных версиях программы, будут устранены благодаря корректному выполнению остальных версий программ.

Идея метода **программирования с восстановлением блоками** – использование *N*-версионного программирования и механизма контрольной точки и перезапуска (рис. III.4.8). Суть метода в следующем. Запускается основная версия программы, которая выполняется и создает контрольные точки до тех пор, пока не обнаруживается ошибка. После обнаружения ошибки запускается альтернативная версия программы, которая начинает выполнение программы с последней контрольной точки, и т.д. Данный метод

сочетает в себе достоинства методов *N*-версионного программирования и контрольной точки и перезапуска.

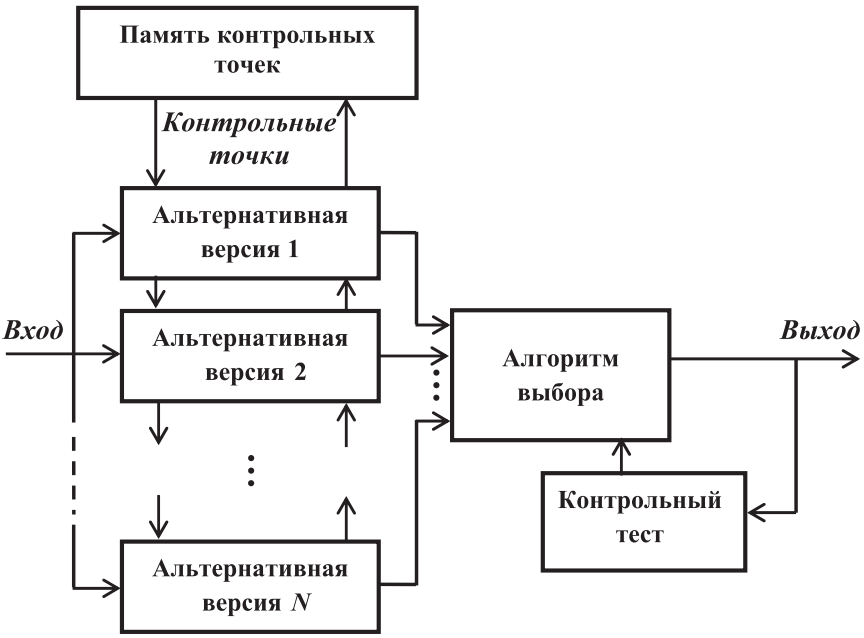


Рис. III.4.8. Метод восстановления блоками

***N*-самотестируемое программирование.** В зависимости от механизма обнаружения ошибки метод может быть двух типов – с использованием контрольных тестов и сравнений. Рассмотрим *N*-самотестируемое программирование с использованием контрольных тестов. В этом подходе собраны преимущества методов *N*-версионного программирования и восстановления блоками. Из единой спецификации независимо создаются различные версии программ и контрольных тестов. Главное отличие от метода восстановления блоками – использование для каждой версии программы своего контрольного теста. *N* версий программы запускаются последовательно или параллельно, затем каждая запущенная версия проходит контрольный тест, а

затем в блоке логического выбора в качестве результирующего значения берется значение наивысшей из версий, прошедшей контрольный тест (рис. III.4.9).

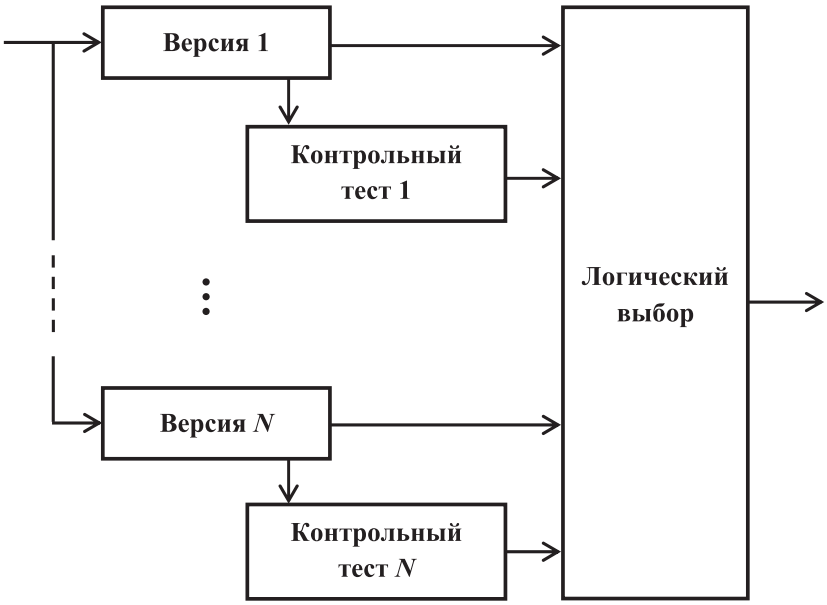


Рис. III.4.9. N-самотестируемое программирование

Стоит заметить, что именно многоверсионное программирование и методы устранения ошибок, описанные выше – основные инструменты обеспечения отказоустойчивости для информационных систем реального времени. Они активно применяются в различных системах реального времени, например, в бортовых вычислительных системах современных американских аэробусов.

Обнаружение ошибок

Если предполагать, что в программном обеспечении какие-то ошибки все же будут, то лучшая (после предупреждения ошибок) стратегия – включить средства обнаружения ошибок в само программное обеспечение.

Большинство методов направлено, по возможности, на незамедлительное обнаружение сбоев. Немедленное обнаружение имеет два преимущества: можно минимизировать влияние ошибки и последующие затруднения для человека, которому придется извлекать информацию о ней, находить ее и исправлять.

Меры по обнаружению ошибок можно разбить на две подгруппы: *пассивные* попытки обнаружить симптомы ошибки в процессе «обычной» работы программного обеспечения и *активные* попытки программной системы периодически обследовать свое состояние в поисках признаков ошибок.

Пассивное обнаружение. Меры по обнаружению ошибок могут быть приняты на нескольких структурных уровнях программной системы. Здесь мы будем рассматривать уровень подсистем, или компонентов, т.е. нас будут интересовать меры по обнаружению симптомов ошибок, предпринимаемые при переходе от одного компонента к другому, а также внутри компонента. Все это, конечно, применимо также к отдельным модулям внутри компонента.

Разрабатывая эти меры, мы будем опираться на следующее.

1. *Взаимное недоверие.* Каждый из компонентов должен предполагать, что все другие содержат ошибки. Когда он получает какие-нибудь данные от другого компонента или из источника вне системы, он должен предполагать, что данные могут быть неправильными, и пытаться найти в них ошибки.

2. *Немедленное обнаружение.* Ошибки необходимо обнаружить как можно раньше. Это не только ограничивает наносимый ими ущерб, но и значительно упрощает задачу отладки.

3. *Избыточность.* Все средства обнаружения ошибок основаны на некоторой форме избыточности (явной или неявной).

Когда разрабатываются меры по обнаружению ошибок, важно принять согласованную стратегию для всей системы. Действия, предпринимаемые после обнаружения ошибки в программном

обеспечении, должны быть единообразными для всех компонентов системы. Это ставит вопрос о том, какие именно действия следует предпринять, когда ошибка обнаружена. Наилучшее решение – немедленно завершить выполнение программы или (в случае операционной системы) перевести центральный процессор в состояние ожидания. С точки зрения предоставления человеку, отлаживающему программу, например системному программисту, самых благоприятных условий для диагностики ошибок, немедленное завершение представляется наилучшей стратегией. Конечно, во многих системах подобная стратегия бывает нецелесообразной (например, может оказаться, что приостанавливать работу системы нельзя). В таком случае используется метод *регистрации ошибок*. Описание симптомов ошибки и «моментальный снимок» состояния системы сохраняются во внешнем файле, после чего система может продолжать работу. Этот файл позднее будет изучен обслуживающим персоналом.

Всегда, когда это возможно, лучше приостановить выполнение программы, чем регистрировать ошибки (либо обеспечить как дополнительную возможность работу системы в любом из этих режимов). Различие между этими методами проиллюстрируем на способах выявления причин возникающего иногда скрежета Вашего автомобиля. Если автомеханик находится на заднем сиденье, то он может обследовать состояние машины в тот момент, когда скрежет возникает. Если вы выбираете метод регистрации ошибок, задача диагностики станет сложнее.

Активное обнаружение ошибок. Не все ошибки можно выявить пассивными методами, поскольку эти методы обнаруживают ошибку лишь тогда, когда ее симптомы подвергаются соответствующей проверке. Можно делать и дополнительные проверки, если спроектировать специальные программные средства для активного поиска признаков ошибок в системе. Такие средства называются *средствами активного обнаружения ошибок*.

Активные средства обнаружения ошибок обычно объединяются в *диагностический монитор* – параллельный процесс, который периодически анализирует состояние системы с целью обнаружить ошибку. Большие программные системы, управляющие ресурсами, часто содержат ошибки, приводящие к потере ресурсов на длительное время. Например, управление памятью операционной системы сдает блоки памяти «в аренду» программам пользователей и другим частям операционной системы. Ошибка в этих самых «других частях» системы может иногда вести к неправильной работе блока управления памятью, занимающегося возвратом сданной ранее в аренду памяти, что вызывает медленное вырождение системы.

Диагностический монитор можно реализовать как периодически выполняемую задачу (например, она планируется на каждый час) либо как задачу с низким приоритетом, которая планируется для выполнения в то время, когда система переходит в состояние ожидания. Как и прежде, выполняемые монитором конкретные проверки зависят от специфики системы, но некоторые идеи будут понятны из примеров. Монитор может обследовать основную память, чтобы обнаружить блоки памяти, не выделенные ни одной из выполняемых задач и не включенные в системный список свободной памяти. Он может проверять также необычные ситуации: например, процесс не планировался для выполнения в течение некоторого разумного интервала времени. Монитор может осуществлять поиск «затерявшихся» внутри системы сообщений или операций ввода-вывода, которые необычно долгое время остаются незавершенными, участков памяти на диске, которые не помечены как выделенные и не включены в список свободной памяти, а также различного рода странностей в файлах данных.

Иногда желательно, чтобы в чрезвычайных обстоятельствах монитор выполнял диагностические тесты системы. Он может

вызывать определенные системные функции, сравнивая их результат с заранее определенным и проверяя, насколько разумно время выполнения. Монитор может также периодически предъявлять системе «пустые» или «легкие» задания, чтобы убедиться, что система функционирует хотя бы самым примитивным образом.

Исправление ошибок

Следующий шаг – методы исправления ошибок; после того как ошибка обнаружена, либо она сама, либо ее последствия должны быть исправлены программным обеспечением. Исправление ошибок самой системой – плодотворный метод проектирования надежных систем аппаратного обеспечения. Некоторые устройства способны обнаружить неисправные компоненты и перейти к использованию идентичных запасных. Аналогичные методы неприменимы к программному обеспечению вследствие глубоких внутренних различий между сбоями аппаратуры и ошибками в программах. Если некоторый программный модуль содержит ошибку, идентичные «запасные» модули также будут содержать ту же ошибку.

Другой подход к исправлению связан с попытками восстановить разрушения, вызванные ошибками, например искажения записей в базе данных или управляющих таблицах системы. Польза от методов борьбы с искажениями ограничена, поскольку предполагается, что разработчик заранее предугадает несколько возможных типов искажений и предусмотрит программно реализуемые функции для их устранения. Это похоже на парадокс, поскольку, если знать заранее, какие ошибки возникнут, можно было бы принять дополнительные меры по их предупреждению. Если методы ликвидации последствий сбоев не могут быть обобщены для работы со многими типами искажений, лучше всего направлять силы и средства на предупрежде-

ние ошибок. Вместо того, чтобы, разрабатывая операционную систему, оснащать ее средствами обнаружения и восстановления цепочки искаженных таблиц или управляющих блоков, следовало бы лучше спроектировать систему так, чтобы только один модуль имел доступ к этой цепочке, а затем настойчиво пытаться убедиться в правильности этого модуля.

Устойчивость к ошибкам

Основное допущение программирования, ***устойчивого к программным ошибкам*,** заключается в том, что как бы хорошо ни была спроектирована и реализована программа, в ней обязательно будет содержаться несколько остаточных ошибок. А раз так, то модули программы, которые могут дать сбой, должны иметь «резервный запас». С этой целью модуль проектируется в виде *блоков восстановления*. Каждый блок восстановления содержит пропускной тест и один или несколько вариантов реализации. Основной вариант инициируется при вызове блока восстановления и, когда его выполнение завершается, происходит проверка значения пропускного теста. Если он дает «истину», то считается, что выполнение блока восстановления успешно завершено. Если же тест дает «ложь», то инициируется другой вариант, за которым следует определение значения пропускного теста и т. д., и так до успешного выполнения блока восстановления. Если же ни один вариант не прошел пропускного теста пропускной тест, то блок восстановления рассматривается как ошибочный и начинается исполнение другого варианта вызываемого модуля. Применяется также и другой технический прием: написаны, часто независимо, несколько различных сегментов программы, каждый из которых предназначен для выполнения одной функции. Программа строится из этих сегментов. Первый сегмент, называемый первичным, выполняется первым. За ним следует приемочное испытание результата вычислений перво-

го сегмента. Если испытание прошло успешно, тогда результат принимается и передается к последующим частям системы. Если испытание было неудачным, любые побочные эффекты первого сегмента сбрасываются и выполняется второй сегмент, называемый первый альтернативный. За ним также следует приемочное испытание, результаты которого рассматриваются как в первом случае. Если необходимо, могут быть реализованы другие альтернативные приемы.

Приведенный способ парирования остаточных ошибок в программе путем проектирования модуля в виде блока восстановления требует порой неоправданных усилий, связанных с проектированием нескольких блоков восстановления, каждый из которых содержит пропускной тест и один или несколько вариантов реализации. В целях практического обеспечения функционирования программной системы при наличии в ней ошибок разработана группа методов, которая разбивается на четыре подгруппы: *динамическая избыточность, методы отступления, методы изоляции ошибок и построение алгоритмов нечувствительных (или не критичных) к различного рода нарушениям информационного процесса (использование алгоритмической избыточности).*

1. Истоки концепции ***динамической избыточности*** лежат в проектировании аппаратного обеспечения. Один из подходов к динамической избыточности – *мажоритарное резервирование* (метод голосования). Данные обрабатываются независимо несколькими идентичными устройствами и результаты сравниваются. Если большинство устройств выработало одинаковый результат, этот результат и считается правильным. И опять, вследствие особой природы ошибок в программном обеспечении, ошибка, имеющаяся в копии программного модуля, будет также присутствовать во всех других его копиях, поэтому идея голосования здесь, видимо, неприемлема. Предлагаемый иногда подход к решению

этой проблемы состоит в том, чтобы иметь несколько неидентичных копий модуля. Это значит, что все копии выполняют одну и ту же функцию, но либо реализуют различные алгоритмы, либо созданы разными разработчиками. Этот подход бесперспективен по следующим причинам. Часто трудно получить существенно различные версии модуля, выполняющие одинаковые функции. Кроме того, возникает необходимость в дополнительном программном обеспечении для организации выполнения этих версий параллельно или последовательно и сравнения результатов. Это дополнительное программное обеспечение повышает уровень сложности системы, что, конечно, противоречит основной идее предупреждения ошибок – стремиться в первую очередь минимизировать сложность.

Второй подход к динамической избыточности – выполнять эти запасные копии только тогда, когда результаты, полученные с помощью основной копии, признаны неправильными. Если это происходит, система автоматически вызывает запасную копию. Если и ее результаты неправильны, вызывается другая запасная копия и т. д.

2. Вторая подгруппа методов обеспечения устойчивости к ошибкам называется ***методами отступления*** или сокращенного обслуживания. Эти методы приемлемы обычно лишь тогда, когда для системы программного обеспечения существенно важно корректно закончить работу. Например, если ошибка оказывается в системе, управляющей технологическими процессами, и в результате эта система выходит из строя, то может быть загружен и выполнен особый фрагмент программы, призванный подстраховать систему и обеспечить безаварийное завершение всех управляемых системой процессов. Аналогичные средства часто необходимы в операционных системах. Если операционная система обнаруживает, что вот-вот выйдет из строя, она может загрузить аварийный фрагмент, ответственный за опове-

щение пользователей у терминалов о предстоящем сбое и за сохранение всех критических для системы данных.

3. Третья подгруппа – ***методы изоляции ошибок***. Основная их идея – не дать последствиям ошибки выйти за пределы как можно меньшей части системы программного обеспечения так, чтобы, если ошибка возникнет, то не вся система оказалась неработоспособной; отключаются лишь отдельные функции в системе либо некоторые ее пользователи. Например, во многих операционных системах изолируются ошибки отдельных пользователей, так что сбой влияет лишь на некоторое подмножество пользователей, а система в целом продолжает функционировать. В телефонных переключающих системах для восстановления после ошибки, чтобы не рисковать выходом из строя всей системы, просто разрывают телефонную связь. Другие методы изоляции ошибок связаны с защитой каждой из программ в системе от ошибок других программ. Ошибка в прикладной программе, выполняемой под управлением операционной системы, должна оказывать влияние только на эту программу. Она не должна сказываться на операционной системе или других программах, функционирующих в этой системе.

В информационной системе изоляция программ является ключевым фактором, гарантирующим, что ошибки в программе одного пользователя не приведут к ошибкам в программах других пользователей или к полному выводу системы из строя. Основные правила изоляции ошибок в программах состоят в следующем:

а) прикладная программа не должна иметь возможности непосредственно ссылаться на другую прикладную программу или данные в другой программе и изменять их;

б) прикладная программа не должна иметь возможности непосредственно ссылаться на программы или данные операционной системы и изменять их. Связь между двумя программами

(или программой и операционной системой) может быть разрешена только при условии использования четко определенных сопряжений и только в случае, когда обе программы дают согласие на эту связь;

в) прикладные программы и их данные должны быть защищены от операционной системы до такой степени, чтобы ошибки в операционной системе не могли привести к случайному изменению прикладных программ или их данных;

г) операционная система должна защищать все прикладные программы и данные от случайного их изменения операторами системы или обслуживающим персоналом;

д) прикладные программы не должны иметь возможности ни остановить систему, ни вынудить ее изменить другую прикладную программу или ее данные;

е) когда прикладная программа обращается к операционной системе, должна проверяться допустимость всех параметров. Прикладная программа не должна иметь возможности изменить эти параметры между моментами проверки и реального их использования операционной системой;

ж) никакие системные данные, непосредственно доступные прикладным программам, не должны влиять на функционирование операционной системы. Ошибка в прикладной программе, вследствие которой содержимое этой памяти может быть случайно изменено, приводит в конце концов к сбою системы;

и) прикладные программы не должны иметь возможности в обход операционной системы прямо использовать управляемые ею аппаратные ресурсы. Прикладные программы не должны прямо вызывать компоненты операционной системы, предназначенные для использования только ее подсистемами;

к) компоненты операционной системы должны быть изолированы друг от друга так, чтобы ошибка в одной из них не привела к изменению других компонентов или их данных;

л) если операционная система обнаруживает ошибку в себе самой, она должна попытаться ограничить влияние этой ошибки одной прикладной программой и, в крайнем случае, прекратить выполнение только этой программы;

м) операционная система должна давать прикладным программам возможность по требованию исправлять обнаруженные в них ошибки, а не безоговорочно прекращать их выполнение.

Реализация многих из этих принципов влияет на архитектуру лежащего в основе системы аппаратного обеспечения. Хотя в формулировке многих из них употребляются слова «операционная система», они применимы к любой программе (будь то операционная система, монитор телеобработки или подсистема управления файлами), которая занята обслуживанием других программ.

4. Четвертая подгруппа – ***введение алгоритмической избыточности*** (см. п. III.1.5).

III.4.4. Проектирование надежного программного обеспечения и его реализация

В проекте функционально надежного программного обеспечения рекомендуется применять ***широкий спектр хорошо апробированных правил и рекомендаций***. К ним относятся следующие (рис. III.4.10):

- *модульный подход*;

- программы создают на основе *существующих стандартов проектирования и кодирования*;

- жестко *типизированные языки программирования*, технология *структурного, а также объектно-ориентированного программирования*;

- при создании программ с очень высокими требованиями функциональной надежности рекомендуется *применять такие языки программирования*, как LadderDiagrams, FunctionalBlocks, StatementList, ADA, MODULA-2, и ограниченно рекомендуются распространенные языки программирования типа C++;

- рекомендуется применять *только аттестованный транслятор или транслятор*, *проверенный в использовании*;

- рекомендуется основываться *на библиотеке проверенных модулей и апробированных компонентов*;

- рекомендуется широко использовать *функциональное тестирование, тестирование методом «черного ящика» и тестирование производительности системы с данным программным обеспечением*;

- необходимо *регистрировать и анализировать данные* и др.

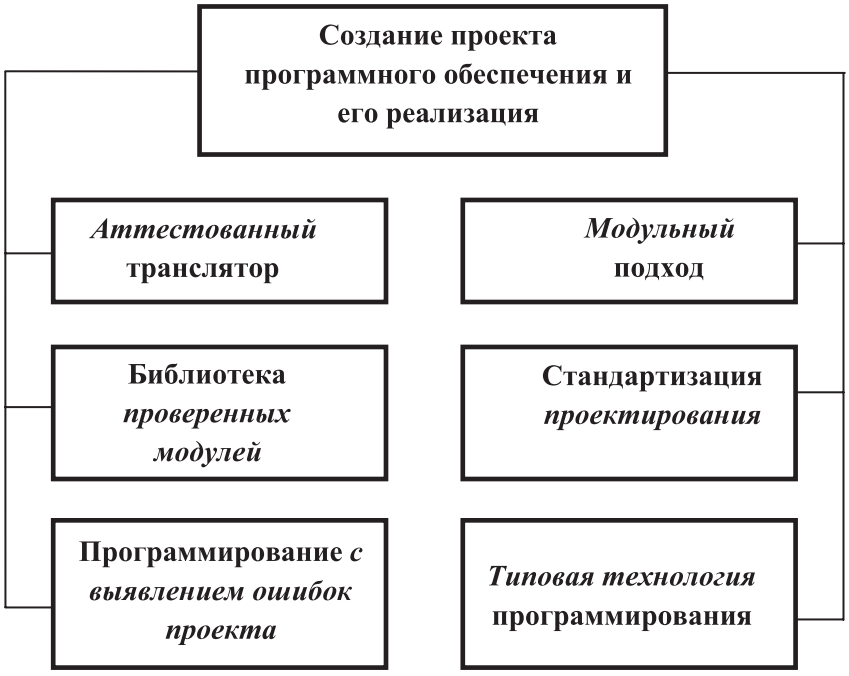


Рис. III.4.10. Методы и способы создания проекта надежного программного обеспечения

Программирование с выявлением ошибок проекта. Цель: обнаружить остаточные ошибки проекта программного обеспечения во время выполнения программы, чтобы предотвратить критически опасные отказы системы и продолжить работу для обеспечения высокой надежности. Суть метода. Производится проверка предварительного условия (прежде, чем последовательность состояний будет выполнена, начальные условия проверяются на предмет их достоверности) и постусловия (результаты проверяются после выполнения последовательности состояний). Если либо предварительное условие, либо постусловие не выполняются, обработка останавливается с ошибкой.

III.4.4.1. Верификация программного обеспечения

Верификация производится последовательно на уровне отдельных программ, программных модулей и программного обеспечения в целом. При этом анализируется программная документация, проверяется корректность исходных и промежуточных данных, корректность алгоритмов, тестируются программы.

Программа может быть тестирована либо полностью, либо выборочно в отдельных точках пространства исходных данных.

При *выборочном тестировании* надежность программы не может быть полностью гарантирована. Если тесты предлагаются программистом, то они могут охватить только те части программы, с которыми программист наиболее знаком. Поэтому многие скрытые ошибки могут оставаться не обнаруженными.

Полное тестирование при всех возможных входных наборах программы или даже тестирование всех путей в структуре программы нереально, так как число тестов будет недопустимо большим. Поэтому часто используется *структурное выборочное тестирование*, основанное на разделении пространства исходных данных на классы, причем каждый класс позволяет

подтвердить определенные свойства или работоспособность определенных элементов структуры программы. *Тестирование – процесс выполнения программы с намерением найти ошибки.*

Тестирование оказывается довольно необычным процессом (вот почему оно и считается трудным), так как это процесс разрушительный. Ведь цель проверяющего – заставить программу сбиться. Он доволен, если это ему удается; если же программа на его тесте не сбивается, он не удовлетворен.

Невозможно гарантировать отсутствие ошибок в программе; в лучшем случае можно попытаться показать наличие ошибок. Если программа правильно ведет себя для значительного набора тестов, нет оснований утверждать, что в ней нет ошибок; со всей определенностью можно лишь утверждать, что неизвестно, когда эта программа не работает. Конечно, если есть причины считать данный набор тестов способным с большой вероятностью обнаружить все возможные ошибки, то можно говорить о некотором уровне уверенности в правильности программы, устанавливаемой этими тестами.

О тестировании говорить довольно трудно, поскольку, хотя оно и способствует повышению надежности ПО, его значение ограничено. Надежность невозможно внести в программу в результате тестирования, она определяется правильностью этапов проектирования. Наилучшее решение проблемы надежности – с самого начала не допускать ошибок в программе. Однако вероятность того, что удастся безупречно спроектировать большую программу, бесконечно мала. Роль тестирования состоит как раз в том, чтобы определить местонахождение немногочисленных ошибок, оставшихся в хорошо спроектированной программе. Попытки с помощью тестирования достичь надежности плохо спроектированной программы совершенно бесплодны.

III.4.2. Интеграция программного обеспечения с аппаратными средствами

При *интеграции* возникает ряд проблем, связанных с обеспечением интероперабельности, правильности и контролируемости программ в аппаратной среде, устойчивостью программ к внешним и внутренним возмущающим воздействиям, в первую очередь, к сбоям аппаратуры (рис. III.4.11), которые являются источниками сбойных ошибок.

Сбойные ошибки – это результаты возмущающих воздействий незащищенных сбоев на процессы выполнения программ, информационных технологий и на информационный процесс в целом. Здесь под незащищенными сбоями подразумеваются сбои функционального характера вследствие однократного искажения перерабатываемой или хранящейся в информационной технике информации. Эти сбои возникают под воздействием внутренних или внешних дестабилизирующих факторов (помех). Сбойные ошибки проявляются в виде искажений, подмены или потерь данных, ошибок в промежуточных и/или в выходных результатах.

Основное деструктивное влияние на результаты информационного процесса оказывают сбои функционального характера. Они приводят к ошибкам в выполнении микроопераций, а те, в свою очередь, к ошибкам в выполнении операций, которые влияют на микропроцессы, затем и на процесс. Сбои функционального характера, в отличие от сбоев типа кратковременных самоустраняющихся отказов, обнаруживаются в редких случаях. Они проявляются в основном через сбойные ошибки. Цена таких ошибок в нарушении надежности выполнения информационной технологии в информационной системе может быть очень высока, поскольку они приводят к серьезным негативным последствиям в результатах управления подчиненными объектами.

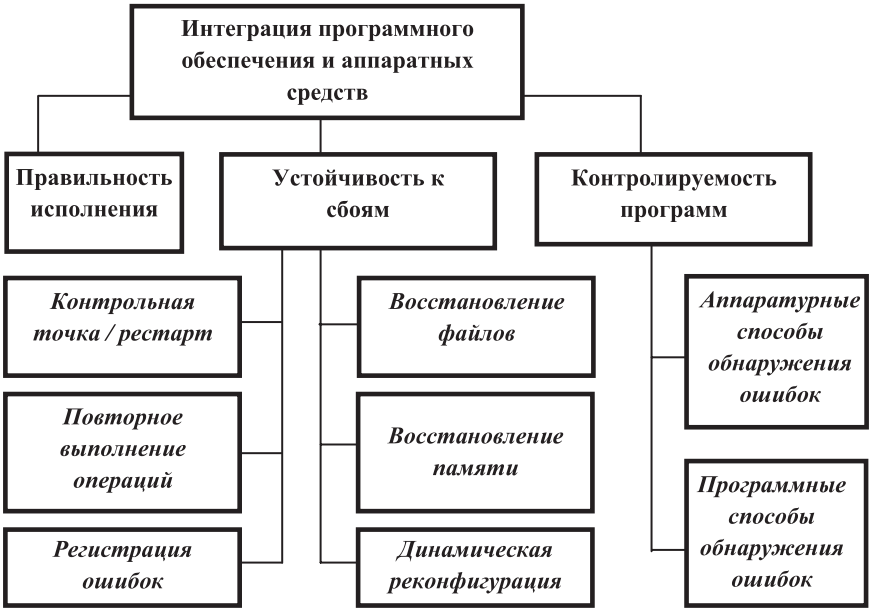


Рис. III.4.11. Способы обеспечения надежности системы при интеграции программных и аппаратных средств

Способы обнаружения сбойных ошибок при исполнении программ разделяются на аппаратные и программные.

Среди аппаратных способов наиболее широкое распространение получили:

• *ошибкообнаруживающие коды* (коды с проверкой на четность в устройствах управления, обработки и передачи информации, коды Хэмминга, циклические и итеративные коды для контроля хранения и передачи информации, равновесные коды для контроля управляющих автоматов);

• *последовательное тестирование* всех функциональных узлов микропроцессоров;

• *самотестирование микропроцессоров* на основе аппаратно-микропрограммных средств;

• *параллельная работа двух идентичных технических средств* с целью сравнения полученных результатов.

Аппаратные способы контроля во многих случаях не обеспечивают обнаружения большого многообразия ошибок, возникающих в ходе работы информационных систем (ИС). Основные из них:

– зацикливания и остановы исполнения программ;

– самоблокировки (клинчи) исполнения программ;

– перегрузки ИС по пропускной способности;

– нарушение последовательностей вызова подпрограмм;

– ошибки взаимного прерывания программ и др.

В целях оперативного обнаружения факта подобных ошибок в рабочем режиме информационной системы применяются программные способы контроля. К ним относятся:

• *проверки на логические несоответствия*. Например, вместо истинного значения некоторой двоичной функции $y = x \wedge \overline{x} = 0$ при исполнении участка программы получено значение $y^* = x \wedge \overline{x} = 1$, где $x = \{1, 0\}$. Поскольку контрольная сумма КС=$y \oplus y^*$=1 отлична от нуля, то принимается решение о наличии ошибки в информационном процессе;

• *проверки по предельным значениям вычисляемых параметров*;

• *статический контроль стробированием* (контролируемые параметры сравниваются с допустимыми значениями);

• *динамический контроль стробированием* (контроль гладкости изменения переменных с течением времени);

• *проверки контрольных соотношений* (например, сравнение с эталоном, вычисление контрольных функций);

• *контроль с использованием избыточных переменных*;

• *способ охранных таймеров*, который основывается на программной реализации таймеров совместно со *счетчиками относительного или текущего времени*;

• *способ контроля ключевых кодов*, при котором формируется таблица ключевых кодов и количества включений каждой подпрограммы.

Рассмотрим в постановочном плане некоторые способы обеспечения устойчивости к сбойным ошибкам:

• ***построение алгоритмов толерантных к сбойным ошибкам.*** Отсутствие чувствительности к проявлениям сбоев аппаратуры достигается, в первую очередь, применением избыточных алгоритмов. Таких алгоритмов существует множество, например, алгоритмы сглаживания временных рядов. Для ослабления влияния случайных выбросов результатов вследствие сбоев аппаратуры данные временного ряда подвергаются фильтрации методом скользящего среднего. Метод реализуется следующим образом. Пусть задана исходная последовательность точек временного ряда $S(1)$, $S(2)$, …, $S(K)$. Для данного временного ряда, содержащего K точек, задается ширина W области усреднения – количество точек ряда, участвующих в формировании сглаженного значения рассматриваемой точки. Как правило, область усреднения симметрична относительно рассматриваемой точки и поэтому W – нечетное. Для любого нечетного значения $W > 1$ можно задать целочисленные симметричные смещения границ области усреднения $i_{\min} = (1 - W)/2$; $i_{\max} = (W - 1)/2$, где центральной точке будет соответствовать $i = 0$. Для каждой i-й точки ($i = i_{\min}$, $i_{\min}+1$, …-1, 0, 1, … $i_{\max}$-1, $i_{\max}$) в области усреднения задается соответствующий весовой коэффициент p_i, причем $\sum_{i=i_{\min}}^{i_{\max}} p_i = 1$. Формула для вычисления скользящего среднего имеет вид

$$F(k) = \sum_{i=i_{\min}}^{i_{\max}} p_i \cdot S(k+i).$$

Таким образом, каждая точка $S(k)$ исходного ряда заменяется на «усредненную» точку $F(k)$, где $k = 1 \ldots K$;

• ***динамическое управление режимами повышения производительности и ошибкоустойчивости информаци-***

онных систем. Суть его в следующем. Из анализа причин сбойных ошибок [24, 29, 38] следует, что существует прямая зависимость между частотой переключения триггеров в цифровой аппаратуре ИС и частотой сбойных ошибок на выходе этих устройств ИС, поскольку триггеры были и остаются их базовыми элементами. Для повышения помехоустойчивости применяется стробирование триггеров [29]. В результате сбои триггеров могут произойти только под воздействием помех во время их переключения. Однако чем чаще поступают входные и стробирующие сигналы, тем выше частота переключения триггеров и тем выше вероятность сбойных ошибок, которые вызваны сбоями как сбоями логической части устройства, так и самих триггеров, и зафиксированы ошибочными состояниями триггеров. В свою очередь, частота поступления входных сигналов находится в прямой зависимости от пропускной способности ИС.

Таким образом, чем выше пропускная способность ИС, тем выше вероятность возникновения сбойных ошибок. Естественен вопрос: нужно ли стремиться к повышению пропускной способности ИС, если при этом увеличится количество ошибок в выходных результатах? Конечно, не нужно добиваться усиления одной характеристики технической эффективности за счет другой. Возможен разумный компромисс между уровнями значений конфликтующих характеристик ИС, при котором достигается наилучший уровень технической эффективности системы. Технически это осуществимо путем регулирования пропускной способности ИС по установленному в работе [24] решающему правилу. В случае снижения пропускной способности системы снижается вероятность сбойных ошибок. При этом также появляется некоторое свободное время в выполнении информационного процесса (ИП), что обусловлено сохранением прежнего высокого быстродействия элементной базы при снижении за-

грузки системы. Это свободное время можно использовать также и для защиты от сбойных ошибок путем возврата ИП на глубину, определяемую рамками этого свободного времени. Через некоторое время может возникнуть потребность в повышении пропускной способности относительно предыдущего уровня. В этом случае имевшееся ранее некоторое свободное время в выполнении ИП либо сокращается, либо полностью исключается. В дальнейшем также возможны управления как в сторону снижения, так и в сторону повышения пропускной способности системы. При отсутствии свободного времени в выполнении ИП управление на повышение пропускной способности не формируется. Это и является динамической стабилизацией устойчивости к сбойным ошибкам относительно информационной нагрузки ИС;

• ***программные методы логического контроля***. Логический контроль основывается на избыточности исходной, промежуточной и результирующей информации. Он включает в себя ряд способов:

– *проверки на логические несоответствия*;

– *статический контроль стробированием* (контролируемые параметры сравниваются с допустимыми значениями);

– *динамический контроль стробированием* (контроль гладкости изменения переменных с течением времени);

– *проверки контрольных соотношений* (например, сравнение с эталоном, вычисление контрольных функций);

– *контроль с использованием избыточных переменных*;

– *способ охранных таймеров*, который основывается на программной реализации таймеров совместно со *счетчиками относительного или текущего времени*;

– *способ контроля ключевых кодов*, при котором формируется таблица ключевых кодов и количества включений каждой подпрограммы.

III.4.5. Обеспечение надежности программных средств в процессе подтверждения соответствия, эксплуатации и сопровождения

III.4.5.1. Подтверждение соответствия программного обеспечения

Основными инструментами подтверждения соответствия программ являются их анализ и испытания. В качестве вспомогательного средства процесса подтверждения соответствия могут использоваться имитация и моделирование программ. Предварительно должна быть разработана и согласована с соответствующими органами программа подтверждения соответствия. Программа должна идентифицировать шаги, необходимые для наглядного подтверждения адекватности в выполнении требований, изложенных в спецификации на программное обеспечение [59]:

- спецификации требований программного обеспечения;
- спецификации архитектуры программного обеспечения;
- спецификации проекта программного обеспечения;
- спецификация проекта каждого программного модуля.

Прогрессивные методы верификации, интеграции с аппаратными средствами, подтверждения соответствия ПС заданным требованиям тесно переплетаются с современными технологиями сертификационных испытаний по требованиям качества (в том числе надежности и функциональной безопасности) и по требованиям безопасности информации.

В настоящее время сформированы два направления сертификационных испытаний программного обеспечения: ***по требованиям к качеству, функциональной надежности и функциональной безопасности*** в соответствии с регламентами и стандартами. Так, на железнодорожном транспорте Российской Федерации вступили в силу регламенты Таможенного Союза [63 – 65] и

поддерживающие их межгосударственные и национальные стандарты. Это направление осуществляется путем декларирования разработчиком (заявителем) соответствия заданным требованиям ПС с привлечением третьей независимой компетентной стороны в соответствии с Федеральным Законом «О техническом регулировании». Независимая и аккредитованная должным образом третья сторона – это испытательный центр (лаборатория), который после аккредитации включен в Реестр Таможенного Союза и имеет право по соответствующим пунктам регламентов давать заключение, которое подтверждает (или не подтверждает) представленную разработчиком декларацию соответствия. Специфика ПО состоит в том, что оно функционирует в составе ИС и к нему отдельно не предъявляются нормы безопасности. С другой стороны, процедуры оценивания качества (функциональной надежности и безопасности) ПО в соответствии с принятыми показателями декларирования соответствия весьма специфичны и основываются, главным образом, на автономных испытаниях на специально разработанных стендах или стендах разработчика программного средства. После чего проводятся интеграционные испытания в составе системы.

В результате сертификационных испытаний ПС выполняется следующее:

- повышается функциональная надежность и качество ПО, поскольку выявляются ошибки в программах и их документальном обеспечении;
- приводится в соответствие со стандартами [58-61] программная документация;
- на хранение передаются такие программы, которые становятся качественными продуктами интеллектуальной собственности организации;
- обеспечивается независимость пользователей от разработчиков ПС;

• в распоряжение заказчика поступает программный продукт, который соответствует его потребностям, сопровождается и может в случае форс-мажорных обстоятельств легко быть восстановлен без дополнительных материальных затрат.

Требования к функциональной надежности программы базируются на условии обеспечения способности программы выполнять предусмотренные функции в реальных условиях эксплуатации. Программные средства, составляющие ИС, широко применяются для управления ответственными объектами (объектами автоматики и телемеханики, оперативно-технологической связи, управления транспортными средствами, управления топливно-энергетическими ресурсами и др.). Ошибки в функционировании этих ПС, а также наличие в них недекларированных возможностей могут привести к опасным ошибкам в управлении и в результате этого к человеческим жертвам, материальному и моральному ущербу, а также к нарушению экологии и т. д. Здесь под недекларируемыми возможностями подразумеваются функциональные возможности ПО, не описанные или не соответствующие описанным в документации, при использовании которых возможно нарушение конфиденциальности, доступности или целостности обрабатываемой информации.

Сертификационные испытания ***по требованиям безопасности информации и отсутствия недекларированных возможностей*** проводятся в соответствии с Руководящими документами Федеральной службы по техническому и экспортному контролю (ФСТЭК России). Контроль отсутствия недекларированных возможностей программного обеспечения предполагает глубокое его исследование и связан с анализом, как исполняемого кода, так и исходных текстов программ. Методологической основой таких исследований являются общие принципы анализа программ с учетом аспектов, связанных с информационной безопасностью.

III.4.5.2. Эксплуатация, сопровождение и конфигурация функционально надежных программных средств

Процесс эксплуатации (operation process) охватывает действия и задачи оператора – организации, эксплуатирующей систему (рис. III.4.12).

Он включает в себя следующие этапы:

1. *Подготовительная работа.* Этот этап эксплуатации предусматривает выполнение оператором следующих задач:

• планирование действий и работ, выполняемых в процессе эксплуатации, и установку эксплуатационных стандартов;

• определение процедур локализации и разрешения проблем, возникающих в процессе эксплуатации.

2. *Эксплуатационное тестирование*. Данный этап осуществляется для каждой очередной редакции программного продукта, после чего она передается в эксплуатацию.

Эксплуатация системы выполняется в предназначенной для этого среде в соответствии с пользовательской документацией.

3. *Поддержка пользователей* заключается в оказании помощи и консультации при обнаружении ошибок в процессе эксплуатации ПС.

Процесс сопровождения (maintenance process) предусматривает действия и задачи, выполняемые сопровождающей организацией (службой сопровождения). Данный процесс активизируется при изменениях (модификациях) программного продукта и соответствующей документации, вызванных возникшими проблемами или потребностями в модернизации либо адаптации ПС. Под сопровождением понимается внесение изменений в ПС в целях исправления ошибок повышения производительности или адаптации к изменившимся условиям работы или требованиям.

Изменения, вносимые в существующее ПС, не должны нарушать его целостность. Процесс сопровождения включает следующие этапы:

Подготовительная работа службы сопровождения включает следующие задачи:

• планирование действий и работ, выполняемых в процессе сопровождения;

• определение процедур локализации и разрешения проблем, возникающих в процессе сопровождения.

Рис. III.4.12. Порядок ввода в эксплуатацию надежного программного обеспечения

Анализ проблем и запросов на модификацию ПС, выполняемый службой сопровождения, включает следующие задачи:

• анализ сообщения о возникшей проблеме или запроса на модификацию ПС относительно его влияния на организацию, существующую систему и интерфейсы с другими системами. При

этом определяются следующие характеристики возможной модификации: тип (корректирующая, улучшающая, профилактическая или адаптирующая к новой среде); масштаб (размеры модификации, стоимость и время ее реализации); критичность (воздействие на производительность, надежность или безопасность);

• оценку целесообразности проведения модификации и возможных вариантов ее проведения;

• утверждение выбранного варианта модификации.

Модификация ПС предусматривает определение компонентов ПС их версий и документации, подлежащих модификации, и внесение необходимых изменений в соответствии с правилами *процесса наработки.* Подготовленные изменения тестируются и проверяются по критериям, определенным в документации. При подтверждении корректности изменений в программах проводится корректировка документации.

Проверка и приемка заключаются в проверке целостности модифицированной системы и утверждении внесенных изменений. При *переносе ПС в другую среду* используются имеющиеся или разрабатываются новые средства переноса, затем выполняется конвертирование программ и данных в новую среду. С целью облегчить переход предусматривается параллельная эксплуатация ПС в старой и новой среде в течение некоторого периода, когда проводится необходимое обучение пользователей работе в новой среде.

Снятие ПС с эксплуатации осуществляется по решению заказчика при участии эксплуатирующей организации, службы сопровождения и пользователей в соответствии с договором. Аналогично переносу ПС в другую среду с целью облегчить переход к новой системе, предусматривается параллельная эксплуатация старого и нового ПС в течение некоторого периода, когда выполняется необходимое обучение пользователей работе с новой системой.

Процесс управления конфигурацией (configuration management process) предполагает применение административных и технических процедур на всем протяжении жизненного цикла ПС для определения состояния компонентов ПС в системе, управления модификациями ПС, описания и подготовки отчетов о состоянии компонентов ПС и запросов на модификацию, обеспечения полноты, совместимости и корректности компонентов ПС, управления хранением и поставкой ПС. Под *конфигурацией ПС* понимается совокупность его функциональных и физических характеристик, установленных в технической документации и реализованных в ПС.

Управление конфигурацией позволяет организовать, систематически учитывать и контролировать внесение изменений в ПС на всех стадиях жизненного цикла. Процесс управления конфигурацией ПО включает в себя следующие этапы:

- *подготовительная работа* заключается в планировании управления конфигурацией;

- *идентификация конфигурации* устанавливает правила, с помощью которых можно однозначно идентифицировать и различать компоненты ПС и их версии. Кроме того, каждому компоненту и его версиям соответствует однозначно обозначаемый комплект документации. В результате создается база для однозначного выбора и манипулирования версиями компонентов ПС, использующая ограниченную и упорядоченную систему символов, идентифицирующих различные версии ПС;

- *контроль конфигурации* предназначен для систематической оценки предполагаемых модификаций ПС и координированной их реализации с учетом эффективности каждой модификации и затрат на выполнение. Он обеспечивает контроль состояния и развития компонентов ПС и их версий, а также адекватность реально изменяющихся компонентов их комплектной документации;

- *учет состояния конфигурации* представляет собой регистрацию состояния компонентов ПС, подготовку отчетов обо всех реализованных и отвергнутых модификациях версий компонентов ПС. Совокупность отчетов обеспечивает однозначное отражение текущего состояния системы и ее компонентов, а также ведение истории модификаций;

- *оценка конфигурации* заключается в оценке функциональной полноты компонентов ПС, а также соответствия их физического состояния текущему техническому описанию.

III.4.6. Вопросы для самоконтроля

1. Опишите стадии жизненного цикла разработки надежного программного обеспечения.
2. Опишите характерные недостатки разрабатываемых программных средств.
3. Что представляет собой маршрутная карта функциональной надежности программного обеспечения?
4. Какими рекомендациями следует руководствоваться при разработке спецификации требований к программам?
5. В чем суть защитного программирования?
6. Опишите способы многоверсионного программирования.
7. Раскройте методы и способы создания проекта надежного программного обеспечения.
8. Изложите способы обеспечения надежности системы при интеграции программных и аппаратных средств.
9. В чем состоит подтверждение соответствия программных средств?
10. Опишите процессы эксплуатации, сопровождения и конфигурации программных средств.

ГЛАВА III.5. ФУНКЦИОНАЛЬНАЯ БЕЗОПАСНОСТЬ ИНФОРМАЦИОННЫХ СИСТЕМ

III.5.1. Введение

Пристальное внимание к функциональной безопасности ответственных (критичных по безопасности) технических объектов проявлялось специалистами еще с середины 60-х годов прошлого столетия. Это относилось, в первую очередь, к связанным с безопасностью управляющим электротехническим устройствам и системам в атомной промышленности, на железнодорожном транспорте, в некоторых системах вооружения и военной техники. Критичные (опасные) отказы вызывали ошибки в управлении, которые приводили или могли привести к человеческим жертвам, недопустимому ущербу внешней среде, экономике и имиджу отрасли. Разрабатывались приемы анализа причин возникновения опасных отказов, технические решения по гарантированному исключению таких отказов. В проектируемых электротехнических устройствах и даже в ряде систем удавалось обнаруживать практически все возможные опасные отказы. Это объясняется простотой релейных схем и возможностью перечислить все ситуации возникновения опасных отказов.

Широкое внедрение информационных технологий, создание связанных с безопасностью многофункциональных программно-аппаратных систем управления в атомной промышленности, на транспорте, в энергетике, нефтегазовом комплексе исключило

возможность ручного и даже автоматизированного перебора всех возможных ситуаций, приводящих к опасным отказам. Поэтому ранее существовавшие практические приемы построения безопасных управляющих устройств оказались малоэффективными. Потребовалось создание научной методологии, теории и практики анализа и синтеза функциональной безопасности критичных по безопасности электронных программируемых устройств и информационных систем на всех этапах их жизненного цикла. Эта проблема начала решаться с середины 80-х годов XX века. Она продолжает активно развиваться в настоящее время. Уже очевидно, что в современных связанных с безопасностью устройствах и системах не существует абсолютной безопасности. Всегда есть некоторый риск, связанный с возникновением опасных отказов, который должен быть сведен с помощью методов и способов теории и практики функциональной безопасности к допустимому остаточному уровню риска. Философия безопасности информационно-управляющих систем состоит в выборе принципа приемлемости допустимого риска, т.е. в определении главной цели достижения безопасности системы.

Приемлемость риска должна определяться на основе признанных принципов. Наряду с принципом ALARP/ (As Low, As Reasonably Possible – риск настолько низкий, насколько это возможно в разумных пределах), наиболее известны Minimum Endogenous Mortality (принцип MEM), который формулируется следующим образом: «Угроза, связанная с новой системой не должна повышать цифру минимальной эндогенной смертности для индивидуума». Globalement Au Moins Aussi Bon (принцип GAMAB), который формулируется следующим образом: «Все новые системы должны в целом представлять глобальный уровень безопасности, по меньшей мере такой же высокий, как в какой-нибудь сравниваемой существующей системе». Формулировка учитывает все, что уже было предпринято и косвенно

требует прогресса, который должен быть достигнут в проектируемой системе, исходя из требования «хотя бы также высок».

Теория функциональной безопасности в настоящее время находится в стадии развития. Практика обеспечения функциональной безопасности опережает теорию. Мировым сообществом разработаны многие методологические положения функциональной безопасности, которые нашли отражения в международных и национальных стандартах [57, 63 и др.], в европейских и международных стандартах и других нормативных документах железнодорожного применения [6 – 11, 55, 56, 70, 72 и др.].

Эта научная дисциплина имеет свой объект и предмет исследования. Объект исследования – электрические/электронные программируемые устройства и системы, а также информационно-управляющие системы, связанные с безопасностью; предмет исследования – опасные функциональные отказы управляющего устройства (системы), под которыми понимаются только такие сбойные, программные ошибки, ошибки операторов, нарушения целостности данных и программ вследствие информационных атак, а также отказы техники, которые приводят к нарушению ответственных функций управления или к критическим ошибкам в выполнении этих функций. Основная цель теории и практики функциональной безопасности состоит в разработке методов и технических решений, обеспечивающих гарантированное обнаружение, устранение опасных функциональных отказов или блокирование управляющих воздействий в случаях невозможности устранения опасных функциональных отказов. Уровень гарантии напрямую зависит от результатов оценивания остаточного допустимого риска.

К основным направлениям развития теории функциональной безопасности относятся:

– разработка теоретических основ расчета вероятности или возможности возникновения опасных функциональных отка-

зов, а также показателей функциональной безопасности сложных восстанавливаемых систем;

– разработка и исследование комплекса моделей функциональной безопасности восстанавливаемых двух, трех и более канальных систем с перезапусками каналов, а также моделей многоуровневых систем обеспечения функциональной безопасности;

– разработка методов и способов построения многоканальных систем с многоверсионным программным обеспечением и перекрестными обратными связями от выходов ко входам аппаратно-программных каналов управления;

– разработка методологии и инструментальных средств для проведения в реальном масштабе времени сертификационных испытаний программного обеспечения и связанных с безопасностью управляющих информационных систем;

– разработка технологии доказательства функциональной безопасности системы на основе комплексного взвешенного применения аналитических методов, экспертных оценок, результатов стендовых и вспомогательных испытаний и, особенно, ускоренных натурных испытаний, которые позволяют (при условии доказательства адекватности) в приемлемые сроки дать корректную имитацию возникновения опасных функциональных отказов системы;

– разработка нормативных документов и методических материалов по оценкам рисков нарушения функциональной безопасности для задания требований к системам и выполнения оценок остаточных рисков и др.

Специалисты вольно или невольно задаются вопросом: в чем общность и различие теории функциональной безопасности и теории надежности. Ответ на этот вопрос частично приведен ранее при определении объекта, предмета и целей теории функциональной безопасности. Объектами исследования теории надежности

являются любые технические и/или эргатические устройства и системы как связанные, так и не связанные с безопасностью, тогда как объектами исследования теории функциональной безопасности являются только связанные с безопасностью устройства и системы, причем, они не могут быть любыми, а обязательно управляют какими-либо внешними объектами. Предметом теории надежности, так же, как и теории функциональной безопасности, являются случайные процессы отказов и восстановлений, сбойные, программные ошибки, ошибки операторов. Однако при этом имеются следующие основные принципиальные различия:

– теория функциональной безопасности оперирует с очень редкими событиями (опасными функциональными отказами), частота возникновения которых в соответствии с оценками международных стандартов на 2 – 4 порядка меньше частоты отказов и ошибок, исследуемых теорией надежности. Эти редкие события остаются за рамками оценок теории надежности.

– теория надежности исследует вероятность и возможность выполнять предусмотренные задачи в реальных условиях эксплуатации, сохраняя параметры в пределах заданных допусков. Это означает, что теория надежности изучает влияние внутренних возмущающих факторов (отказов и ошибок) только на работу системы, без оценки их воздействия на внешнюю среду. Теория функциональной безопасности изучает воздействие отказов и ошибок в работе системы на объекты управления и в целом на внешнюю среду. При этом центральная задача состоит в исследовании вероятности и возможности парирования предусмотренными функциями безопасности опасных функциональных отказов;

– функции безопасности предназначены для оперативного обнаружения и устранения или блокирования отказов. Решение задач устранения или блокирования обнаруженных отказов не представляет особых трудностей. Проблема состоит в том, что-

бы любым возможным способом своевременно и достоверно обнаружить очень редкие события опасных отказов (п. III.1.7). Философия теории надежности имеет принципиальное отличие – возможности средств обнаружения отказов объекта должны быть достаточными для сохранения приемлемого уровня готовности (технического использования) объекта. Эта позиция объясняется тем, что при введении средств контроля в объект без избыточности безотказность этого объекта снижается за счет отказов средств контроля. При этом чем меньше объем средств контроля, тем ниже его эффективность. Поэтому стремятся оптимизировать объем (или время контроля) таким образом, чтобы обеспечить приемлемый уровень готовности объекта при допустимых потерях его безотказности.

– при наличии избыточных ресурсов (структурных, временных, информационных, функциональных) налицо два диаметрально противоположных подхода к архитектуре объекта: с позиций теории надежности эти ресурсы используются в качества резерва для повышения безотказности готовности объекта; с позиций теории функциональной безопасности эти избыточные ресурсы используются для обеспечения обнаружения опасных функциональных отказов путем построения многоканальных объектов с параллельной обработкой информации и аппаратными или программными средствами анализа или/и сравнения результатов. При этом допускается снижение уровня надежности объекта за счет отъема в интересах функциональной безопасности избыточных ресурсов.

– все работы по функциональной безопасности основываются на оценках рисков возможных ущербов внешней среде. На этой основе определяют требования к уровню гарантии обнаружения опасных функциональных отказов, требования к архитектуре объекта, требования к надежности аппаратных и программных средств. Оцениваются остаточные риски на всех этапах жизнен-

ного цикла объекта. Согласовываются с заказчиком допустимые остаточные риски и уточняются по мере эксплуатации системы. Эти работы слабо коррелированы с работами по обеспечению надежности и имеют важное самостоятельное значение.

Приведенные положения означают, что вопрос о превалировании теории функциональной безопасности над теорией надежности не должен иметь прямой постановки – каждое из этих научных направлений решает свои блоки задач, хотя и базируется на общем исходном материале – отказах и ошибках составных средств. Естественно, что уменьшение интенсивности отказов и ошибок объекта влечет за собой повышение надежности и в некоторой мере функциональной безопасности. Однако учитывая, что кардинальных достижений в этом вопросе не следует ожидать, главным и определяющим инструментом обеспечения функциональной безопасности аппаратно-программного объекта должны быть функции безопасности, сформированные по результатам оценки рисков.

Стандарты по функциональной безопасности направлены на обеспечение безопасной работы оборудования и программного обеспечения, хотя содержащиеся в них рекомендации способствуют повышению качества и бесперебойной работы аппаратных и программных средств и систем. Гарантией более высокого качества и бесперебойной работы является строгое выполнение требований системы стандартов по качеству и надежности систем. Применение стандартов по функциональной безопасности на всех этапах жизненного цикла устройств и систем, связанных с безопасностью, дает основание утверждать, что: 1) будут корректно заданы требования, 2) в соответствии с этими требованиями будут созданы безопасные для внешней среды изделия и 3) результаты их эксплуатации с позиций функциональной безопасности будут соответствовать тем прогнозам, которые имели место при разработке и производстве данных изделий.

В соответствии с национальными и международными стандартами подтверждение функциональной безопасности обязательно. Более того, для каждого объекта, связанного с безопасностью, в соответствии с требованиями международных стандартов должен быть создан документ «Доказательство безопасности» (см. п. III.1.4). Он включает в себя описание объекта, концепцию и программу обеспечения функциональной безопасности, все виды выполненных доказательств безопасности (аналитические, экспериментальные, моделирование, экспертные оценки и др.), имеющиеся сертификаты соответствия на составные разработанные или покупные средства, отзывы специалистов. Этот документ представляет собой паспорт объекта, подтверждающий его функциональную безопасность на всех этапах жизненного цикла, начиная с этапа задания требований и заканчивая его утилизацией.

Оценку соответствия функциональной безопасности объекта на основании результатов сертификационных испытаний осуществляют негосударственные органы, которые называются Органами по сертификации. Эти органы должны быть обязательно аккредитованы в системе сертификации государственных органов. В Германии, например, Органом по сертификации является всемирно известный институт TUV, который аккредитован при правительстве страны. Для оценивания соответствия любой Орган по сертификации должен назначить испытательную лабораторию (испытательный центр), аккредитованную при данном органе. Это испытательное подразделение должно быть независимо от заявителя объекта на сертификационные испытания.

Механизмы и процедуры оценки соответствия изложены в «Законе о техническом регулировании» Российской Федерации 2002 г. [62] и гармонизированы с международными нормами. Оценка соответствия по требованиям функциональной безопасности востребована в любых отраслях, где имеют место связан-

ные с безопасностью объекты. В большей степени это относится к тем отраслям, где имеют место критически важные объекты: системы вооружения и военной техники, системы транспорта, системы атомной, нефтегазовой промышленности, системы энергетики и др.

Прямое копирование подходов развитых зарубежных стран к оцениванию соответствия функциональной безопасности устройств и систем, связанных с безопасностью, на российской почве нецелесообразно и практически невозможно из-за различий в инфраструктуре, условиях эксплуатации этих изделий, их производстве, обслуживании и т. д. Для подтверждения соответствия показателей функциональной безопасности информационных систем управления заданным требованиям необходима гармонизация национальных стандартов с международными, развитие и внедрение методов ускоренных натурных испытаний, автоматизация процессов испытаний, создание жесткого контроля со стороны государственных органов за выполнением требований стандартов и процедур испытаний и оценки соответствия по их результатам.

III.5.2. Состояния безопасности. Функция безопасности и полнота безопасности.

III.5.2.1. Состояния безопасности

К основным понятиям функциональной безопасности относятся:

– *система, связанная с безопасностью*;

– *состояния безопасности;*

– *функция безопасности и полнота безопасности;*

– *уровень полноты безопасности*.

Система, связанная с безопасностью, – специальная система, которая:

– обеспечивает выполнение функций безопасности, необходимых для достижения или поддержания безопасного состояния объекта управления;

– предназначена для достижения необходимой полноты безопасности – самостоятельно или совместно с другими связанными с безопасностью встроенными системами, а также внешними средствами снижения риска.

Система, связанная с безопасностью, может быть предназначена:

а) для предотвращения опасного события (т. е. при выполнении такой системой своих функций, опасные события не возникают);

б) для смягчения последствий опасного события путем снижения риска за счет уменьшения последствий;

в) для обеспечения сочетания функций, указанных в пп. а) и б).

Программные средства этих систем также связаны с безопасностью. Примерами могут служить программные средства систем управления движением самолета, автомобиля, систем электрической и диспетчерской централизации на железной дороге, средств управления атомными реакторами и т.д. Иными словами, любое программное средство, отказ которого может повлиять на возникновение аварийной ситуации, следует считать связанным с безопасностью.

Человек может являться частью системы, связанной с безопасностью. Например, человек может получать информацию от программируемого электронного устройства и выполнять действия по обеспечению безопасности на основе этой информации, или же он может выполнять действия по обеспечению безопасности, используя такое устройство.

Состояния безопасности:

1. Исправное или неисправное состояния – состояния системы, при котором обеспечиваются все требования технической документации или не обеспечивается хотя бы одно из этих тре-

бований соответственно. Неисправное состояние программного средства – состояние с необнаруженными ошибками в программе или конструкторской (проектной) документации на программу, могущее вызвать снижение или утрату программным средством способности выполнять требуемую функцию. Например, в результате аппаратного сбоя возникла сбойная ошибка, которая выразилась в искажении заданного количества циклов в модуле программы. При определенном наборе исходных данных такое состояние может привести к «зацикливанию» или останову программы. Другой пример. Программа плохо документирована, что затрудняет ее эксплуатацию и при определенных обстоятельствах может возникнуть ошибка в результате выполнения программы.

2. Работоспособное или неработоспособное состояния – состояния системы, при котором значения всех параметров, характеризующих способность выполнять заданные функции, соответствуют требованиям технической документации или не обеспечивается значение хотя – бы одного параметра соответственно.

3. Защитное состояние – состояние системы, при котором отключено выполнение всех предусмотренных системных функций в случае своевременного обнаружения отказа любого элемента управления или обеспечения безопасности управления. Защитное состояние имеет место только в системах, связанных с безопасностью. В качестве примера в двухканальной системе, связанной с безопасностью, защитное состояние формируется в результате обнаружения неработоспособного состояния хотя бы одного канала путем отключения системы от управляемого оборудования и восстановления работоспособности отказавшего канала. В защитном состоянии не выполняются системные функции, но выполняются все предусмотренные функции безопасности.

4. Опасное состояние – неработоспособное состояние системы, при котором не выполняется хотя бы одна функция безопасности.

5. Неопасное состояние – работоспособное или защитное состояние системы.

Перечисленные термины, определения и их взаимосвязь проиллюстрируем с помощью образов теории множеств. Пусть полное множество состояний безопасности системы (или программного средства, в случае его автономного рассмотрения) обозначается символом S (рис. III.5.1). Подмножество работоспособных состояний – $S_Р$. Подмножество неработоспособных состояний – $\overline{S}_Р$. Очевидно, что оба эти подмножества образуют полное множество состояний безопасности (рис. III.5.1 а) Аналогичное замечание можно сделать относительно подмножества исправных состояний $S_И$ и неисправных состояний $\overline{S}_И$ ($S = S_И \bigcup \overline{S}_И$).

Важно подчеркнуть, что подмножества исправных и работоспособных состояний – это не одно и тоже. Исправные состояния обязательно работоспособны, однако работоспособные состояния не обязательно исправны. Например, в программном средстве содержатся не выявленные ошибки, – это средство по существу неисправно. Однако при многих наборах входных данных эти ошибки не влияют на правильность выходных результатов, т. е. на работоспособность программного средства. Таким образом, подмножество исправных состояний включается в подмножество работоспособных состояний $S_И \subset S_Р$. В подмножество работоспособных состояний входит также часть неисправных состояний.

Само подмножество неработоспособных состояний включается в подмножество неисправных $\overline{S}_Р \subset \overline{S}_И$ (рис. III.5.1 б). При этом подмножество неисправных состояний за вычетом подмножества неработоспособных состояний $\left(\overline{S}_И \setminus \overline{S}_Р\right)$ является

составной частью подмножества работоспособных состояний. Подмножество неработоспособных состояний $\overline{S}_{\text{Р}}$ в свою очередь делится на два подмножества – защитных состояний ($S_{\text{З}}$) и опасных состояний ($\overline{S}_{\text{Н}}$) (см. рис. III.5.1 в) Подмножества неопасных ($S_{\text{Н}}$) и опасных ($\overline{S}_{\text{Н}}$) состояний системы образуют полное множество состояний безопасности (см. рис. III.5.1 г).

Приведенные определения в полной мере распространяются и на программные средства систем, связанных с безопасностью.

Любая связанная с безопасностью информационная система управления должна иметь, по крайней мере, одно безопасное состояние. Состояние называют безопасным, если риск системы, находящейся в этом определенном состоянии, является меньшим, чем приемлемый риск. Под риском подразумевают сочетание частоты (вероятности) события причинения вреда и тяжести этого вреда (ущерба). В общем случае риск – это сочетание вектора частот событий и вектора ущербов от них. В простейшем случае риск оценивают произведением вероятности события на величину ущерба от него. Уровень приемлемого риска устанавливают специалисты, эксплуатирующие информационную систему управления и обеспечивающие ее безопасное функционирование совместно с объектом управления.

Каждая информационная система управления имеет, по крайней мере, два безопасных состояния:

– нормальное эксплуатационное состояние;

– состояние останова (система выключена).

Предполагается, что система свободна от отказов в нормальном эксплуатационном состоянии. Состояние останова обычно представляет собой состояние, в котором система не выполняет системные функции. Безопасное состояние останова должно быть достигнуто в достаточно короткое время через прекращение работы системных функций. Прекращение действия сис-

темных функций может быть активным процессом (дополнительная функция системы).

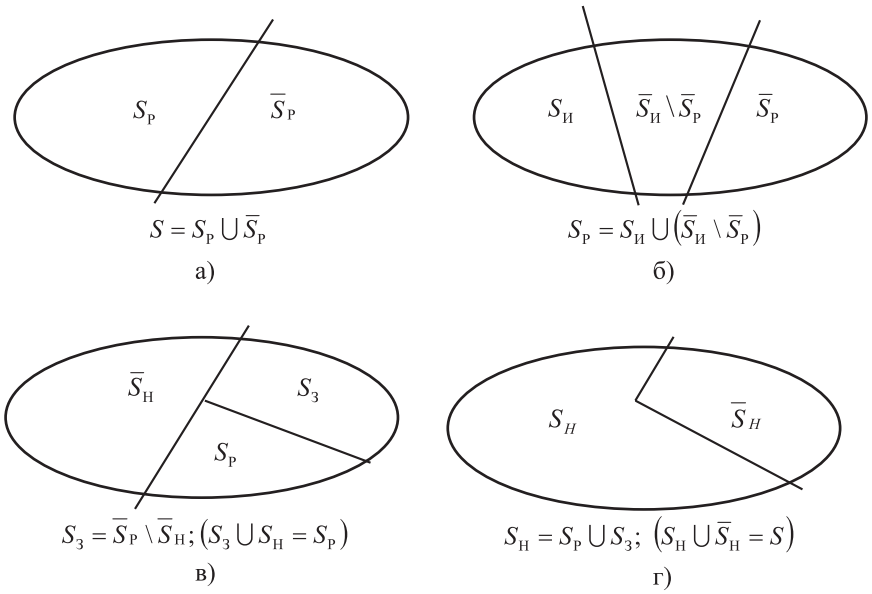


Рис. III.5.1. Состояния безопасности

Могут быть ситуации, в которых ранее существовавшее (безопасное) состояние информационной системы управления не может быть достигнуто в заданное время, например, когда объект управления – самолет – находится в полете. В этом случае прекращение функционирования системы может быть выполнено после посадки самолета на землю.

Таким образом, любая информационная система управления или ее части должны реализовывать функции безопасности.

III.5.2.2. Функция безопасности и полнота безопасности

Функция безопасности – функция, реализуемая связанной с безопасностью системой или внешними средствами снижения

риска, предназначенная для обеспечения или поддержания безопасного состояния применительно к конкретному опасному событию. Внешние средства снижения риска – меры по снижению или смягчению рисков, предпринимаемые отдельно, без использования связанных с безопасностью систем. Внешним средством снижения риска является, например, в системе передачи информации, брандмауэр.

В целом под термином ***«функциональная безопасность» понимается способность системы, связанной с безопасностью, выполнять все предусмотренные в системе функции безопасности при всех предусмотренных условиях в течение заданного периода времени с сохранением остаточного риска возникновения опасных событий на допустимом уровне***.

Эффективность обеспечения безопасности информационных систем управления оценивается с помощью показателей полноты безопасности.

Полнота безопасности – уровень удовлетворительного выполнения системой, связанной с безопасностью, требуемых функций безопасности при всех заданных условиях в течение заданного периода времени. Чем выше уровень полноты безопасности систем, связанных с безопасностью, тем меньше вероятность отказа этих систем при выполнении ими требуемых функций безопасности. При определении полноты безопасности следует учитывать все причины отказов, ведущих к опасному состоянию, например, отказы аппаратных средств, отказы, вызванные программным обеспечением, сбоями аппаратуры, ошибками операторов.

Введение самого понятия «полнота безопасности» и характеристик полноты безопасности следует рассматривать как крупное достижение в решении проблем анализа и синтеза функциональной безопасности программно-аппаратных комплексов управления. Дело в том, что опасные отказы – очень

редкие события. Оценить реальное состояние функциональной безопасности информационных систем управления по статистическим данным об опасных отказах часто невозможно. Для сложных современных средств информационных систем управления характерна возможность сложных многократных комбинаций событий, вероятность каждого из которых мала, а в сумме их набирается немало. Работа таких систем зависит подчас от деятельности нескольких операторов, включая обслуживающий персонал, их квалификации и мастерства. При дальнейшем внедрении современной техники вопросы ее надежности и безопасности, дисциплины и организованности персонала приобретают первостепенное значение. Таким образом, есть ряд факторов, для которых возможна лишь их качественная, но не точная количественная оценка. Это относится, в первую очередь, к программному обеспечению, к отказам, вызванным ошибками операторов. Из-за совокупности этих причин был разработан и стандартизован качественно-количественный подход к оценке функциональной безопасности систем, который определен как полнота их безопасности. Суть полноты безопасности в следующем. Все информационные системы, связанные с безопасностью, по качественным характеристикам и количественным значениям разделяются на несколько групп. В стандартах МЭК/ГОСТ Р 61508-12, МЭК 62278 и др. приводятся четыре уровня полноты безопасности (УПБ).

Качественные характеристики определяют технологию разработки, технические решения по обеспечению функциональной безопасности программных и аппаратных средств в системе и защите от ошибок операторов, внешних и внутренних возмущающих воздействий, а также меры по поддержанию функциональной безопасности в процессе эксплуатации системы. Например, по технологии разработки программное обеспечение (ПО) разделяется на уровни полноты безопасности по спецификации

требований к ПО, по составу требований к архитектуре ПО, по требованиям к интеграции программных и аппаратных средств и др.; по техническим решениям уровни полноты безопасности систем разделяются на одноканальные двухканальные, трехканальные технические средства и др., средства и системы, содержащие одноверсионное или многоверсионное ПО и т.д.

Первый уровень полноты безопасности (УПБ 1) достигается относительно легко, если на всех стадиях разработки системы и ее производства применяют требования стандартов качества. Это самый низкий уровень, однако, требующий использования хорошего опыта разработок. Заметим, что уровень УПБ 1 фигурирует в стандарте МЭК 61508-12 и в других документах как «не связанный с безопасностью».

Второй уровень (УПБ 2) достигается не на много сложнее, чем предыдущий, однако для его обеспечения требуется большее число проверок и испытаний. Этот уровень полноты безопасности требует хорошего проекта и практики применения в рамках требований стандарта ISO 9001. В результате повышается стоимость системы.

Для достижения третьего уровня полноты безопасности (УПБ 3) требуются более существенные усилия и более высокая компетенция разработчиков, чем для первого и второго уровней, применение многоканальных аппаратных и многоверсионных программных средств. Важными факторами являются стоимость системы и время ее разработки. При этом выбор исполнителей становится ограниченным, так как немногие из них способны обеспечить этот уровень.

Четвертый уровень (УПБ 4) требует проведения разработки «на грани искусства», включая применение «формальных методов» [8, 55 и др.]. Стоимость проекта будет предельно большой и при создании системы потребуется исключительно высокая компетентность. В ряде случаев удается избегать применения

четвертого уровня безопасности, дополнительно используя уровни защиты.

Для связанных с безопасностью систем применяют несколько количественных показателей функциональной безопасности. Например, на железнодорожном транспорте распространены такие показатели, как интенсивность опасных отказов, их вероятность, вероятность безопасной работы за заданное время, средняя наработка до опасного отказа, средняя наработка на защитный отказ, коэффициент безопасности. При этом рекомендуется определять эти параметры экспериментально, расчетным путем или с помощью моделирования. Как уже ранее отмечалось, опасный отказ – это редкое событие. Для определения его вероятностных параметров экспериментальными методами потребуется время, значительно превышающее время жизни исследуемого устройства.

Математическое моделирование процессов возникновения опасных отказов является мощным инструментом исследования устройств и систем управления на соответствие требованиям безопасности. Для реализации моделирования необходимо соответствующее математическое описание объекта исследования – процесса появления опасных отказов, что не может быть в полной мере реализовано в силу высказанных ранее причин.

Для создания сложных многоуровневых информационных систем управления необходимо выработать комплексный подход к рациональному использованию аналитических и экспериментальных способов и методов доказательства безопасности, объединить разнородную информацию для получения достоверных оценок доказательства функциональной безопасности таких систем. С этой целью целесообразно сочетать результаты математического моделирования с ускоренными натурными испытаниями, результатами экспертизы технической и конструкторской документации, испытаниями имитационных моделей

программно-аппаратных средств, стендовыми испытаниями, а также с оценками безопасности по статистическим данным об отказах в процессе эксплуатации.

При первоначальном определении требований к функциональной безопасности возможно, что некоторые конструктивные характеристики противоречивы или принципиально нереализуемы в данном проекте. Кроме того, заданные заказчиком ограничения ресурсов не всегда могут учитывать ряд особенностей этого проекта. В результате не сбалансированные требования к безопасности и доступные ресурсы проявятся как риски – ущерб в виде потерь в уровне функциональной безопасности или в перерасходовании ресурсов. Для устранения или снижения рисков до допустимых пределов потребуется изменение требований к функциональной безопасности и/или к некоторым конструктивным характеристикам. Таким образом, целесообразно анализ и разработку требований к системам, связанным с безопасностью, проводить в два этапа: предварительно повышая безопасность, а затем, снижая требования к функциональной безопасности до значений (показателей), при которых риски не выходят за пределы установленных допустимых уровней или используемых ресурсов.

Требования к функциональной безопасности должны быть предварительно проверены разработчиками на их реализуемость с учетом доступных ресурсов конкретного проекта и при необходимости откорректированы по составу и значениям с учетом рисков.

Количественные значения уровней полноты безопасности системы, представленные показателями интенсивности опасных отказов и вероятности опасного отказа в течение часа работы, приведены в табл. III.5.1

Отметим, что если выразить значения требований к УПБ при высокой интенсивности запросов, приведенные в средней колонке таблицы, в других единицах – через «число отказов в

год», то средняя и правая колонка совпали бы численно. Такое представление могло бы вызвать путаницу, поскольку это принципиально разные параметры, имеющие даже различные размерности. В связи с этим, чтобы избежать указанной путаницы, и была выбрана иная размерность (число отказов в час).

Причина того, что требования к УПБ задаются фактически двумя таблицами (для высокой и для низкой интенсивности запросов) состоит в том, что для задания уровня полноты безопасности могут потребоваться оба упомянутых выше подхода. Разницу между ними лучше пояснить на примерах.

Таблица III.5.1. Уровни полноты безопасности (УПБ)

Уровень полноты безопасности (SIL)	**При высокой интенсивности запросов (опасных отказов в час)** $Q_{оп}(1)=\lambda_{оп}$	**При низкой интенсивности запросов (вероятность отказа)** $Q_{оп}(1)=\lambda_{оп}$
4	от 10^{-9} до 10^{-8}	от 10^{-5} до 10^{-4}
3	от 10^{-8} до 10^{-7}	от 10^{-4} до 10^{-3}
2	от 10^{-7} до $< 10^{-6}$	от 10^{-3} до 10^{-2}
1	от 10^{-6} до $< 10^{-5}$	от 10^{-2} до 10^{-1}

Рассмотрим операционную систему АРМ поездного диспетчера на железнодорожном транспорте. Здесь важна интенсивность ее отказов, поскольку каждый раз, когда отказ происходит, есть риск попасть в аварийную ситуацию. Этому случаю соответствует средняя колонка табл. III.5.1. Рассмотрим теперь прикладную программу поддержки принятия решений руководителем в чрезвычайных ситуациях. К такому средству защиты предъявляются низкие требования в том смысле, что потребность в нем возникает нечасто. Оно может оставаться невостребованным годы и даже десятки лет. Частота его отказов представляет

малый интерес для описания полноты безопасности, поскольку отказ программы поддержки принятия решений непосредственно не связан с возникновением аварийной ситуации.

При этом следует принять во внимание интервал между проверками. Другими словами, поскольку реальная потребность в этой программе возникает нечасто, то отказы, возникшие в период между ее проверками, могут остаться незамеченными. Интерес представляет комбинация частоты появления и длительности существования отказов. Таким образом, мы оцениваем УПБ с помощью значений вероятности отказа при наличии запроса. Эту величину и характеризует правая колонка табл. III.5.1.

Внимание читателя следует обратить на то обстоятельство, что интенсивность опасных отказов в единицу времени (час) равна с высокой точностью вероятности опасного отказа системы в течение 1 часа работы. Покажем это.

Поток опасных отказов есть многократно разреженный случайным образом поток исходных отказов системы. В соответствие с теорией редеющих потоков Реньи и теоремой Григелиониса [49] любой поток случайных событий, который многократно разрежен, случайным образом преобразуется в простейший поток с интенсивностью опасных отказов. Следовательно, случайное время между опасными отказами (можно уверенно полагать, что это также справедливо и относительно времени до опасного отказа) распределено по экспоненциальному закону. Таким образом,

$$Q_{\text{ОП}}(t) = 1 - \exp(-\lambda_{\text{ОП}} \cdot t).$$

Ограничившись первыми тремя членами разложения экспоненциальной функции в ряд, находим, что

$$Q_{\text{ОП}}(t) = 1 - 1 + \lambda_{\text{ОП}} \cdot t - \frac{\lambda_{\text{ОП}}^2 t^2}{2}.$$

При $t = 1$ ч и при общепринятом условии, что интенсивность опасных отказов измеряется величиной $\lambda_{оп}<10^{-5}$(см. табл. III.5.1), находим с погрешностью $\delta<(\lambda^2_{оп}/2)$, что $Q_{оп}(1)=\lambda_{оп}$. Эта погрешность значительно меньше той погрешности, которая допустима в инженерной практике (5% – 10%). Поэтому вполне допустимо одинаковыми числами оценивать интенсивность опасных отказов и вероятность опасного отказа в течение часа работы системы.

III.5.3. Принципы безопасности

Теория функциональной безопасности изучает комплекс вопросов, связанных с определением возможности возникновения аварий в результате отказов технических средств, действий персонала или недопустимых воздействий внешней среды. К наиболее важным задачам, решаемым теорией функциональной безопасности, относится разработка методов защиты от аварий и борьба с их последствиями. При этом необходимо рассматривать не только поведение системы в экстремальных ситуациях, но и категории угроз безопасности (в том числе военные, экологические, социальные), методы снижения негативных последствий от аварий, правовое и экологическое регулирование безопасности и другие вопросы, связанные с функционированием информационных систем управления.

Концепция обеспечения функциональной безопасности основывается на том, что информационная система управления должна:

– во-первых, содержать такие функции и средства их реализующие, которые с высокой вероятностью не допускают возникновение опасных отказов;

– во-вторых, обладать такой реакцией на возникший опасный отказ, что последствия его не приводят к недопустимому

ущербу. Например, отказ обнаружен и устранен за допустимое время или выполнен безопасный останов системы (система переведена в состояние защитного отказа).

В работе Х. Шебе и Й.-Т. Гайена «Неправильное понимание принципов безопасности» [82] описаны принципы обеспечения безопасности. Рассмотрим эти и другие принципы.

III.5.3.1. Принцип отказобезопасности.

Этот принцип предназначен для управления отказами. Он определяет способность системы управления совместно с объектом управления оставаться в безопасном состоянии или переходить в безопасное состояние в случае возникновения отказа. На рис. III.5.2 наглядно продемонстрирован этот принцип

Согласно стандартам [8, 9, 56, 57 и др.], в которых сосредоточены принципы и практические способы обеспечения функциональной безопасности систем управления, принцип отказобезопасности определяется как «выйти из работы безопасным образом или выйти из работы без опасных последствий». Заметим, что в этих стандартах установлено требование обеспечения допустимой (заданной) интенсивности опасных отказов даже при наличии в системе управления совместно с объектом управления одного случайного отказа.

Принцип может быть реализован тремя различными способами [82]:

– свойственная системе отказобезопасность;

– реактивная отказобезопасность;

– составная или комбинированная отказобезопасность.

Обратите внимание, что слово «составной» означает, что в системе имеются два канала, которые используются для перекрестных проверок. Это не имеет никакого отношения к избыточности путем дублирования. Обязательным условием реализации принципа безопасности есть условие отсутствия опасных последствий.

Рис. III.5.2. Строгое представление принципа отказобезопасности

Свойственная отказобезопасность

Система разрабатывается согласно свойственному системе принципу отказобезопасности, если все возможные режимы отказа не приводят к опасным последствиям. Физические, химические, информационные и другие процессы, которые свойственны системе, гарантируют это. Система всегда остается в безопасных состояниях, даже если происходит отказ. Если эти предположения выполнены, может быть создана безопасная система. Так как свойственная отказобезопасность систем основана на свойствах, которые не могут исчезнуть, такие системы не могут выполнять усложненные функции. Это вызвано тем, что функции безопасности являются простыми, и никакие сложные решения ими не могут быть приняты.

Отсюда следует, что:

– никакая свойственная отказобезопасность не может существовать, без сохранения в системе безопасного состояния или перехода к другим безопасным состояниям;

– никакой отказобезопасности не может быть без существования защитного состояния (безопасного состояния останова).

Примером такого безопасного состояния в системе может рассматриваться мажоритарное резервирование с логикой 2/3.

В случае отказа любого одного из трех однотипных элементов система остается в безопасном состоянии, т. к. при условии совпадения результатов на выходе двух работоспособных элементов считывается правильная информация. Если же эти результаты не совпадают, то система переводится в защитное состояние.

Реактивная, составная или комбинированная отказобезопасность

Реактивная отказобезопасность характеризуется быстрым обнаружением любой отказавшей функции информационной системы управления и активной реакцией по переводу системы в безопасное состояние. Хотя обычно имеется только одна подсистема для функций управления, но может быть и вторая подсистема для обнаружения неисправностей, диагностики и испытаний. Примером такого типа системы является система с безопасной функцией контроля или наблюдения. Становится ясно, что этот вид систем нуждается в механизме обнаружения, который в состоянии обнаружить с достаточным охватом все опасные отказы за допустимое время перерыва в работе. Кроме того, необходим механизм реагирования, активной реакцией переводящий систему в защитное состояние.

Системы, спроектированные согласно принципу комбинированной отказобезопасности, должны отвечать следующим условиям:

– в составе системы должно содержаться несколько подсистем (по крайней мере, две), которые обеспечивают ту же самую функцию безопасности;

– эти подсистемы должны быть независимыми друг от друга;

– результаты работы всех этих подсистем должны быть идентичными (комбинированная отказобезопасность со сравнением), или большинство подсистем должно иметь одинаковый результат (решение голосованием большинства);

– обнаружение отказа одной подсистемы и последующая реакция должно быть достаточно быстрыми, чтобы вероятность отказа второй подсистемы была незначительна;

– должно существовать защитное состояние;

– процедуры сравнения или решение, принятое большинством голосов должны быть выполнены в отказобезопасном режиме и достаточно быстро.

Сравнительный анализ возможностей по обнаружению отказов свидетельствует о том, что только системы с комбинированной отказобезопасностью в состоянии обнаруживать случайные отказы аппаратуры и систематические ошибки. Это обусловлено тем, что такие системы построены из нескольких подсистем. Отсюда очевидно, почему для более высоких уровней безопасности рекомендуется создавать системы из нескольких подсистем, так как обнаружение отказа очень важно для обеспечения безопасности.

Идентификация отказов является важным свойством систем. Для систем с реактивной отказоустойчивостью и систем с избыточными подсистемами отказы должны быть оперативно обнаружены в течение допустимого времени, чтобы активизировать реакцию, которая выключает систему, включает подсистемы или активизирует резервные подсистемы. Степень обнаружения отказа (охват отказов) строго влияет на уровень безопасности, который может быть достигнут данной системой. Это является одной из причин, из-за которой стандарт [57] устанавливает минимальный уровень охвата отказов для каждого уровня безопасности.

Чтобы определить полноту обнаружения отказов в системе, необходимо проанализировать, какие отказы могут произойти, и какими средствами и в какой степени они могут быть обнаружены. Кроме того, должны быть учтены отказы программы-агента. Следовательно, необходимо выполнить распределение отказов по группам [82], как показано на рис. III.5.3. Отказ про-

исходит с определенной вероятностью. Эта вероятность должна быть разделена на часть опасных отказов и часть тех отказов, которые не опасны, согласно различным режимам отказа. Это должно быть обосновано анализом, например, FMEA (анализ характера и последствий отказов). На рис. III.5.3 опасные отказы показаны в сером цвете. Они являются или необнаруженными отказами или обнаруженными опасными отказами, приведшими к аварии объекта управления.

Необнаруженные безопасные отказы сами по себе не опасны, но если они накапливаются, могут быть запрещены или ухудшены необходимые системные функции (на рис. III.5.3 они показаны светло серым цветом). Несколько необнаруженных безопасных отказов вместе могут уменьшить способность обнаружения отказа системы, что может привести к тому, что произойдут опасные отказы системы.

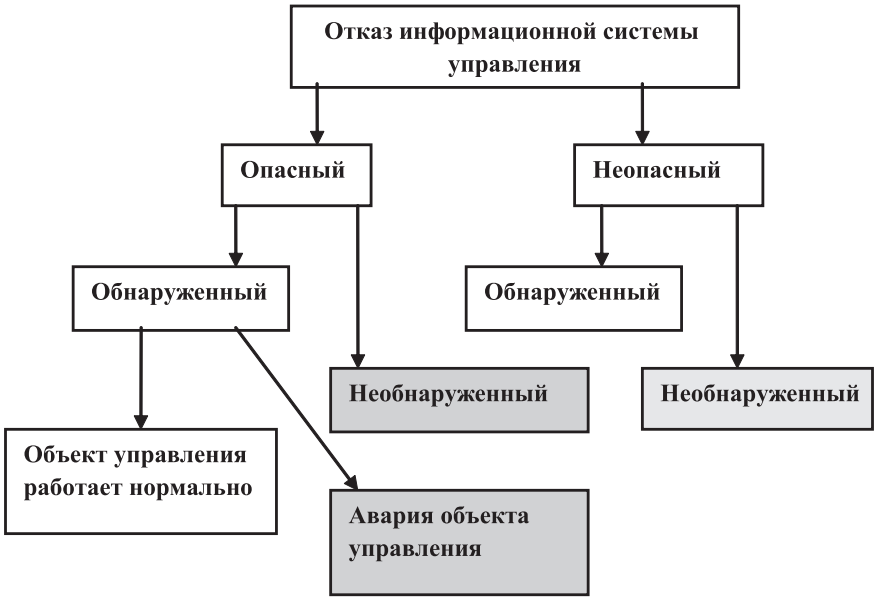


Рис. III.5.3. Распределение отказов системы управления по группам их состояний

III.5.3.2. Принцип избыточности

Этот принцип связан с введением избыточности в создаваемые элементы, узлы, устройства и системы. Избыточность может быть параметрической (в узле имеется запас прочности), схемной (в устройство вводятся безопасные логические элементы, компараторы, ключи и др.), структурной (дублирование, троирование в устройстве или системе аппаратных средств, функциональных узлов и элементов), программной (для выполнения задачи используют, например, два независимых программных продукта), функциональной (одну и ту же задачу решают с помощью полной или упрощенной функции, но с меньшей точностью), информационной (информация кодируется внутри системы с последующим декодированием и проверкой ее безошибочности перед использованием), временной (увеличено время восприятия или выдачи воздействия), комбинированной (использовано несколько из перечисленных методов). Таким образом, требования безопасности накладывают дополнительные условия на комплектующие изделия и материалы, конструкцию, на схемные решения и структуру системы, на представление информации в ней и др. Комплексное сочетание нескольких видов избыточности позволяет сформировать многоуровневую избыточность. Детальное рассмотрение каждого вида избыточности приведено в п. III.1.5.

III.5.3.3. Принцип разнообразия

Следует отметить, что в системах с комбинированной отказобезопасностью обнаружение неисправностей не происходит автоматически. Обнаружение неисправностей осуществляется только в той степени, в какой степени подсистемы разнообразны. Это означает, что комбинированные системы с разнообразными подсистемами обнаруживают только определенные ошибки из-за вида разнообразия.

Разнообразие характеризует ситуацию, где несколько подсистем выполняют ту же самую функцию различным способом. Разнообразие может быть относительно процесса разработки, алгоритма, программного обеспечения, аппаратных средств и т. д. Принцип разнообразия рекомендуется для сложных отказоустойчивых и избыточных систем. В большинстве случаев, сравнения и принятие решения большинством голосов могут присутствовать только на функциональном уровне, вызванном различиями между подсистемами.

Существуют *следующие виды разнообразия:*

- разнообразные группы разработчиков;
- разнообразие проекта;
- разнообразие аппаратных средств;
- разнообразные операционные системы;
- разнообразие прикладного программного обеспечения ;
- разнообразные языки программирования;
- разнообразные алгоритмы и др.

Основное назначение принципа разнообразия состоит в том, чтобы исключить отказы по общим причинам в максимально возможной степени, так как они приводят к опасному отказу системы. Отказы по общей причине нужно отличать от общего режима отказов. Это не всегда делается, иногда оба термина используются как синонимы. Отказ по общей причине представляет собой общий отказ в нескольких подсистемах, которые являются независимыми [8]. В стандарте [9] отказ по общей причине определяется как отказ, вызванный одним или более событиями, которые приводят к отказам в двух или более отделенных подсистемах системы, приводя к системному отказу. Отказом по общей причине может быть короткое замыкание, которое приводит к отказу электропитания всех подсистем. Причина может состоять в том, что источники питания подсистем недостаточно разделены. Отказ по общей причине – это систе-

матический отказ, который, однако, может быть вызван случайным отказом, служащим инициатором. Принцип разнообразия рекомендуется применять в отношении отказов и ошибок в функционировании, которые не являются результатом спецификации требований, но возникают позже в разработке и производственном процессе. Эти неисправности можно парировать, выбирая правильные методы разнообразия. Например, функциональные отказы блоков управления могут быть устранены путем применения различных типов процессоров. Ошибка аппаратных средств в одиночном процессоре может быть вскрыта при использовании разнообразного программного обеспечения при выполнении тех же самых или эквивалентных задач.

III.5.3.4. Принцип локализации развития неблагоприятных процессов

В соответствии с данным принципом обеспечение безопасности достигается применением средств, локализующих развитие неблагоприятных процессов. Эти средства защищают систему от выдачи неправильных воздействий, предупреждают о возможном наступлении экстремальных ситуаций и управляют функционированием объекта в критических случаях (парируют развитие отказа и переводят объекты управления в защитное состояние). Для этих целей используются контролирующие и диагностирующие устройства, которые оценивают значения выходных параметров системы и значения специальных диагностических признаков, а в необходимых случаях и окружающей среды (вибрации, температуры, электромагнитной обстановки и др.). Сравнивать измеренные сигналы с их заданными значениями, обрабатывать информацию и принимать решения о необходимых действиях для предотвращения аварийной ситуации должны устройства, которые сами обладают высокой достоверностью, т. е. отвечают требованиям безопасности.

III.5.4. Нормирование допустимого времени существования опасного отказа

Нормирование допустимого времени существования опасного отказа в информационных системах управления ответственными и критически важными объектами управления требует взвешенного подхода. С одной стороны, завышение допустимого времени может привести к катастрофическим последствиям. С другой стороны, чрезмерное занижение этого времени влечет за собой значительное увеличение затрат на повышение реактивности средств обеспечения отказобезопасности и вследствие этого ограничивает возможности по повышению надежности систем управления массового изготовления. При этом необходимо учитывать специфику производства и эксплуатации информационных систем управления во многих недостаточно развитых промышленных странах, которая заключается в необходимости создавать безопасные системы управления за счет оригинальных и эффективных технических решений при сравнительно низкой надежности составных элементов.

Для нормирования допустимого времени существования опасного отказа должны быть построены корректные математические модели, получены из них наглядные и удобные для практического использования аналитические выражения, которые обеспечивают чувствительность показателей безопасности к надежности составных устройств систем управления и согласованы с требованиями существующих стандартов.

Решению указанной проблемы посвящено ряд работ. Одни из первых публикаций связаны с системами автоматики и телемеханики, осуществляющими управление движением поездов. Это работы С. Адомита [8], Д. Брабанда [73, 74 и т. д.], Дж. Кольера [64], Е. Н. Розенберга [70, 75 и т. д.], Вл. и В. Сапожнико-

вых [65, 66 и т. д.], Д.В. Шалягина [69, 72 и т. д.], В. Швира [66, 76], Х. Шебе [35, 67, 68 и т. д.], И.Б. Шубинского [35, 69, 70 и т. д.] и других авторов. Так, В.Швиром установлено, что

$$a\overline{t_0} \le 10^{-3},$$

где a – сумма интенсивностей отказов рассматриваемых узлов, совместный отказ которых может быть опасным, а $\overline{t_0}$ – среднее время обнаружения первого отказа. В работе [67] также определено среднее время до второго (опасного) отказа двухканальной системы управления

$$T_{\text{ВТОР}} = \frac{1}{a_1 a_2 \overline{t}_0},$$

где α_1 означает интенсивность первого отказа, α_2 – интенсивность второго отказа.

Модель В. Швира включена в стандарт Германии Mü 8004 применительно к системам, критичным по безопасности, построенным по двухканальной и трехканальной архитектуре.

В моделях С. Адомита, представленных фирмой Сименс (Siemens), и включенных в стандарт [8], предлагается оценивать допустимое время обнаружения двойного отказа по формуле $\tau_{\text{ДОП}}=(2/3\lambda)$ (λ – интенсивность отказов одного канала), полученной при условии $h = 0{,}1$ (доля опасных отказов от общего числа двойных отказов).

Вместе с тем, в моделях В. Швира и С. Адомита имеются некоторые ограничения. Основное из них то, что при обосновании требований к допустимому времени обнаружения опасного отказа не учитывается классификация уровней безопасности, а именно частотная составляющая показателя риска – вероятность наступления опасного отказа за час эксплуатации, как это определено стандартами [6, 57].

Нами разработаны следующие модели для оценки допустимого времени обнаружения опасного отказа применительно к двухканальным и трехканальным системам, которые можно интерпретировать как уточнения указанных выше моделей в отношении заданных требований по уровням полноты безопасности.

III.5.4.1. Оценка допустимого времени обнаружения одиночного опасного отказа

Система 2x2

Пусть система содержит два однотипных канала, объединенных безопасным и идеально надежным устройством сравнения (система типа 2x2). Каналы также охвачены непрерывным контролем неисправностей, который предполагается идеально надежным с эффективностью правильного обнаружения отказов на уровне вероятности, равной 1 при условии, что для достижения такой эффективности требуется определенное время обнаружения отказа t_0, которое в общем случае имеет случайный характер. При обнаружении отказавшего одного канала система продолжает работать с исправным каналом, а отказавший канал восстанавливается. В модели не рассматривается перевод системы в защитный отказ, потому что решается другая задача, а именно задача оценки влияния времени обнаружения отказа канала на возможность возникновения опасного отказа. Опасный отказ двухканальной системы управления наступает при отказе обоих каналов. Возможность возникновения опасного отказа определяется показателем вероятности опасного отказа в течение одного часа работы системы. Выбор этого интервала времени обусловлен тем, что в этом случае появляется возможность связать время обнаружения одиночного опасного отказа с требованием по уровню полноты безопасности.

Исходные предпосылки следующие. Потоки отказов, восстановлений и обнаружений отказов являются простейшими с параметрами λ, μ и $\gamma = 1/\overline{t}_0$ соответственно ($\overline{t}_0$ – среднее время обнаружения неисправности). Возможные сбои каналов в системе не учитываются. Система восстанавливается одной ремонтной бригадой.

При отказе одного канала система продолжает выполнять свои задачи с пониженным уровнем безопасности. Опасный отказ системы наступает в том случае, когда во время обнаружения или восстановления первого отказа наступит второй отказ (отказ второго канала). Построим граф состояний двухканальной системы (рис. III.5.4), где состояние 0 – оба канала исправны; состояние 1 – отказал один канал и происходит обнаружение отказа с интенсивностью γ; состояние 2 – отказал один канал, отказ обнаружен и происходит восстановление с интенсивностью μ, либо возник отказ второго канала с интенсивностью λ; состояние 3 – отказали оба канала (опасный отказ) – поглощающее состояние.

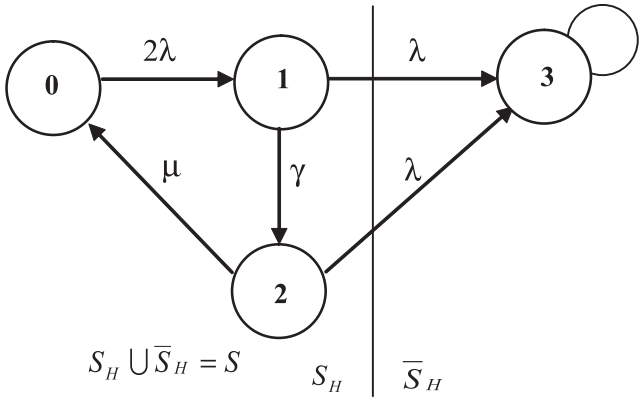


Рис. III.5.4. Граф состояний двухканальной системы с двойным отказом

Состояния 0, 1 и 2 относятся к подмножеству состояний $S_{\text{Н}}$ отсутствия опасных отказов.

Определим среднее время пребывания системы $T_{\text{ОП}}$ в подмножестве состояний $S_{\text{Н}}$ – это и будет среднее время до опасного отказа системы (то есть среднее время до попадания в состояние 3).

Для решения поставленной задачи воспользуемся разработанными нами графовыми полумарковскими методами [1, 70, 71].

Одной из форм задания полумарковского процесса поведения рассматриваемой системы является матрица математических ожиданий $T = (T_i)$ безусловных времен пребывания системы в каждом из состояний $i = 0…3$ и матрица переходных вероятностей $\prod=(p_{ij})$ переходов системы из i-го состояния в j-е в установившемся режиме. Этого будет достаточно для определения среднего времени $T_{\text{ОП}}$ до опасного отказа.

В нашем случае модель, показанная на рис. III.5.4, является Марковской. Тогда исходные данные T_i и p_{ij} определяются совсем просто. Времена T_i находятся путем деления 1 на сумму интенсивностей выходов из i-го состояния, а переходные вероятности p_{ij} находятся путем деления интенсивности перехода из i-го состояния в j-ое на сумму интенсивностей выходов из i-го состояния, то есть

$$T_0 = \frac{1}{2\lambda};\ T_1 = \frac{1}{\lambda+\gamma};\ T_2 = \frac{1}{\gamma+\mu};\ p_{01} = \frac{2\lambda}{2\lambda} = 1;\ p_{12} = \frac{\gamma}{\lambda+\gamma};$$

$$p_{13} = \frac{\lambda}{\lambda+\gamma};\ p_{20} = \frac{\mu}{\lambda+\mu};\ p_{23} = \frac{\lambda}{\lambda+\mu}.$$

Вес контура 0-1-2-0 равен $C_0=1–p_{01}p_{12}p_{20}$.

В соответствии с топологическим методом [71], непосредственно по графу состояний находим в множестве $S_{\text{Н}}$ среднее время до опасного отказа системы при условии, что начальным является состояние 0:

$$T_{ОП} = \frac{T_0 + p_{01}T_1 + p_{01}p_{12}T_2}{1 - p_{01}p_{12}p_{20}} = \frac{\frac{1}{2\lambda} + 1 \cdot \frac{1}{\lambda + \gamma} + 1 \cdot \frac{\gamma}{(\lambda + \gamma)} \frac{1}{(\lambda + \mu)}}{1 - 1 \cdot \frac{\gamma}{(\lambda + \gamma)} \frac{\mu}{(\lambda + \mu)}} =$$

$$= \frac{3}{2\lambda} + \frac{\gamma\mu}{2\lambda^2(\lambda + \gamma + \mu)}. \qquad (5.1)$$

Проверим корректность полученного результата. Если неисправность обнаруживается мгновенно (то есть $\gamma \to \infty$), то $\lambda >> \mu$, $\gamma >> \lambda$, и мы получим классическую формулу среднего времени до отказа дублированной системы с нагруженным резервом, восстановлением, одной ремонтной бригадой, идеально надежным и эффективным контролем и мгновенным временем переключения на резерв.

Введем следующие обозначения

$$a_1 = \frac{2\lambda}{3}; \mu = \frac{1}{T_В}; \gamma = \frac{1}{\overline{t_0}},$$

где a_1 – интенсивность первого отказа, $T_В$ – среднее время восстановления отказавшего канала.

Применительно к поставленной задаче обоснования требования к предельно допустимому времени обнаружения первого отказа $\tau_{ДОП}$ преобразуем выражение (5.1), имея в виду, что в информационных системах управления $\gamma >> \lambda$, $\gamma >> \mu$ и $\tau_{ДОП} \le \overline{t_0}$:

$$T_{ОП} \ge \frac{1}{a_1} + \frac{2}{9a_1^2\tau_{ДОП}}. \qquad (5.2)$$

Для технических систем, к которым в соответствии со стандартами [55, 56, 57 и др.] предъявляются требования по функциональной безопасности на уровне полноты безопасности не ниже УПБ 1 (SIL1), вероятность опасного отказа системы в соответствии с ее оценкой, приведенной в п. III.5.2, с высо-

кой точностью может быть определена по следующей формуле $Q_Б(1) \cong \lambda_{ОП} \cong 1/T_{ОП}$. Точность этой формулы будет тем выше, чем более высокие требования предъявляются к функциональной безопасности систем. Следовательно, $T_{ОП} \cong 1/Q_Б(1)$.

Используя формулу (5.2) и учитывая, что оценивается только обнаружение без последующего устранения отказа, получаем искомое условие

$$\tau_{ДОП} \le \frac{0,2 \cdot Q_Б(1)}{a_1 (a_1 - Q_Б(1))} \cong \frac{0,2 \cdot Q_Б(1)}{a_1^2}, \qquad (5.3)$$

поскольку на практике $\alpha_1 >> Q_Б(1)$.

Это неравенство, в соответствии с достигнутым уровнем технического развития, требованиями стандартов, реальными возможностями восстановления отказов в системе и реальными возможностями получения исходных данных, позволяет определить максимально допустимое время обнаружения отказа.

Система 2∨3

Определим предельно допустимое время обнаружения первого отказа для модели безопасности системы, построенной по мажоритарному способу с логикой 2 из 3 (2∨3). Предполагается, что все три канала однотипны, восстанавливающий орган безопасен и идеально надежен. Контроль в системе осуществляется попарным сравнением результатов на выходе каналов системы. Время сравнения результатов и выполнения функции восстанавливающим органом пренебрежимо мало. В системе предусматривается восстановление отказавших каналов одной ремонтной бригадой. Возможные сбои в каналах системы не учитываются. Восстанавливающий орган предполагается идеально надежным.

При отказе одного канала система продолжает выполнять свои задачи, если совпадают результаты работы двух оставшихся исправных каналов. Опасный отказ системы наступает в

случае несовпадения результатов вследствие отказа любых двух каналов.

По описанной выше методике с помощью графового метода в предположении простейших потоков отказов и восстановлений получим неравенство для определения допустимого времени обнаружения первого отказа в системе управления, построенной по логике 2∨3, в которой опасными могут быть отказы двух каналов.

$$\tau_{\text{ДОП}} \le \frac{0,2 \cdot Q_{\text{Б}}(1)}{a_1^2}, \text{ где } a_1 = \frac{6\lambda}{5}. \qquad (5.4)$$

Примечательно то, что полученные для обеих рассмотренных моделей результаты имеют общий характер и сводятся к тому, что можно рекомендовать простую практическую формулу расчета:

$$\tau_{\text{ДОП}} \le \frac{0,2 \cdot Q_{\text{Б}}(1)}{a_1^2}, \qquad (5.5)$$

где $\alpha_1=(2\lambda/3)$ – интенсивность первого отказа системы 2x2, а $\alpha_1=(6\lambda/5)$ – интенсивность первого отказа системы 2∨3

III.5.4.2. Оценка допустимого времени обнаружения двойного отказа

Оценим теперь требования, которые должны предъявляться к времени обнаружения двойного отказа, когда опасными становятся тройные отказы. С этой целью рассмотрим модель трехканальной системы в виде графа состояний, который показан на рис. III.5.5.

Состояния 0, 1, 2 – работоспособные; состояние 5 – состояние неопасного тройного отказа; состояние 6 – состояние опасного тройного отказа, который составляет долю h ($0 \le h \le 1$) от тройных отказов.

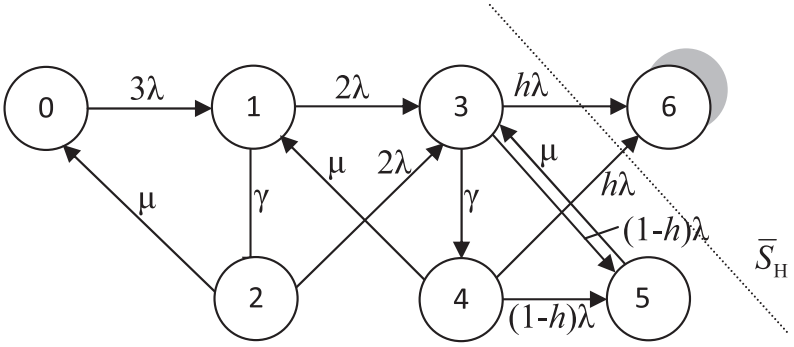


Рис. III. 5.5. Граф состояний трехканальной системы с тройным отказом

В данном случае модель, показанная на рис. III.5.5, является Марковской. Тогда исходные параметры T_i и p_{ij} в множестве неопасных состояний $S_{\text{Н}}$ определяются в виде:

$$T_0 = \frac{1}{3\lambda}; T_1 = \frac{1}{2\lambda + \gamma}; T_2 = \frac{1}{2\lambda + \mu}; T_3 = \frac{1}{\lambda + \gamma}; T_4 = \frac{1}{\lambda + \mu}; T_5 = \frac{1}{\mu};$$

$$p_{01} = 1; p_{12} = \frac{\gamma}{2\lambda + \gamma}; p_{13} = \frac{2\lambda}{2\lambda + \gamma}; p_{20} = \frac{\mu}{2\lambda + \mu}; p_{23} = \frac{2\lambda}{2\lambda + \mu};$$

$$p_{34} = \frac{\gamma}{\lambda + \gamma}; p_{35} = \frac{(1-h)\lambda}{\lambda + \gamma}; p_{41} = \frac{\mu}{\lambda + \mu}; p_{45} = \frac{(1-h)\lambda}{\lambda + \mu}; p_{53} = 1.$$

Веса контуров на данном графе в множестве неопасных состояний определяются следующим образом:

$$C_1 = p_{01}p_{12}p_{20}; C_2 = p_{34}p_{45}p_{53}; C_3 = p_{13}p_{34}p_{41}; \; C_4 = p_{12}p_{23}p_{34}p_{41}.$$

Заметим, что контуры C_1 и C_2 не имеют общих вершин, что следует учитывать при определении веса разложения графа без вершины 6.

В соответствии с графовым методом, непосредственно по графу состояний находим в множестве $S_{\text{Н}}$ среднее время до опасного отказа системы при условии, что начальным является состояние 0:

$$T_{\text{ОП}} = \frac{T_0(1-C_2-C_3-C_4)+p_{01}T_1(1-C_2)+p_{01}p_{12}T_2(1-C_2)}{1-C_1-C_2-C_3-C_4+C_1C_2} + \\ + \frac{(p_{01}p_{12}p_{23}++p_{01}p_{13})T_3+(p_{01}p_{12}p_{23}+p_{01}p_{13})p_{34}T_4}{1-C_1-C_2-C_3-C_4+C_1C_2} + \\ + \frac{(p_{01}p_{12}p_{23}+p_{01}p_{13})(p_{34}p_{45}+p_{35})T_5}{1-C_1-C_2-C_3-C_4+C_1C_2}. \qquad (5.6)$$

Применительно к поставленной задаче обоснования требования к предельно допустимому времени обнаружения двойного отказа $\tau_{\text{ДОП}}$, преобразуем выражение (5.6) в неравенство, имея в виду, что в информационных системах управления $\gamma >> \lambda$, $\gamma >> \mu$, $Q_{\text{Б}}(1) \cong \lambda_{\text{ОП}} \cong 1/T_{\text{ОП}}$ и $\tau_{\text{ДОП}} \le 1/\gamma = \bar{t}_0$.

В результате преобразования определяем предельно допустимое время обнаружения двойного отказа в виде следующего неравенства:

$$\tau_{\text{ДОП}} \le \left(\frac{Q_{\text{Б}}(1)}{h^2 a^3} \right)^{\frac{1}{2}}, \qquad (5.7)$$

где $\alpha=(11\lambda/5)$ – интенсивность двойных отказов (отказов двух каналов) трехканальной резервированной системы без восстановления.

III.5.4.3. Обсуждение результатов

Для проведения анализа применимости предлагаемых моделей и формул (5.5), (5.7) в сравнении с подобными формулами стандарта Мü 8004 соответственно, проведены расчеты в широком диапазоне значений интенсивности отказов устройств λ для

четырех уровней полноты безопасности, результаты которых сведены в табл. III.5.2.

В таблице приведены результаты расчетов в часах допустимого времени обнаружения одиночного отказа для систем 2x2 и 2∨3 (верхние строки) и двойного отказа для трехканальной резервированной системы (нижние строки).

В проведенных исследованиях так же, как и в работе [66], наблюдается тенденция к уменьшению допустимого времени обнаружения первого отказа при переходе от двухканальной системы 2x2 к мажоритарной системе 2∨3 при одинаковых предпосылках об идеальной безопасности и надежности компараторов. Чтобы соответствовать уровню безопасности двухканальной системы, трехканальная система с мажоритарной логикой должна обладать большей реактивностью на защиту от отказов. Это обстоятельство свидетельствует о том, что с позиций безопасности в системе с мажоритарной логикой не следует ограничиваться контролем исправности каналов на основе сравнения их выходных результатов, – целесообразно охватить каждый из трех однотипных каналов надежными, безопасными и эффективными средствами контроля, что позволило бы избежать опасного отказа даже в случае отказов двух из трех каналов при условии, что отказавшие каналы обнаружены.

В выделенных клетках табл. III.5.2 содержатся результаты расчетов по предложенным формулам, которые совпадают с расчетами по формулам Мü 8004 (а, следовательно, по формуле В. Швира для обнаружения одиночного отказа и по формуле С. Адомита для обнаружения двойного отказа соответственно).

Анализ выделенных клеток таблицы показывает, что в реальном диапазоне значений λ (от 10^{-7}до 10^{-4} 1/ч) первая Мü 8004 – формула (формула В. Швира) определения допустимого времени обнаружения первого отказа пригодна для систем с УПБ-2 ($10^{-7} \le Q_{Б}(1) \le 10^{-6}$), когда согласно стандартам [6, 8, 9 и др.]

возникновение отказа есть маловероятное, но возможное событие. Это в полной мере относится как к системе типа 2x2, так и к системе типа 2∨3. Для особо ответственных систем уровня УПБ-4 следует при определении допустимого времени $\tau_{доп}$ руководствоваться формулой (5.5). Вторая формула Мü 8004 также пригодна, главным образом, для систем с ограниченными требованиями к функциональной безопасности (УПБ-2, $10^{-7} \le Q_Б(1) \le 10^{-6}$), когда событие опасного отказа «может иметь место в какой-то момент жизненного цикла системы. Разумно ожидать, что данная опасность может возникнуть». При проектировании систем, к которым предъявляются повышенные требования по функциональной безопасности (УПБ-3 и УПБ-4) рекомендуется применять формулу (5.7) .

На рис. III.5.6. приведены два семейства графических зависимостей допустимого времени обнаружения отказов от близких к граничным интенсивностей каналов ($\lambda=10^{-7}$ 1/ч и $\lambda=10^{-4}$ 1/ч) и от

Таблица III.5.2

Q(1) λ_k(1/ч)	Модели допустимого времени обнаружения отказов Модели Мü 8004									
	10^{-9} УПБ-4		10^{-8} УПБ-3		10^{-7} УПБ-2		10^{-6} УПБ-1		2x2	2v3
	2x2	2v3	2x2	2v3	2x2	**2v3**	**2x2**	**2v3**		
10^{-7}	$4{,}5\cdot 10^{4}$	$1{,}4\cdot 10^{4}$	$4{,}5\cdot 10^{5}$	$1{,}4\cdot 10^{5}$	$4{,}5\cdot 10^{6}$	$1{,}4\cdot 10^{6}$	$4{,}5\cdot 10^{7}$	$1{,}4\cdot 10^{7}$	$5\cdot 10^{3}$	$3\cdot 10^{3}$
	$3{,}1\cdot 10^{6}$		$9{,}7\cdot 10^{6}$		$3{,}1\cdot 10^{7}$		$9{,}7\cdot 10^{7}$		**$6{,}7\cdot 10^{6}$**	
10^{-6}	$4{,}5\cdot 10^{2}$	$1{,}4\cdot 10^{2}$	$4{,}5\cdot 10^{3}$	$1{,}4\cdot 10^{3}$	$4{,}5\cdot 10^{4}$	$1{,}4\cdot 10^{4}$	$4{,}5\cdot 10^{5}$	$1{,}4\cdot 10^{5}$	$5\cdot 10^{2}$	$3\cdot 10^{2}$
	$9{,}7\cdot 10^{3}$		$3{,}1\cdot 10^{4}$		$9{,}7\cdot 10^{4}$		$3{,}1\cdot 10^{5}$		**$6{,}7\cdot 10^{5}$**	
10^{-5}	$4{,}5\cdot 10^{0}$	$1{,}4\cdot 10^{0}$	$4{,}5\cdot 10^{1}$	$1{,}4\cdot 10^{1}$	$4{,}5\cdot 10^{2}$	$1{,}4\cdot 10^{2}$	$4{,}5\cdot 10^{3}$	$1{,}4\cdot 10^{3}$	$5\cdot 10^{1}$	$3\cdot 10^{1}$
	$3{,}1\cdot 10^{3}$		$9{,}7\cdot 10^{3}$		$3{,}1\cdot 10^{4}$		$9{,}7\cdot 10^{4}$		**$6{,}7\cdot 10^{4}$**	
10^{-4}	$4{,}5\cdot 10^{-2}$	$1{,}4\cdot 10^{-2}$	$4{,}5\cdot 10^{-1}$	$1{,}4\cdot 10^{-1}$	$4{,}5\cdot 10^{0}$	$1{,}4\cdot 10^{0}$	$4{,}5\cdot 10^{1}$	$1{,}4\cdot 10^{1}$	**$5\cdot 10^{0}$**	$3\cdot 10^{0}$

уровня полноты безопасности многоканальной системы. Каждое из двух семейств включает в себя графические зависимости допустимого времени обнаружения одиночных отказов от УПБ для систем 2x2 и 2v3, построенные как по предложенным моделям (сплошная кривая и пунктирная кривая), так и по стандартным формулам (сплошная прямая и пунктирная прямая).

Кроме того, в состав каждого из двух семейств включены графические зависимости допустимого времени обнаружения двойного отказа от УПБ, построенные по формуле (5.6) (штрих-пунктирная кривая) и по стандартной формуле (штрих-пунктирная прямая).

Анализ графических зависимостей показывает, что при сравнительно низкой надежности элементов системы ($\lambda_к=10^{-4}$) расчеты допустимого времени обнаружения одиночных отказов, а также двойных отказов по стандартным формулам и предложенным (5.5) и (5.7) практически не отличаются во всем диапазоне значений УПБ, за исключением УПБ-1. Последнее обстоятельство не критично при проектировании безопасных информационных систем управления.

При высокой надежности элементов системы ($\lambda_к=10^{-7}$) расчеты по стандартным формулам приводят к значительно завышенным требованиям к допустимому времени обнаружения как одиночных, так и двойных отказов практически во всем диапазоне значений УПБ. В этих формулах отсутствуют различия к заданию требований по обнаружению отказов в зависимости от уровня полноты безопасности системы. В результате расчетные требования к допустимому времени обнаружения одиночных отказов на 2, а то и 3 порядка меньше тех, которые приемлемы для соответствующего уровня безопасности. Например, система управления с УПБ-4 должна на порядок быстрее реагировать на одиночный отказ, чем этого было бы достаточно по оценкам с помощью формулы (5.5). Для систем управления с УПБ-1 расхождение в этих оценках составляет три порядка. В отношении

определения допустимого времени обнаружения двойных отказов разброс оценок существенно меньше, однако также довольно значительный. Так, если для систем управления с УПБ-4 результаты расчетов по стандартной и предложенной формуле практически совпадают, то для систем с другими уровнями полноты безопасности расхождения достигают 5 – 10 раз. Основная причина в том, что в стандартной формуле не учитываются уровни полноты безопасности системы.

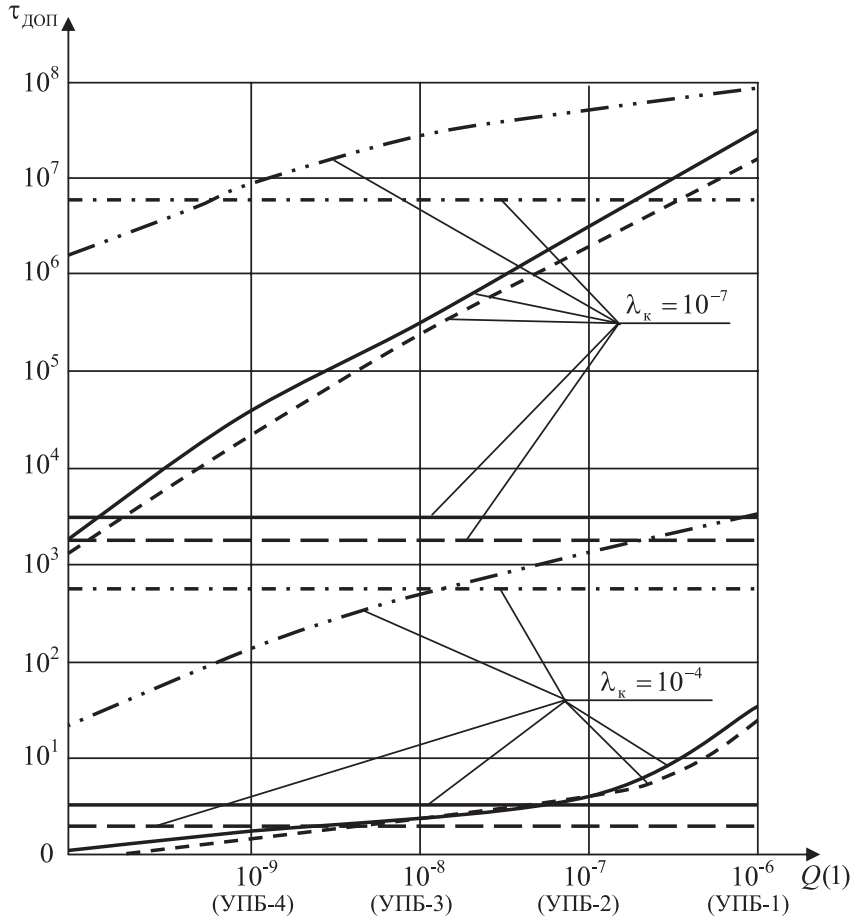


Рис. III.5.6. Графики зависимости допустимого времени обнаружения отказов от уровня полноты безопасности многоканальной системы и от интенсивности отказов каналов

Следует отметить, что расчеты по стандартным формулам приводят к формированию завышенных требований к системам, связанным с безопасностью. Это означает, что применение стандартных формул с присущими им ограничениями не повлияет на снижение функциональной безопасности проектируемых систем. Однако это влечет за собой неоправданные, порой значительные затраты ресурсов.

Предложенные модели и формулы (5.5) и (5.7) обеспечивают достаточно точную оценку допустимого времени обнаружения отказов и позволяют проводить расчеты в соответствии с требованиями стандартов [6, 8, 9 и др.] и, кроме того, позволяют учитывать уровень критичности к безопасности анализируемой системы при обосновании требований к оперативности обнаружения отказов в информационных системах управления.

III.5.5. Безопасность двухканальных систем

III.5.5.1. Введение

Известно большое количество вариантов организации связанных с безопасностью двухканальных информационных систем. Например, система содержит два идентичных и независимых канала обработки информации, а также средства диагностики, которые с периодичностью, меньшей допустимого времени обнаружения одиночных отказов, проверяют состояние работоспособности каждого канала и сравнивают их выходные результаты. При совпадении выходных результатов каналов считывается информация. Отказы каналов несимметричны. При исправных средствах диагностики обнаруживается факт отказа в работе любого одного канала, после чего осуществляется перевод системы в состояние защитного отказа. В случае отказа средства диагностики возникает неопасный отказ системы. Последующий за этим событием отказ канала

приводит к опасному отказу системы. Опасный отказ системы может произойти и в том случае, если средства диагностики пропустят отказ канала.

В работе [72] рассмотрено несколько вариантов построения архитектуры двухканальных систем. Одним из них является следующая архитектура системы. Решение задачи в системе производится одновременно двумя равноправными каналами обработки информации. На выходе каждого канала формируются общие контрольные сигналы, которые сравниваются между собой схемой сравнения (обычно эта схема представляет собой безопасный компаратор). При положительном результате сравнения, когда сигналы совпадают, схема сравнения отпирает ключ, и выход первого канала соединяется со входом объекта управления. При несовпадении сигналов схема сравнения запирает ключ, и система управления переходит в состояние защитного отказа. Особенность этого варианта организации двухканальной системы состоит в том, все элементы системы периодически контролируются с помощью внешнего (функционального) контроля. Период контроля на несколько порядков меньше времени безотказной работы каждого из составных элементов системы. Возможна еще одна реализация безопасной информационной системы управления на основе группы из двух элементов – с решающей обратной связью. Здесь выходные сигналы канала обработки информации поступают в инверсный преобразователь, осуществляющий преобразование выходных слов канала в соответствующие им входные слова, которые сравниваются с входными словами данного канала. Система переходит в состояние защитного отказа при любом первом несовпадении этих слов. Известно, что сложность инверсного преобразователя может быть выше, чем сложность второго канала. Это ограничивает использование такой архитектуры связанной с безопасностью двухканальной системы.

Оценим функциональную безопасность и надежность двухканальных систем. При этом ограничимся рассмотрением двух характерных вариантов организации системы: со встроенным контролем состояния надежности элементов (вариант 1) и с внешним функциональным контролем (вариант 2).

III.5.5.2. Модель функциональной безопасности двухканальной системы со встроенными средствами диагностики (вариант 1)

Граф состояний безопасности двухканальной системы со встроенными средствами диагностики (вариант 1) показан на рис. III. 5.7

Описание состояний:

0 – исправное состояние;

1 – отказ средств диагностики;

2 – защитный отказ системы вследствие обнаруженного штатными средствами диагностики с вероятностью ν отказа одного из каналов;

3 – необнаруженный отказ канала вследствие отказа или недостаточной эффективности средств диагностики (опасный отказ системы).

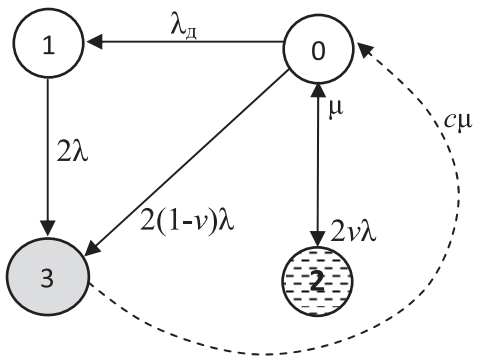


Рис. III.5.7. Граф состояний безопасности двухканальной системы

Предполагается, что потоки отказов и восстановлений, а также поток обнаружений отказов одного канала – простейшие с интенсивностями λ, μ, $\lambda_д$. Восстановление осуществляется в состоянии защитного отказа 2.

Ребра графа на рис. III. 5.7 помечены следующими параметрами $\lambda_д$ – интенсивность отказов средств диагностики; 2λ – интенсивность отказов двух работающих каналов; μ – интенсивность восстановления отказов одной ремонтной бригадой.

Из поглощающего состояния 3 опасного отказа показан условной (прерывистой) линией переход в начальное состояние 0. Это мнимое ребро (3-0) графа введено в качестве приема для определения коэффициента безопасности двухканальной системы Данное ребро помечено параметром $c\mu$ – интенсивность устранения опасного отказа системы, где коэффициент $0<c\leq 1$. Если для устранения опасного отказа не требуется дорабатывать систему, то $c = 1$ и интенсивность устранения опасного отказа равна интенсивности восстановления системы. Если требуется дорабатывать систему, то в зависимости от длительности доработки τ данный коэффициент будет иметь значение $c=1/\tau$, которое меньше 1.

Модель функциональной безопасности двухканальной системы на рис. III.5.7 предусматривает следующую логику функционирования системы. Начальное состояние 0 (все элементы системы исправны). В случае отказа средств диагностики происходит переход в состояние 1. Если же при исправных средствах диагностики отказал один любой канал (состояние 2) и отказ канала своевременно обнаружен с вероятностью ν, то система переводится в состояние защитного отказа (система не функционирует, канал ремонтируется). При скрытом отказе канала с вероятностью $\bar{\nu} = 1 - \nu$ или при отказе одного канала после отказа средств диагностики (путь 0 – 1 – 3) происходит переход в состояние 3 опасного отказа.

Решение данной модели будет состоять в аналитическом определении и исследовании основных показателей безопасности двухканальной системы: средней наработки до опасного отказа, интенсивности опасных отказов, вероятности опасного отказа, коэффициента безопасности. При указанных предпосылках поведение системы описывается с помощью Марковского случайного процесса. Для решения задачи предварительно определяют исходные параметры – вероятности переходов и математические ожидания безусловного времени пребывания устройства в перечисленных состояниях.

– *вероятности переходов:*

$$p_{01} = \frac{\lambda_д}{2\lambda + \lambda_д};\ p_{02} = \frac{2\nu\lambda}{2\lambda + \lambda_д};\ p_{03} = \frac{2(1-\nu)\lambda}{2\lambda + \lambda_д};$$

$$p_{20} = p_{13} = 1;\ p_{30} = 1 \ \text{(условно)};$$

– *математические ожидания времени* пребывания устройства в состояниях:

$$T_0 = \frac{1}{2\lambda + \lambda_д};\ T_1 = \frac{1}{2\lambda};\ T_2 = \frac{1}{\mu};\ T_3 = \frac{1}{c\mu} \ \text{(условно)}.$$

После этого определяют веса путей и контуров на графе:

– *веса путей*:

$$l_{01} = p_{01};\ l_{02} = p_{02};\ l_{03} = p_{01}p_{13} + p_{03} = p_{01} + p_{03};\ l_{13} = l_{20} = l_{30} = 1;$$

– *веса контуров* в подмножестве неопасных состояний $S_Н = \{0, 1, 2\}$:

$$C_0 = p_{02}p_{20} = p_{02}.$$

Средняя наработка до опасного отказа устройства определяется с помощью топологического метода [71] в множестве неопасных состояний $S_Н = \{0, 1, 2\}$ выражением:

$$T_{\text{ОП}} = \frac{T_0 + l_{01}T_1 + l_{02}T_2}{1 - C_0} = \frac{2\lambda(\mu + 2\nu\lambda) + \lambda_{\text{д}}\mu}{2\lambda\mu \cdot [2\lambda(1-\nu) + \lambda_{\text{д}}]}.$$

Если учесть, что $\lambda << \mu$; $(\lambda_{\text{Д}}/\lambda) = k$, то с погрешностью менее первого порядка малости справедливо приближенное выражение

$$T_{\text{ОП}} \cong \frac{2+k}{2\lambda \cdot (2\overline{\nu} + k)}, \text{ где } \overline{\nu} = 1 - \nu. \qquad (5.8)$$

Так как поток опасных отказов многократно разрежен относительно исходного потока неопасных отказов устройства, который является пуассоновским (простейшим), то, как отмечено в п. III.5.2, многократно разреженный случайным образом простейший поток отказов также является простейшим с постоянным параметром

$$\lambda_{\text{ОП}} = 1/T_{\text{ОП}} \cong \frac{2\lambda \cdot (2\overline{\nu} + k)}{2+k} \approx 2\lambda\nu + \lambda_{\text{д}} \text{ (при } \lambda >> \lambda_{\text{д}}).$$

Этот параметр является интенсивностью опасных отказов устройства.

В свою очередь, вероятность опасного отказа исследуемого устройства равна:

$$Q_{\text{ОП}}(t) = 1 - \exp(-\lambda_{\text{ОП}} t) \cong \frac{2\lambda \cdot (2\overline{\nu} + k)}{2+k} t$$

Коэффициент безопасности устройства находят на основании формулы (4.18) [1] графового метода в множестве неопасных состояний $S_{\text{Н}} = \{0, 1, 2\}$ следующим образом:

$$K_{\text{Б}} = P_0 + P_1 + P_2 = 1 - P_3 = 1 - \frac{\Delta G^3 T_3}{\Delta G^0 T_0 + \Delta G^1 T_1 + \Delta G^2 T_2 + \Delta G^3 T_3},$$

где P_i – финальные вероятности пребывания Марковского процесса в соответствующих вершинах графа; веса разложе-

ний графа без отмеченных вершин 0, 1, 2, 3 соответственно равны:

$$\Delta G^0 = \Delta G^2 = 1; \Delta G^1 = \Delta G^3 = 1 - C_0 = 1 - p_{02}p_{20} = 1 - p_{02}.$$

После подстановки исходных параметров формула коэффициента безопасности для системы, построенной по варианту 1, определяется как

$$K_{Б1} = 1 - \frac{2\lambda\left[2\lambda(1-\nu)+\lambda_д\right]}{2\lambda\left(\mu+2\lambda+\lambda_д\right)+(2\lambda+\mu)\left[2\lambda(1-\nu)+\lambda_д\right]}.$$

Поскольку $\mu >> 2\lambda+\lambda_Д$, то с погрешностью, не превышающей первого порядка малости, коэффициент безопасности рассматриваемой двухканальной системы определяется как

$$K_{Б1} = 1 - \frac{2\lambda\left[2\lambda(1-\nu)+\lambda_д\right]}{\mu\left[2\lambda(2-\nu)+\lambda_д\right]}, \qquad (5.9)$$

а коэффициент опасности

$$K_{ОП1} = 1 - K_{Б1} = \rho\frac{2(2\overline{\nu}+k)}{2(1+\overline{\nu})+k} = \rho\cdot F(\nu), \qquad (5.10)$$

где $\rho = \frac{\lambda}{\mu}; k = \frac{\lambda_д}{\lambda}$ $(0 < k < 1)$.

Из формулы (5.10) следует, что коэффициент опасности рассматриваемой двухканальной системы находится в прямой зависимости от интенсивности отказов канала, а также в обратной зависимости от скорости восстановления канала. Чем выше безотказность и готовность канала, тем выше функциональная безопасность системы. Отсюда следует целесообразность резервирования каналов системы. Окончательный ответ на этот вопрос возможен после анализа показателей надежности системы.

Повышение коэффициента функциональной безопасности системы (соответственно, снижение коэффициента опасности) имеет место при высокой эффективности обнаружения отказов каналов встроенными средствами диагностики, когда эти средства обнаруживают не менее (80 – 90)% отказов. Однако в условиях применения встроенных средств диагностики и контроля повышение эффективности обнаружения отказов влечет за собой увеличение объема средств диагностики и контроля, что, в свою очередь, оказывает негативное влияние на безопасность системы.

Оценим надежность системы по показателю средней наработки до отказа.

Для расчета и исследования показателей надежности исследуемой системы все множество состояний графа на рис. III.5.7. делят на подмножество работоспособных состояний $S_{\text{Р}} = \{0, 1\}$ и неработоспособных состояний $\overline{S}_{\text{Р}} = \{2, 3\}$, поскольку защитное состояние 2 также неработоспособно (в этом состоянии не выполняются системные функции).

В подграфе, содержащем состояния 0, 1, отсутствуют замкнутые контуры. Поэтому все веса разложений графа равны 1, а показатель средней наработки до отказа системы определяется в виде:

$$T_{\text{СР}} = T_0 + l_{01}T_1 = \frac{1}{2\lambda + \lambda_{\text{д}}} + \frac{\lambda_{\text{д}}}{2\lambda + \lambda_{\text{д}}}\frac{1}{2\lambda} = \frac{1}{2\lambda}.$$

Таким образом, организация двухканальной безопасной системы со встроенным контролем каналов (вариант 1) приводит к снижению ее надежности по показателю средней наработки до отказа в два раза по отношению к исходной одноканальной системе.

Оценим среднее время пребывания системы в состоянии 2 защитного отказа $T_{\text{ЗАЩ}}$, когда не выполняются системные функции, до наступления опасного отказа. Эта величина равна [69]

$$T_{\text{ЗАЩ}} = P_2 T_{\text{ОП}}. \tag{5.11}$$

Поскольку в информационных системах на практике всегда имеет место соотношение $\mu >> \lambda$, то с погрешностью менее первого порядка малости находим стационарную вероятность пребывания системы в состоянии 2: $P_2 \cong \dfrac{\lambda}{\mu \cdot (1+\nu)}$.

Подставляя последнее выражение и формулу (5.8) в формулу (5.11), находим

$$T_{\text{ЗАЩ}} \cong \frac{2+k}{2\mu \cdot \left[2+k(1+\nu)\right]}. \tag{5.12}$$

Полученное выражение среднего времени пребывания системы в состоянии защитного отказа показывает, что в целях обеспечения функциональной безопасности системы на выполнение защитных функций расходуется небольшое время, соизмеримое со средним временем восстановления системы. Также, вследствие того, что данная организация двухканальной системы не обеспечивает приемлемый уровень функциональной безопасности и ее надежность существенно ниже исходной одноканальной системы, такой вариант построения двухканальной безопасной системы не может быть рекомендован для широкого применения.

III.5.5.3. Модель функциональной безопасности двухканальной системы с внешним контролем (вариант 2)

Граф состояний безопасности двухканальной системы с внешним контролем, безопасным средством сравнения выходных результатов обработки информации каналами (безопасным компаратором) и электронным ключом (вариант 2) показан на рис. III. 5.8.

Описание состояний:

0 – исправное состояние;

1 – отказ компаратора и/или электронного ключа;

2 – отказ одного любого из двух каналов системы;

3 – защитный отказ системы;

4 – опасный отказ системы.

Предполагается, что потоки отказов и восстановлений – простейшие с интенсивностями λ, μ соответственно. Восстановление осуществляется в состоянии защитного отказа 3.

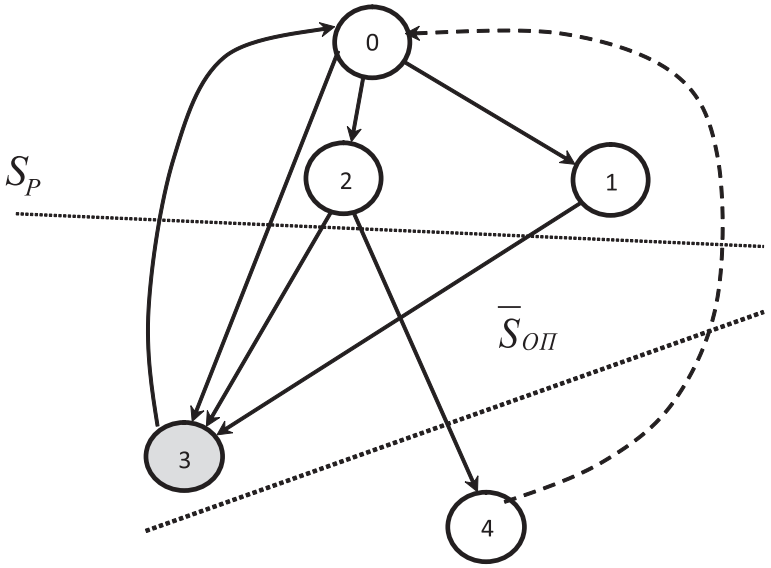


Рис. III.5.8. Граф состояний безопасности двухканальной системы с внешним контролем

Из поглощающего состояния 4 опасного отказа показан условной (прерывистой) линией переход в начальное состояние 0. Это мнимое ребро (5-0) графа введено в качестве приема для определения коэффициента безопасности двухканальной системы. Данное ребро помечено параметром $c\mu$ – интенсивность устранения опасного отказа системы, где коэффициент $0<c\leq 1$.

Если для устранения опасного отказа не требуется дорабатывать систему, то $c = 1$ и интенсивность устранения опасного отказа равна интенсивности восстановления объекта. Если требуется дорабатывать систему, то в зависимости от длительности доработки τ данный коэффициент будет иметь значение $c=1/\tau$, которое меньше 1.

Модель функциональной безопасности двухканальной системы с компаратором и внешним контролем на рис. III.5.8 предусматривает следующую логику функционирования системы. Начальное состояние 0 (все элементы объекта исправны). В случае отказа компаратора и/или ключа происходит переход в состояние 1. Если же при исправных средствах диагностики отказал один любой канал (состояние 2) и отказ канала своевременно обнаружен, то система переводится в состояние защитного отказа (система не функционирует, канал ремонтируется). Если при условии отказавшего одного канала отказ второго канала происходит до очередного внешнего контроля, который проводится через интервалы времени $T_к$, то система переходит в состояние 4 опасного отказа.

Решение данной модели будет состоять в аналитическом определении и исследовании основных показателей безопасности двухканальной системы: средней наработки до опасного отказа, интенсивности опасных отказов, вероятности опасного отказа, коэффициента безопасности.

Принятые предпосылки. При обнаружении отказов двух каналов, компаратора или ключа, а также в случае несовпадения результатов работы каналов система переводится в защитное состояние. При необнаруженном отказе одного канала система продолжает выполнять системные функции. При указанных предпосылках поведение системы описывается с помощью Марковского случайного процесса. Для решения задачи предварительно определяют исходные параметры – вероятности

переходов и математические ожидания безусловного времени пребывания системы в перечисленных состояниях.

Вероятности переходов. Определим вероятность перехода из состояния 0 в состояние 1. Эта вероятность должна оценивать следующие два несовместных события: во-первых, отказ вспомогательного средства (компаратора или ключа) произошел раньше, чем отказ любого одного из двух каналов; во-вторых, отказ этого средства еще не обнаружен внешним контролем. Таким образом,

$$p_{01} = p_1 p_2, \text{ где } p_1 = \int_0^\infty e^{-2\lambda \cdot t} \lambda_{КК} e^{-\lambda_{КК} t} dt = \frac{\lambda_{КК}}{2\lambda + \lambda_{КК}},$$

$$p_2 = 1 - \gamma = \bar{\gamma} \text{ и } \lambda_{КК} = \lambda_{\text{КОМПАРАТОРА}} + \lambda_{\text{КЛЮЧА}}.$$

Здесь γ – вероятность обнаружения отказа путем периодического контроля с периодом $T_к$, $\lambda_{КК}$ – интенсивность отказов вспомогательного оборудования (ключа и компаратора). Периодический контроль системы осуществляется в течение всего времени его исправной работы, т. е. до очередного отказа. Средняя наработка до отказа системы обозначается как $T_{СР}$. В течение интервала времени $T_к$ отказы не обнаруживаются. До этого времени отказы обнаруживаются. Предполагается, что внешний контроль полный и достоверный (вероятность правильного обнаружения практически равна 1). Длительность каждого внешнего контроля настолько мала, что может не приниматься в расчет.

$$\text{Тогда } \gamma = \frac{T_{СР} - T_к}{T_{СР}} = 1 - \frac{T_к}{T_{СР}} = 1 - \bar{\gamma} \text{ и } p_{01} = \frac{\frac{T_к}{T_{СР}} \lambda_{КК}}{2\lambda + \lambda_{КК}}.$$

Вероятность перехода из состояния 0 в состояние 3 (в состояние защитного отказа) относительно отказов вспомогательных средств оценивается также как и предыдущая вероятность p_{01}

с помощью произведения вероятностей двух несовместных событий, с той разницей, что второе событие является противоположным тому событию, которое рассматривалось в предыдущем случае. Речь идет о том, что отказ вспомогательного средства уже обнаружен внешним контролем, что потребовало перевода системы в состояние защитного отказа.

Следовательно,

$$p_{03}^{\text{ВСП}} = p_1 \overline{p}_2 = \frac{\lambda_{\text{КК}}(1 - \frac{T_{\text{к}}}{T_{\text{СР}}})}{2\lambda + \lambda_{\text{КК}}}.$$

Вероятность перехода из состояния 0 в состояние 2 должна оценивать следующие два несовместных события: во-первых, раньше отказ любого одного из двух каналов, чем отказ вспомогательного средства; во-вторых, отказ этого канала еще не обнаружен внешним контролем. Вероятность перехода из состояния 0 в состояние 3 (в состояние защитного отказа) относительно отказов любого одного канала оценивается так же, как и предыдущая вероятность p_{02} с помощью произведения вероятностей двух несовместных событий, с той разницей, что второе событие является противоположным тому событию, которое рассматривалось в предыдущем случае. Речь идет о том, что отказ канала уже обнаружен внешним контролем, что потребовало перевода системы в состояние защитного отказа.

Таким образом,

$$p_{02} = \frac{2\lambda \frac{T_{\text{к}}}{T_{\text{СР}}}}{2\lambda + \lambda_{\text{КК}}} \text{ и } p_{03}^{\text{КАН}} = \frac{2\lambda \cdot \left(1 - \frac{T_{\text{к}}}{T_{\text{СР}}}\right)}{2\lambda + \lambda_{\text{КК}}};$$

$$p_{03} = p_{03}^{\text{ВСП}} + p_{03}^{\text{КАН}} = (1 - \frac{T_{\text{к}}}{T_{\text{СР}}}) \frac{\lambda_{\text{КК}} + 2\lambda}{2\lambda + \lambda_{\text{КК}}} = 1 - \frac{T_{\text{к}}}{T_{\text{СР}}}.$$

Проверка. Все вероятности выхода из состояния 0 образуют полную группу событий. Их сумма должна равняться 1. Убедимся в этом.

$$p_{01}+p_{02}+p_{03}=\frac{\lambda_{КК}\dfrac{T_к}{T_{СР}}}{2\lambda+\lambda_{КК}}+\frac{2\lambda\dfrac{T_к}{T_{СР}}}{2\lambda+\lambda_{КК}}+\left(1-\frac{T_к}{T_{СР}}\right)=1.$$

В свою очередь, $p_{13}=p_{30}=1$.

Определим вероятности переходов из состояния 2 системы. В этом состоянии исправны компаратор, ключ и один любой из двух каналов. Переход в состояние опасного отказа произойдет в случае отказа второго канала, а переход в состояние защитного отказа при отказе компаратора или ключа. Вероятности этих событий соответственно равны

$$p_{24}=\int_0^\infty e^{-\lambda_{КК}t}\lambda e^{-\lambda\cdot t}dt=\frac{\lambda}{\lambda+\lambda_{КК}};\ p_{23}=\frac{\lambda_{КК}}{\lambda+\lambda_{КК}}.$$

Математические ожидания времени пребывания системы в состояниях графа рис. III.5.9 находятся по следующим формулам

$$T_0=\frac{1}{2\lambda+\lambda_{КК}};T_1=\frac{1}{2\lambda};T_2=\frac{1}{\lambda+\lambda_{КК}};T_3=\frac{1}{\mu};\ T_4=\frac{1}{c\mu}.$$

После этого определяют веса путей и контуров на графе:

– веса путей:

$$l_{01}=p_{01};l_{02}=p_{02};l_{03}=p_{01}p_{13}+p_{03}+p_{02}p_{23}=p_{01}+p_{03}+p_{02}p_{23};$$
$$l_{23}=p_{23};l_{24}=p_{24};l_{13}=l_{30}=l_{40}=1;$$

– веса контуров в подмножестве неопасных состояний $S_Н=\{0,1,2,3\}$: $C_1=p_{01}p_{13}p_{30}=p_{01}$; $C_2=p_{02}p_{23}p_{30}=p_{02}p_{23}$; $C_3=p_{03}p_{30}=p_{03}$, так как $p_{30}=1$.

Средняя наработка до отказа системы определяется на основании метода [71] в множестве состояний $S_{\text{Н}} = \{0, 1, 2\}$ по формуле:

$$T_{\text{СР}} = T_0 + p_{01}T_1 + p_{02}T_2 = \frac{1}{2\lambda + \lambda_{\text{КК}}} +$$
$$+ \frac{T_{\text{к}}}{T_{\text{СР}}} \frac{\lambda_{\text{КК}}}{(2\lambda + \lambda_{\text{КК}})} \frac{1}{2\lambda} + \frac{T_{\text{к}}}{T_{\text{СР}}} \frac{2\lambda}{(2\lambda + \lambda_{\text{КК}})} \frac{1}{(\lambda + \lambda_{\text{КК}})}.$$

Запишем соотношение между интенсивностью отказов канала обработки информации и вспомогательного оборудования в виде отношения $k=(\lambda_{kk}/\lambda)$ или $\lambda_{kk}=k\lambda$.

Здесь очевидно, что коэффициент k отношения интенсивности отказов вспомогательного оборудования и канала обработки информации не должен быть больше 1 вследствие того, что вспомогательное оборудование выполняет значительно меньший объем функций, чем канал обработки информации. Следовательно, этот коэффициент находится в пределах $0<k\leq 1$. С учетом этого отношения выражение показателя средней наработки до отказа объекта преобразуется к виду:

$$T_{\text{СР}} = \frac{2(1+k)T_{\text{СР}} + T_{\text{к}}(4+k+k^2)}{2\lambda(2+k)(1+k)T_{\text{СР}}}.$$

Неизвестный параметр $T_{\text{СР}}$ находится и в левой и в правой части данного выражения. Здесь имеет место квадратичное уравнение вида

$$AT_{\text{СР}}^2 - BT_{\text{СР}} - C = 0,$$

где $A=2\lambda(2+k)(1+k)$; $B=-2(1+k)$; $C=-T_{\text{к}}(4+k+k^2)$.

Поскольку дискриминант $D=B^2-4AC$ положителен, то данное квадратичное уравнение имеет следующее решение:

$$T_{\text{СР}} = \frac{(1+k) \pm \sqrt{(1+k)^2 + 2\lambda T_{\text{к}}(2+3k+k^2)(4+k+k^2)}}{2\lambda(2+k)(1+k)}. \quad (5.13)$$

В подкоренном выражении формулы (5.13) имеют место две несоизмеримые по величине составляющие, а именно $(1+k)^2 >> 2\lambda T_{\text{к}}(2+3k+k^2)(4+k+k^2)$. Это объясняется тем, что в реальном диапазоне значений частоты контролей и интенсивностей отказов каналов их произведение на несколько порядков меньше 1, т.е. $\lambda T_{\text{к}} < 10^{-3}$, и только при периоде внешнего контроля, исчисляемого тысячами часов указанные выше величины соизмеримы. Однако такие периоды контроля не имеют отношения к информационным системам.

Таким образом, с высокой точностью находим выражение средней наработки до отказа исследуемой системы

$$T_{\text{СР}} = \frac{1}{\lambda(2+k)}. \quad (5.14)$$

Анализ формулы (5.14) свидетельствует, во-первых, о том, что интервалы между контролями объекта в реальном диапазоне их значений практически не влияют на безотказность данной двухканальной системы и, во-вторых, как и следовало ожидать, безотказность двухканальной системы по показателю средней наработки до отказа более, чем в два раза хуже безотказности исходной одноканальной системы.

Средняя наработка до опасного отказа системы определяется в множестве неопасных состояний $S_{\text{Н}} = \{0, 1, 2\}$ следующей формулой:

$$T_{\text{ОП}} = \frac{T_0 + p_{01}T_1 + p_{02}T_2 + (p_{01}p_{13} + p_{02}p_{23} + p_{03})T_3}{1 - p_{30}(p_{01}p_{13} + p_{02}p_{23} + p_{03})}. \quad (5.15)$$

Подставляя формулу (5.14) в формулу (5.15) и учитывая, что $T_0+p_{01}T_1+p_{02}T_2=T_{СР}$; $1+k>>\lambda T_к$; $\mu>>\lambda$, находим с погрешностью, не превышающей первого порядка малости, что

$$T_{ОП}=\frac{(1+k)}{2\lambda^2 T_к(2+k)}. \qquad (5.16)$$

Из формулы (5.16) следует, что в двухканальных системах, построенных по варианту 2, эффективное повышение функциональной безопасности достигается путем сокращения интервалов времени между внешними контролями. При этом интенсивность восстановления отказов практически не влияет на управление безопасностью системы.

Обращает на себя внимание то, что средняя наработка до опасного отказа обратно пропорциональна квадрату интенсивности отказов канала обработки информации. Это означает, что даже относительно небольшое уменьшение интенсивности отказа канала может привести к существенному повышению безопасности системы.

Интервалы между внешними контролями в зависимости от требований к уровню полноты безопасности системы не должны превышать допустимое время существования скрытого отказа, определяемого по формуле (5.3), т.е.

$$T_к \le \tau_{ДОП}=\frac{0,2\cdot Q_Б(1)}{a_1^2}=\frac{Q_Б(1)}{2\lambda^2}.$$

Отношение средней наработки до опасного отказа к средней наработке до отказа системы определяется следующим выражением:

$$\frac{T_{ОП}}{T_{СР}}=\frac{(1+k)}{2\lambda T_к}.$$

Это отношение показывает, что при данном двухканальном варианте организации обеспечения функциональной безопас-

ности системы время до возникновения ее опасного отказа на несколько порядков больше, чем время до возникновения отказа канала обработки информации. Этот результат свидетельствует о высокой эффективности данного варианта обеспечения безопасности.

В диапазоне реальных соотношений между интенсивностью отказа канала обработки информации и интенсивностью отказа вспомогательного оборудования ($k \le 1$) формула (5.16) может быть с достаточной для практики точностью преобразована к следующему виду:

$$T_{\text{ОП}} = \frac{1}{3\lambda^2 T_{\text{к}}}. \qquad (5.17)$$

Коэффициент безопасности системы определяется с помощью метода [71] в множестве неопасных состояний $S_{\text{Н}} = \{0, 1, 2, 3\}$ следующим образом:

$$K_{\text{Б}} = P_0 + P_1 + P_2 + P_3 = 1 - P_4 =$$
$$= 1 - \frac{\Delta G^4 T_4}{\Delta G^0 T_0 + \Delta G^1 T_1 + \Delta G^2 T_2 + \Delta G^3 T_3 + \Delta G^4 T_4},$$

где P_i –финальные вероятности пребывания Марковского процесса в соответствующих вершинах графа; веса разложений графа без отмеченных вершин 0, 1, 2, 3, 4 соответственно равны:

$$\Delta G^0 = 1;\ \Delta G^1 = 1 - C_2 - C_3 = 1 - p_{02}p_{23} - p_{03};\ \Delta G^2 = 1 - C_1 - C_3 = 1 - p_{01} - p_{03};$$
$$\Delta G^3 = 1;\ \Delta G^4 = 1 - C_1 - C_2 - C_3 = 1 - p_{01} - p_{02}p_{23} - p_{03}.$$

После подстановки исходных параметров с учетом того, что $1+k >> \lambda T_{\text{к}}$; $\mu >> \lambda$, формула коэффициента безопасности для системы, построенной по варианту 2, с погрешностью не выше первого порядка малости имеет следующий вид:

$$K_{Б2} = 1 - K_{ОП2} = 1 - \frac{2\lambda^2 T_к (2+k)}{\mu(1+k)}. \qquad (5.18)$$

С учетом формулы (5.12) коэффициент опасности двухканальной системы, построенной по варианту 2, может быть записан как

$$K_{ОП2} = \frac{2\lambda T_к}{T_{СР}\mu(1+k)} = \frac{2\bar{\gamma}\lambda}{\mu(1+k)} = \frac{2\bar{\gamma}\rho}{1+k} = \rho \cdot F(\gamma).$$

Сравнение коэффициентов безопасности двух типовых вариантов построения функционально безопасных двухканальных систем представим в виде отношения $R = \frac{K_{ОП1}}{K_{ОП2}} = \frac{F(\nu)}{F(\gamma)}$. Графики этого отношения в зависимости от эффективности обнаружения отказов показаны на рис. III.5.9.

Из графиков рис. III.5.9. следует, что при относительно низкой вероятности правильного обнаружения отказов (до уровня 0,7 – 0,8) оба варианта построения системы не имеют заметных преимуществ и не обеспечивают приемлемый коэффициент безопасности. Однако, уже начиная с уровня вероятности 0,8 – 0,9 становится очевидным большое преимущество варианта 2 перед вариантом 1 – коэффициент опасности в несколько раз ниже. По мере роста вероятности правильного обнаружения это преимущество возрастает по экспоненциальному закону.

К этому следует добавить, что нереально добиться высокого уровня вероятности правильного обнаружения отказов с помощью встроенных средств контроля и диагностики, потому что в этом случае практически невозможно обеспечить необходимую полноту и достоверность контроля. При применении внешнего контроля состояния надежности объекта по выходному эффекту можно добиться правильной оценки с заданным уровнем гарантии.

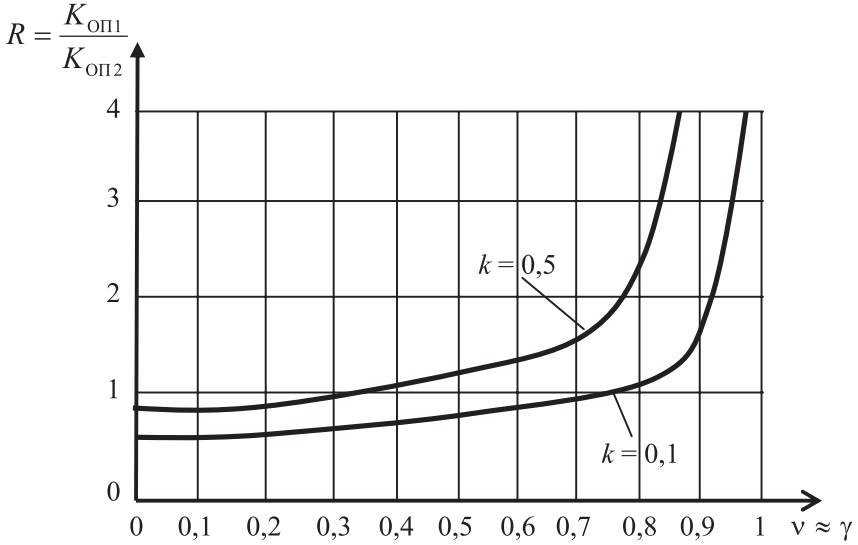


Рис. III.5.9. Графики отношений коэффициента опасности двухканальной системы, построенной по варианту 1, к коэффициенту опасности системы, построенной по варианту 2, в зависимости от эффективности обнаружения отказов

Учитывая возможность сокращения интервалов времени между контролями, в данной организации системы представляется возможным обеспечить указанный выше требуемый уровень вероятности правильного обнаружения отказов.

Оценим среднее суммарное время пребывания системы в состоянии защитного отказа $T_{ЗАЩ2}$ в течение времени до наступления опасного отказа. Эта величина равна [70]

$$T_{ЗАЩ2}=P_З T_{ОП}. \qquad (5.19)$$

Поскольку в информационных системах на практике всегда имеет место соотношение $\mu >> \lambda$, то с погрешностью менее первого порядка малости находим стационарную вероятность пребывания системы в состоянии 3: $P_3 \cong \frac{\lambda(2+k)}{\mu}$. Подставляя последнее выражение и формулу (5.16) в формулу (5.19), находим

$$T_{\text{ЗАЩ 2}} \cong \frac{1+k}{2\lambda\mu \cdot T_K}. \qquad (5.20)$$

Полученное выражение среднего времени пребывания системы в состоянии защитного отказа показывает, что в целях обеспечения функциональной безопасности существенная часть времени работы системы непроизводительно расходуется на выполнение защитных функций. Это время соизмеримо со средним временем исправной работы одного канала обработки информации. Оно может быть уменьшено путем повышения интенсивности восстановления системы.

В целом сравнение двух типовых вариантов организации двухканальной функционально безопасной системы позволяет сделать следующие выводы:

– вариант 1 предпочтительнее с точки зрения непроизводительных потерь в работе системы, вызванных переходами в защитное состояние;

– вариант 2 имеет значительные преимущества по обеспечению функциональной безопасности. Непроизводительные потери в работе системы вследствие переходов в защитные состояния есть не что иное, как естественный результат эффективного обнаружения отказов составных средств системы. С этим приходится мириться при необходимости обеспечения высоких уровней функциональной безопасности системы.

III.5.6. Оценка вероятности возникновения опасных отказов при перезапуске двухканальных систем

III.5.6.1. Проблема перезапуска каналов

С помощью формирования двух, трех и более независимых однотипных каналов во многом решается проблема функцио-

нальной безопасности информационных управляющих систем с заданным уровнем полноты безопасности. В случаях несовпадения выходных результатов работы каналов система переводится в защитное состояние, устраняется неисправность канала и продолжается работа системы. С одной стороны, чем выше надежность канала, тем реже случаи несовпадения результатов, тем реже система находится в защитном состоянии и тем выше ее готовность к выполнению системных функций, а следовательно, тем больше ее производительность. С другой стороны, чем чаще не совпадают выходные результаты каналов, тем больше простаивает система в защитных состояниях и тем ниже эффективность информационной системы управления совместно с объектом управления. Вывод очевиден: нужно повышать надежность каналов. Задача не вызывает трудностей, имеется богатый арсенал методов и практических способов ее решения, многие из которых рассмотрены в предыдущих разделах книги. Но в информационных системах имеет место определенный вид неисправностей, которые не удается нейтрализовать известными методами. Даже применение в этих целях активной защиты затруднено, потому что связано с большими дополнительными затратами ресурсов. Речь идет о сбоях функционального характера [2, 36, 37, 38].

Практика испытаний и эксплуатации информационных систем различного назначения показала, что наиболее опасными угрозами функциональной надежности являются сбои информационной техники, вызванные внутренними или внешними дестабилизирующими факторами – помехами. Именно помехи, главным образом по цепи питания, по заземлению и по входу, в сочетании со входными сигналами, передаточными и амплитудно-временными характеристиками интегральных схем являются в совокупности теми факторами, приводящими к сбоям, которые, в свою очередь, влияют на правильность выполнения

логических функций, микроопераций, операций, процессов и информационных технологий в целом [2, 77, 78]. Эта, на первый взгляд, длинная цепочка условий возникновения ошибок в выполнении информационных технологий кажется маловероятной. Однако весь вопрос в том, что понимается под сбоем функционального характера и как часто возникают такие события.

Сбой функционального характера – событие, заключающееся в однократном искажении перерабатываемой или хранящейся в информационной технике информации, возникающее под воздействием внутренних или внешних дестабилизирующих факторов (помех).

При испытаниях автоматизированных систем управления было установлено [2], что интенсивность сбоев функционального характера средств информационной техники, работающей в реальном масштабе времени, более чем на два порядка выше интенсивности отказов [77]. По мере перехода к интегральным схемам, по мере создания больших и сверхбольших интегральных схем, значительно выросло быстродействие информационной техники, что повлекло за собой повышение интенсивности их сбоев. Это связано со снижением их помехоустойчивости. У современных информационных систем управления интенсивность сбоев функционального характера примерно на четыре порядка выше интенсивности отказов аппаратных средств.

По характеру влияния на результаты выполнения информационных технологий сбои подразделяются на две группы: защищенные сбои и сбойные ошибки. Защищенные сбои – часть сбоев информационной техники, которая своевременно (в пределах допустимого времени перерыва в работе) обнаружена системой контроля и автоматически устранена с помощью предусмотренной в системе избыточности.

Сбойные ошибки – это результаты возмущающих воздействий незащищенных сбоев на процессы выполнения программ,

информационных технологий и на информационный процесс в целом. Эти воздействия проявляются в виде искажений, подмены или потерь данных, ошибок в промежуточных и/или в выходных результатах. Основное деструктивное влияние на результаты информационного процесса оказывают сбои функционального характера. Они приводят к ошибкам в выполнении микроопераций, а те, в свою очередь, к ошибкам в выполнении операций, которые влияют на микропроцессы, затем и на процесс. Поток сбоев многократно разрежается. Остается поток сбойных ошибок. И даже у этого потока событий интенсивность сбойных ошибок значительно превышает интенсивность отказов цифровых устройств системы управления. Также как это имеет место в части сбоев функционального характера, остается проблематичной нейтрализация сбойных ошибок.

Именно вследствие этих неисправностей функционального характера интенсивность защитных отказов в системе управления во многих случаях неприемлема. Это особенно характерно для микропроцессорных систем. Такие системы обеспечивают безопасность управления объектами, но при частом несовпадении выходных результатов каналов не могут выполнять системные функции. Естественный выход из сложившейся ситуации – перезапуск каналов. Перезапуск каналов подразумевает следующие действия в системе. В случае несовпадения выходных результатов повторяется обработка информации в каналах (с начала выполнения задачи или с последней контрольной точки). Выходные результаты сравниваются. В случае их совпадения система выполняет предусмотренные системные функции. В противном случае снова повторяется обработка результатов. Количество повторений предусматривается при проектировании функциональной безопасности системы на основе результатов моделирования надежности системы.

III.5.6.2. Постановка задачи

Предполагается, что в двухканальной системе каналы идентичны и независимы, устройство сравнения безопасно (всегда выдает правильное решение о совпадении или несовпадении выходных результатов). Если возникли отказы или сбои в двух или более неэквивалентных элементах обоих каналов и выходные результаты работы каналов не совпадают, то имеют место неэквивалентные отказы. В этих случаях, несмотря на ошибки в выходных результатах работы обоих каналов, опасные отказы отсутствуют, поскольку работа системы прекращается и формируется сигнал защитного отказа. При сравнительно частых сбоях микропроцессорной техники подобные события возникают с недопустимой вероятностью и приводят к прекращению выполнения системных функций, например, прекращению движения поездов на железнодорожном транспорте. Отсюда возникает необходимость в перезапуске каналов для продолжения их работы, не дожидаясь выезда ремонтной бригады на объект для устранения отказа.

Если причиной защитного отказа были сбои, то при перезапуске они устраняются. Если же причиной защитного отказа были отказы неэквивалентных (разных) элементов обоих каналов, то после перезапуска одиночные отказы в каналах остаются, и возникает опасность возникновения новых отказов, при которых выходные результаты работы каналов совпадают. Такие отказы принято называть эквивалентными. В этой ситуации отказ в системе не обнаруживается, он становится опасным и может вызвать аварию, например, поезда или привести к значительному ущербу. Следует отметить, что эквивалентный отказ может быть следствием различных событий. Например, при наличии одиночных отказов в обоих каналах после перезапуска возник второй отказ в одном из каналов, который оказался эквивалентным уже существующему отказу другого канала.

Или, если все три отказа неэквивалентны, то после очередного перезапуска может возникнуть четвертый отказ (второй отказ в каждом из каналов или третий отказ в одном канале при одном отказе в другом канале) который может привести к эквивалентному (опасному) отказу в системе, и т.д.

Задача заключается в оценке вероятности опасного отказа двухканальной системы вследствие ее перезапуска.

Принятые предпосылки:

– потоки отказов в обоих каналах информационных микропроцессорных систем являются простейшими с одинаковыми интенсивностями λ;

– каждый канал содержит m составных элементов. Все элементы нумеруются в пределах значений от 1 до m. Предполагается, что все элементы одинаковы по объему оборудования и, следовательно, по интенсивности отказов. Это означает, что интенсивность отказов одного элемента равна λ/m;

– накопленное количество отказов элементов в одном канале в результате перезапусков не может быть больше $m - 1$;

– эквивалентный (опасный) отказ системы имеет место в том случае, если отказывают одинаковые элементы в обоих каналах.

III.5.6.3. Модель для оценки вероятности возникновения опасных отказов

При простейшем потоке отказов вероятность возникновения двух и более отказов элементов в одном канале в течение времени t определяется следующим выражением

$$Q(t, n \geq 2) = \sum_{n=2}^{m} \frac{(\lambda_n t)^n}{n!} \exp(-\lambda_n t), \qquad (5.21)$$

где $\lambda_n = \dfrac{(m-n)\lambda}{m}$, $m \geq n$.

Данное выражение является модификацией формулы Пуассона для условия дискретно уменьшающейся интенсивности отказов канала по мере отказов его составных элементов. Справедливость данного выражения следует из теоремы Григелиониса [49], согласно которой в результате случайных разрежений простейшего потока образуется опять-таки простейший поток с меньшим значением параметра потока.

Относительно двух каналов необходимо учитывать различные сочетания количества отказавших элементов в каналах после перезапусков, например, один элемент в одном канале и два или более отказавших элементов в другом канале, равное количество отказавших элементов и т.д. Все возможные ситуации показаны на графе рис. III.5.10.

На графе рис. III.5.10 вершинами показаны состояния отказов элементов каналов. Вершины пронумерованы цифрами 0, 1, 2, …, *M*. Количество вершин конечное, поскольку в каждом канале системы не более m элементов. Ряд вершин пронумеровано одинаковыми цифрами (1 и 1; 2 и 2; 4 и 4; 5 и 5 и др.). Такие вершины называются зеркальными, поскольку они зеркально отражают состояния отказов элементов в разных каналах. Например, вершина 2 соответствует отказу одного элемента в одном канале и трех элементов в другом канале. Вторая вершина 2 соответствует отказам трех элементов в одном канале и одного элемента в другом канале. Для пояснения этих состояний в вершинах графа приведены цифры. Первая цифра отражает количество отказов элементов одного канала, вторая цифра – другого канала. Например, в исходном состоянии 0 исследуемой системы до первого перезапуска отказали по одному элементу каждого канала (11).

Определим вероятности переходов из начального (исходного) состояния во все другие состояния графа. С этой целью найдем

функцию распределения времени перехода из состояния 0 в состояние 1 (основное или зеркальное). Она определяется с помощью формулы (5.21) при $n = 1$

$$Q_{01}(t) = Q(t, n = 3) = \lambda_3 t \exp(-\lambda_3 t) = \frac{(m-3)\lambda t}{m} \exp\left(-\frac{(m-3)\lambda t}{m}\right).$$

Функция плотности условного времени пребывания системы в состоянии 0 до перехода в основное (или зеркальное) состояние 1 равна

$$f_{01}(t) = \left(1 - Q_{01}(t)\right) \frac{dQ_{01}(t)}{dt} = \frac{(m-3)\lambda}{m} \left[1 - \frac{(m-3)\lambda t}{m}\right] \times$$
$$\times \left[1 - \frac{(m-3)t}{m-1} \exp\left(-\frac{(m-3)\lambda t}{m}\right)\right] \cdot \exp\left(-\frac{(m-3)\lambda t}{m}\right),$$

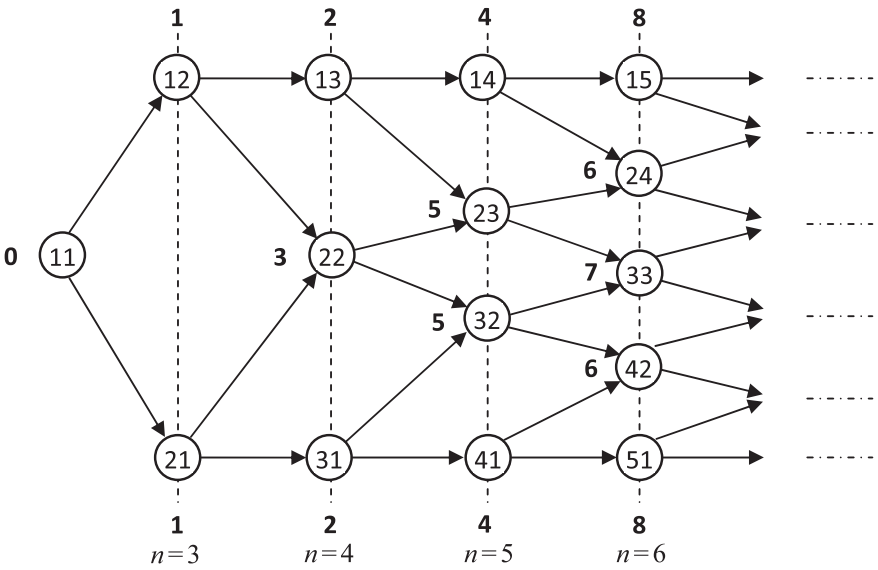


Рис. III.5.10. Диаграмма сочетания количества отказавших элементов в каналах после их перезапусков

а вероятность перехода:

$$p_{01} = \int_0^\infty f_{01}(t)dt = 1/2 \text{ и сумма } p_{01} = p_{01}^{\text{зер}} = 1.$$

Тривиальные выкладки вероятностей переходов по методике, приведенной для определения $p_{01} = \int_0^\infty f_{01}(t)dt$, показали, что все вероятности переходов между смежными вершинами равны 1/2. Это естественно для экспоненциальных распределений в двухканальной системе с двумя выходами из каждого предыдущего состояния. Однако веса путей из начальной вершины 0 в вершину *i* имеют различные значения. Так, вес пути из вершины 0 в вершину 2 равен $l_{02}=p_{01}p_{12}=1/4$, а вес пути из вершины 0 в вершину 3 $l_{03}=p_{01}p_{13}=1/2$. В свою очередь, суммарный вес путей из вершины 0 в вершину 5 равен $l_{05} = 2(l_{02}p_{25} + l_{03}p_{35}) = 2(\frac{1}{4}\frac{1}{2} + \frac{1}{2}\frac{1}{2}) = \frac{3}{4}$, а вес пути в вершину 4 равен $l_{04} = 2l_{02}p_{24} = \frac{2}{4}\frac{1}{2} = \frac{1}{4}$.

Для определения общего выражения суммарного веса путей из начальной вершины 0 в вершину *i* слоя *n*, под которым понимается количество накопленных в обоих каналах отказов, необходимо, во-первых, определить общее выражение веса каждого пути, во-вторых, общее выражение количества путей, ведущих из начальной вершины в вершину *i*. Из графа на рис. III.5.10 и из приведенных выше аргументов следует, что вес одного пути равен $l_0^1(i, n-i) = l_{0n}^1 = \frac{1}{2^n}$. Количество путей, ведущих из начальной вершины 0 в вершину *i* слоя *n*, как следует из рис. III.5.10, можно подсчитать с помощью треугольника Паскаля:

n = 3 1;
n = 4 1 1;
n = 5 1 2 1;

$n = 6\ 1\ 3\ 3\ 1;$
$n = 7\ 1\ 4\ 6\ 4\ 1;$
$n = 8\ 1\ 5\ 10\ 10\ 5\ 1;$
$n = 9\ 1\ 6\ 15\ 20\ 15\ 6\ 1.$

Таким образом, общее выражение количества путей, ведущих из начальной вершины 0 в вершину i слоя n, имеет следующий вид:

$$M_0(i, n-i) = C_{n-2}^{n-1-i}.$$

Тогда суммарный вес путей, ведущих из начальной вершины 0 в вершину i слоя n, вычисляется следующим образом:

$$l_0(i, n-i) = M_0(i, n-i) l_0^1(i, n-i) = C_{n-2}^{n-1-i} \frac{1}{2^n}. \qquad (5.22)$$

Заметим, что по формуле полной вероятности сумма весов путей из начальной вершины во все вершины i, $n-i$ любого слоя n (см. рис. III.5.10) всегда равна 1, т.е.

$$\sum_{i=1}^{n-1} l_0(i, n-i) = 1.$$

Для решения поставленной задачи требуется найти функции распределения времени пребывания системы в состояниях 1, 2, …, i, …, M. Эти функции находятся с помощью графа рис. III.5.10 по следующей схеме: для любой вершины с цифрами i, $n–i$, приведенными в вершинах графа, находятся по формуле (5.22) и перемножаются как независимые события вероятности отказов элементов каналов

$$Q(t, i, n-i) = Q(t, i) \cdot Q(t, n-i).$$

Принятая предпосылка о независимости отказов элементов канала основывается на экспериментальных данных о простейшем потоке отказов цифровых устройств.

Руководствуясь приведенной схемой, находим

$$Q(t,12) = Q(t,21) = \lambda_1 t e^{-\lambda_1 t} \frac{(\lambda_2 t)^2}{2} e^{-\lambda_2 t} = \frac{\lambda_1 \lambda_2^2 t^3}{2} e^{-(\lambda_1+\lambda_2)t};$$

где $\lambda_2 = \frac{m-2}{m}\lambda;$

$$Q(t,13) = Q(t,31) = \lambda_1 t e^{-\lambda_1 t} \frac{(\lambda_3 t)^3}{3!} e^{-\lambda_3 t} = \frac{\lambda_1 \lambda_3^3 t^4}{6} e^{-(\lambda_1+\lambda_3)t};$$

где $\lambda_3 = \frac{m-3}{m}\lambda;$

$$Q(t,22) = \left[\frac{(\lambda_2 t)^2}{2!} e^{-\lambda_2 t}\right]^2 = \frac{(\lambda_2 t)^4}{4} e^{-2\lambda_2 t};$$

$$Q(t,23) = Q(t,32) = \frac{\lambda_2^2 \lambda_3^3 t^5}{12} e^{-(\lambda_2+\lambda_3)t};$$

$$Q(t,14) = Q(t,41) = \lambda_1 t e^{-\lambda_1 t} \frac{(\lambda_4 t)^4}{4!} e^{-\lambda_4 t} = \frac{\lambda_1 \lambda_4^4 t^5}{24} e^{-(\lambda_1+\lambda_4)t};$$

где $\lambda_4 = \frac{m-4}{m}\lambda;$

$$Q(t,15) = Q(t,51) = \lambda_1 t e^{-\lambda_1 t} \frac{(\lambda_5 t)^5}{5!} e^{-\lambda_4 t} = \frac{\lambda_1 \lambda_5^5 t^6}{120} e^{-(\lambda_1+\lambda_5)t};$$

где $\lambda_5 = \frac{m-5}{m}\lambda;$

$$Q(t,24) = Q(t,42) = \frac{\lambda_2^2 \lambda_4^4 t^6}{120} e^{-(\lambda_2+\lambda_4)t};$$
$$Q(t,33) = \left[\frac{(\lambda_3 t)^2}{3!} e^{-\lambda_3 t}\right]^2 = \frac{(\lambda_3 t)^6}{36} e^{-2\lambda_3 t},$$

и т.д.

В общем случае справедливо выражение

$$Q(t,i,n-i) = \frac{\lambda_i^i \lambda_{n-i}^{n-i} t^n}{(n-i)!\,i!} \exp\left[-(\lambda_i + \lambda_{n-i})t\right]. \qquad (5.23)$$

Теперь есть исходные условия для определения вероятности того, что в результате перезапусков в обоих каналах в течение времени t в сумме отказали n составных элементов из общего числа $2m$

$$Q(t,n) = \sum_{i=1}^{n-1} l_0(i,n-i) Q(t,i,n-i).$$

Подставив в последнее выражение формулы (5.22) и (5.23), получим

$$Q(t,n) = \sum_{i=1}^{n-1} \frac{C_{n-2}^{n-1-i}}{2^n} \frac{\lambda_i^i \lambda_{n-i}^{n-i} t^n}{(n-i)!\,i!} \exp\left[-(\lambda_i + \lambda_{n-i})t\right]. \qquad (5.24)$$

Формула (5.24) позволяет рассчитать суммарную вероятность того, что в течение времени t отказали 1 и $n-1$ элементов первого и второго каналов соответственно, 2 и $n-2$ элементов, …, i и n-i элементов,…, $n-1$ и 1 элементов обоих каналов.

Для определения вероятности опасного отказа необходимо для каждого указанного выше события определить вероятность возникновения эквивалентного отказа в двух каналах. Под эквивалентным отказом в двух каналах понимается событие отказа двух одинаковых элементов в обоих каналах. Данное событие может иметь место тогда, когда в одном кана-

ле отказал элемент i и в другом канале через некоторое время также отказал такой же элемент i. С этой целью рассмотрим следующую модель.

Пусть в двух урнах размещено одинаковое количество m шаров. Общее количество шаров в двух урнах равно $2m$. Все шары в каждой урне пронумерованы натуральными числами 1, 2, …, m. Из каждой урны произвольным образом изымается по одному шару до первого совпадения номеров шаров.

Лемма. Если суммарное количество шаров, изъятых из двух урн, равно m, из одной любой урны изъят хотя бы один шар, номера ранее изъятых из двух урн шаров не совпали, то вероятность совпадения номеров очередного изымаемого из любой урны шара и ранее изъятого шара равна $h = 1$.

Доказательство. Предположим из первой урны изъято $m - 1$ шаров, а из второй урны один шар. Поскольку номера всех изъятых шаров не совпали, то номер оставшегося в первой урне шара равен номеру шара, изъятого из второй урны. Если далее изымается шар из первой урны, то он обязательно совпадет с номером шара ранее изъятого из второй урны. Если же изымается второй шар из второй урны, то его номер обязательно совпадет с ранее изъятым шаром из первой урны. Теперь предположим, что из первой урны изъято $m - 2$ шаров, а из второй урны – 2 шара. Любой изъятый из первой урны шар из двух оставшихся по номеру совпадет с одним из двух ранее изъятых из второй урны шаров, и наоборот – номер третьего шара, изъятого из второй урны, обязательно совпадет с номером шара, ранее изъятого из первой урны. Продолжая указанные рассуждения, приходим к состоянию изъятия из второй урны $m - 1$ шаров при условии изъятия из первой урны одного шара. Поскольку все шары в урнах одинаковы и имеют идентичную нумерацию, то последнее состояние совпадает с первым. Это свидетельствует о том, что при всех возможных

состояниях изъятия из двух урн m шаров (при условии, что из одной урны изъят хотя бы один шар) при очередном изъятии шара его номер с вероятностью единица совпадет с номером ранее изъятого шара. Лемма доказана.

Теорема. Если каждый из двух одинаковых каналов системы можно разделить на m равных по интенсивности отказов функционально законченных элементов и число накопленных неэквивалентных отказов элементов в двух каналах равно n (где $n = 2, \ldots, m$), то вероятность того, что очередной i-й отказ элемента одного канала будет эквивалентным отказу элемента другого канала равна

$$h_i(n) = \frac{(n-i)i}{(m-i)(m-n+i)}, \qquad (5.25)$$

где i – число отказов элементов в одном канале ($i = 1, 2, \ldots, n-1$), а $(n-i)$ – число отказов элементов в другом канале.

Доказательство. Для доказательства представим оба канала в виде линеек из m одинаковых (по интенсивности отказов) элементов в каждой линейке (рис. III.5.11). Эквивалентный отказ возникнет при отказе двух элементов в разных каналах с одинаковыми номерами. Отказавшие ранее элементы каналов на рис. III.5.11 выделены темным цветом.

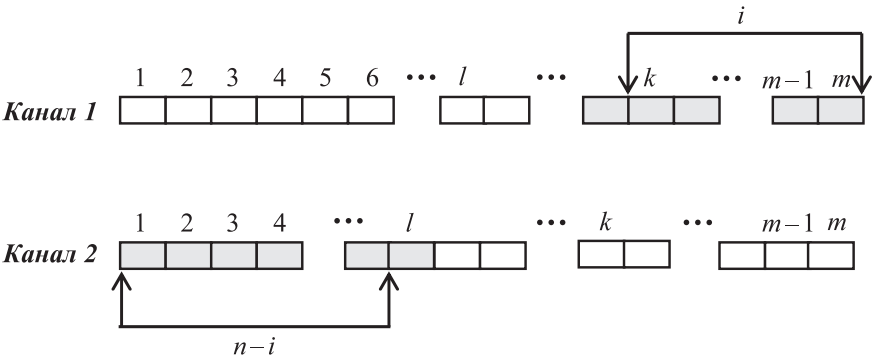


Рис. III.5.11. Модель двух каналов с отказавшими элементами

Эквивалентный отказ в двух каналах наступит в том случае, если возникнет отказ одного из элементов $1\ldots l-1$ одного канала (обозначим вероятность этого события $P(A)$) при условии того, что не откажет один из элементов $l\ldots k-1$ другого канала (обозначим условную вероятность этого события $P(B|A)$). Вероятность возникновения эквивалентного отказа (обозначим ее $P(B) = h_i(n)$) определяется по формуле произведения вероятностей зависимых событий $P(B) = P(A)P(B|A)$, где $P(B|A)$ – условная вероятность наступления события B при условии того, что произошло событие A.

Согласно рис. III.5.11: $P(A)=((n–i)/(m–i))$. Это вероятность того, что отказал один из $1\ldots k-1$ элементов одного канала, причем отказ этого элемента эквивалентен отказу одного из $1\ldots l-1$ элементов другого канала.

В свою очередь,

$$P(B \mid A) = 1 - \frac{m-n+i-i}{m-n+i} = \frac{i}{m-n+i}.$$

Это условная вероятность того, что при условии отказа одного из $1\ldots l-1$ элементов первого канала не откажет ни один из $1\ldots k-1$ элементов другого канала. Таким образом,

$$h_i(n) = \frac{(n-i)i}{(m-i)(m-n+i)},$$

что и требовалось доказать.

Если суммарное количество отказов элементов в обоих каналах в результате перезапусков накопилось до минимального числа $n = 3$, то имеют место следующие выражения:

$$h_1(3) = \frac{(3-1)1}{(m-1)(m-3+1)} = \frac{2}{(m-1)(m-2)};$$

$$h_2(3) = \frac{(3-2)2}{(m-2)(m-3+2)} = \frac{2}{(m-1)(m-2)}.$$

При $n = 4$

$$h_1(4) = \frac{(4-1)1}{(m-1)(m-4+1)} = \frac{3}{(m-1)(m-3)};$$

$$h_1(4) = \frac{(4-2)2}{(m-2)(m-4+2)} = \frac{4}{(m-2)(m-2)};$$

$$h_3(4) = \frac{(4-3)3}{(m-3)(m-4+3)} = \frac{3}{(m-1)(m-3)}.$$

При $n = m$ согласно формуле (5.25) имеет место равенство $h_1(m)=h_2(m)=\ldots=h_{m-1}(m)=1$, что соответствует доказанной выше лемме.

Приведенные примеры свидетельствуют о симметрии вероятностей возникновения эквивалентных отказов.

Подставив выражение (5.25) в формулу (5.24), находим вероятность возникновения опасного отказа в двухканальной системе в результате накопления n отказов при перезапусках каналов в течение времени t

$$Q_{\text{ОП}}(t,n) = \sum_{i=1}^{n-1} \frac{C_{n-2}^{n-1-i}}{2^n} \frac{\lambda_i^i \lambda_{n-i}^{n-i} t^n}{(n-i)!\,i!} \exp[-(\lambda_i + \lambda_{n-i})t] \frac{(n-i)i}{(m-i)(m-n+i)}, \quad (5.26)$$

где $\lambda_i = \frac{m-i}{m}\lambda;\ \lambda_{n-i} = \frac{m-n+i}{m}\lambda.$

На практике не всегда удается разделить канал обработки информации на одинаковые по интенсивности отказов функционально законченные элементы. Разброс интенсивностей отказов может составлять до 20%. Однако и в этих случаях оста-

ется справедливой формула (5.26), поскольку она не привязана к интенсивностям отказов, а учитывает только эквивалентные элементы в обоих каналах.

Некоторые результаты исследований вероятностей возникновения опасных отказов выполнены на примере двухканальной микропроцессорной системы автоматического управления движением поездов. Полученные результаты с учетом накопления отказов показаны в виде графических зависимостей на рис. III.5.12 и III.5.13. Они свидетельствуют о следующем:

– вероятность накопления в обоих каналах в течение года (10000 ч) более 5 отказов и возникновения в результате этого опасных отказов ничтожно мала (менее $Q_{ОП}(10000)=10^{-15}$) даже при явно завышенной интенсивности отказов канала $\lambda = 10^{-6}$ 1/ч (см. рис. III.5.12 и III.5.13);

– в реальном диапазоне значений интенсивностей отказов каналов в микропроцессорных системах управления движением

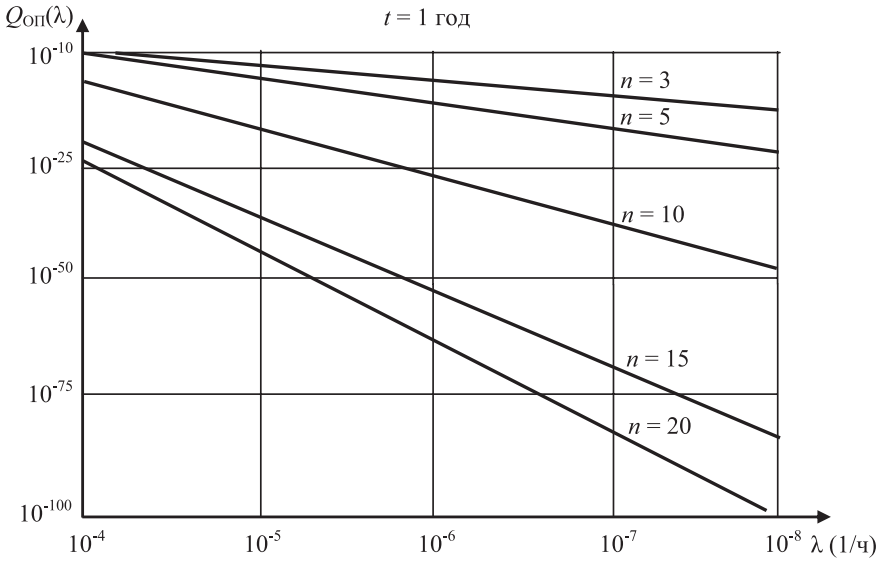


Рис. III.5.12. Зависимости вероятности опасного отказа канала обработки информации от интенсивности отказов и количества накопленных отказов в каналах (n) при времени работы канала один год

поездов (10^{-8} 1/ч < λ < 10^{-6} 1/ч) в течение года вероятно накопление от 3 до 5 отказов в обоих каналах. При этом вероятность возникновения опасных отказов системы в течение 10000 часов составляет менее 10^{-10} и более 10^{-22} (см. рис. III.5.12). Эти результаты свидетельствуют, во-первых, о возможности и целесообразности для обеспечения пропускной способности применять перезапуски каналов и, во-вторых, не следует ограничивать число перезапусков каналов;

– увеличение длительности работы двухканальной системы с перезапуском каналов без обслуживания каналов на порядок приводит к увеличению интенсивности отказов системы почти на четыре порядка (см. рис. III.5.13). Это обстоятельство свидетельствует о необходимости восстанавливать работоспособность элементов каналов не реже одного раза в год.

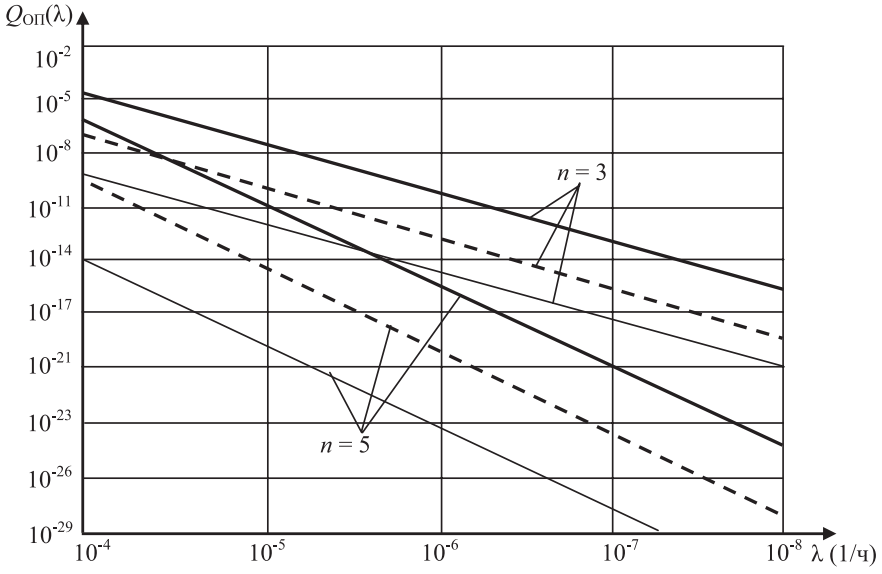


Рис. III.5.13. Зависимости вероятности опасного отказа канала обработки информации от интенсивности отказов и количества накопленных отказов в каналах (n = 3 и n = 5) при времени работы канала: один год (жирные сплошные линии); 1000 ч (штриховые линии); 100 ч (тонкие сплошные линии)

III.5.7. Организация информационной системы с комбинированной (двухуровневой) функциональной безопасностью

III.5.7.1. Проблемы обеспечения высоких уровней полноты безопасности

Ранее в п. III.5.2.1 рассмотрены уровни полноты безопасности информационной системы. Было отмечено, что для реализации системы с первым и даже со вторым уровнями полноты безопасности не требуется применения специальных мер защиты – достаточно руководствоваться требованиями стандартов качества изделий. Для достижения третьего уровня безопасности (УПБ 3) требуются более существенные усилия и более высокая компетенция разработчиков, чем для первого и второго уровней, требуется применение многоканальных аппаратных и многоверсионных программных средств. Важными факторами являются стоимость системы и время ее разработки. При этом выбор исполнителей становится ограниченным, так как не многие их них способны обеспечить этот уровень.

Четвертый уровень (УПБ 4) требует проведения разработки на грани искусства, включая применение формальных методов [57]. Стоимость проекта будет предельно большой и при создании системы потребуется исключительно высокая компетентность разработчиков систем. Однако для ряда ответственных и, особенно, критически важных систем требуется обеспечение безопасности функционирования на высоких уровнях требований.. Поэтому возникла потребность в создании новых методов обеспечения функциональной безопасности информационных систем на уровнях полноты безопасности УПБ 3 и УПБ 4, которые не требуют уникальности специалистов и пригодны для

широкого круга разработчиков безопасных информационных систем. Это важная, но не единственная причина.

Не менее значительными причинами, побудившими не ограничиваться известными технологиями обеспечения безопасности информационных систем, являются причины неуверенности в безопасности применяемых в информационных системах импортных аппаратных и программных средств. Проблема не только и не столько в том, что разработчик целенаправленно создает небезопасные продукты. Причина часто заключается в том, что во многих случаях не представляется возможным выявить все случайные ошибки в проектировании и изготовлении системы, которые со временем при определенных ситуациях трансформируются в опасные отказы. К этому следует добавить и целенаправленные воздействия на нарушения целостности и доступности информационных систем путем применения так называемых кибератак (п. III.1.7.2.). Защита от кибератак лежит в той же плоскости, что и защита от случайных проявлений угроз от аппаратных и программных средств. Обе группы угроз приводят к нарушению целостности и доступности информации. Основная цель нейтрализации кибератак и угроз технических средств, состоит в том, чтобы из небезопасных элементов создать безопасную систему, отвечающую требованиям третьего и даже четвертого уровней полноты безопасности. Для достижения этой цели над исходной системой следует создать некий защитный колпак, т.е. дополнительно один или более уровней защиты. Это может быть дополнительный комплекс технических средств, что приведет к дополнительным, порой чрезмерным, затратам. Это может быть некое сочетание небольшого дополнительного объема аппаратных средств и реализованных программно логических решений. Второй путь экономичен и, как будет показано ниже, эффективен.

III.5.7.2. Постановка задачи

Исследуется информационная система управления с комбинированной (двухуровневой) архитектурой для обеспечения функциональной безопасности. Она состоит из двух небезопасных систем и безопасного устройства управления ими. Под небезопасной системой понимается такая система, вероятность (интенсивность) опасного отказа которой ниже требований третьего или четвертого уровня полноты безопасности. Под опасным отказом понимается событие, при котором система не находится ни в работоспособном, ни в защитном состоянии. В частности, состояние необнаруженного отказа является опасным. Безопасным считается устройство, отвечающее требованиям о третьего или четвертого уровня полноты безопасности.

В каждой информационной системе содержится определенным образом организованная подсистема контроля и обнаружения опасных отказов. Это может быть либо встроенная аппаратная подсистема, либо программная, либо аппаратно-программная, либо подсистема внешнего функционального контроля, либо комбинированная подсистема, построенная с применением перечисленных методов и средств контроля. Полнота охвата контролем, непрерывность, достоверность контроля, оперативность принятия решения об обнаружении опасного отказа системы могут оцениваться частными количественными показателями. В данной задаче можно ограничиться комплексным показателем эффективности контроля и обнаружения опасного отказа, а именно вероятностью правильного обнаружения опасных отказов. Этот показатель, который имеет разное значение для каждой из двух систем (α_1 и α_2 соответственно) и для устройства управления (α_0). События не обнаружения опасных отказов наступают в исследуемой системе с вероятностями $\overline{\alpha}_1 = 1 - \alpha_1$, $\overline{\alpha}_2 = 1 - \alpha_2$, $\overline{\alpha}_0 = 1 - \alpha_0$.

Рассматриваются две исходные стратегии обеспечения функциональной безопасности комбинированной (двухуровневой) системы (ДС):

Стратегия 1. При обнаружении отказа штатными средствами или функциональным контролем одной из двух небезопасных систем ДС продолжает функционировать, опираясь на результаты работы исправной системы.

Опасные отказы любой из двух составных небезопасных систем трансформируются на уровне ДС в неопасные отказы. К опасным отказам ДС могут привести следующие события: необнаруженный отказ устройства управления; необнаруженные отказы (опасные отказы) двух составных систем; обнаруженный отказ одной системы и необнаруженный отказ другой системы.

Стратегия 2. В течение цикла управления в устройстве управления обеспечение надежности и безопасности информационной системы осуществляется путем периодического сравнения результатов работы двух разных составных систем, выполняющих одинаковые системные функции принятия решения, но с помощью разных технических средств. В случае несовпадения результатов приоритет отдается той составной системе, которая предложила более осторожное решение. Остальные условия функционирования аналогичны тем, которые описаны в стратегии 1.

Стратегии 31 и 32. Эти стратегии основаны на запоминании, анализе, сопоставлениями с указаниями составных систем логических последовательностей смены состояний управления. Например, на железнодорожном транспорте анализируются логические последовательности смены состояний напольного оборудования систем автоматики и телемеханики. Все это позволяет информационной системе управления в течение среднего времени $\overline{t}_{\text{р}} = 1/\gamma$ (γ – интенсивность принятия решения) принимать достоверные решения о наличии отказа, который не

обнаружен штатными средствами контроля составной системы железнодорожной автоматики и телемеханики.

Эти возможности логического анализа дают основание для исследования двух новых стратегий: ***31*** – принятие решения о наличии необнаруженного отказа одной составной системы в условиях стратегии 1; стратегия ***32*** – принятие решения о наличии необнаруженного отказа одной составной системы в условиях стратегии 2.

III.5.8. Модели функциональной безопасности комбинированной двухуровневой системы

III.5.8.1. Оценка безопасности двухуровневой системы, построенной по стратегии 1

Принятые предпосылки.

Потоки опасных отказов двух систем и устройства управления ими являются простейшими с интенсивностями λ_1, λ_2, λ_0 соответственно. Это предположение базируется на том, что поток опасных отказов представляет собой многократно разряженный поток отказов элементов исследуемого устройства или системы. В соответствии с теорией редеющих потоков многократное случайное разрежение исходного произвольного потока приводит к простейшему потоку [49]. Потоки восстановлений отказов составных систем в рассматриваемой системе также простейшие с одинаковой интенсивностью μ.

Для выбора наиболее эффективной из указанных выше стратегий можно ограничиться такими показателями функциональной безопасности как: средняя наработка ДС до опасного отказа, средняя наработка ДС до защитного отказа, среднеквадратичные отклонения наработки ДС до опасного и защитного отказов соответственно.

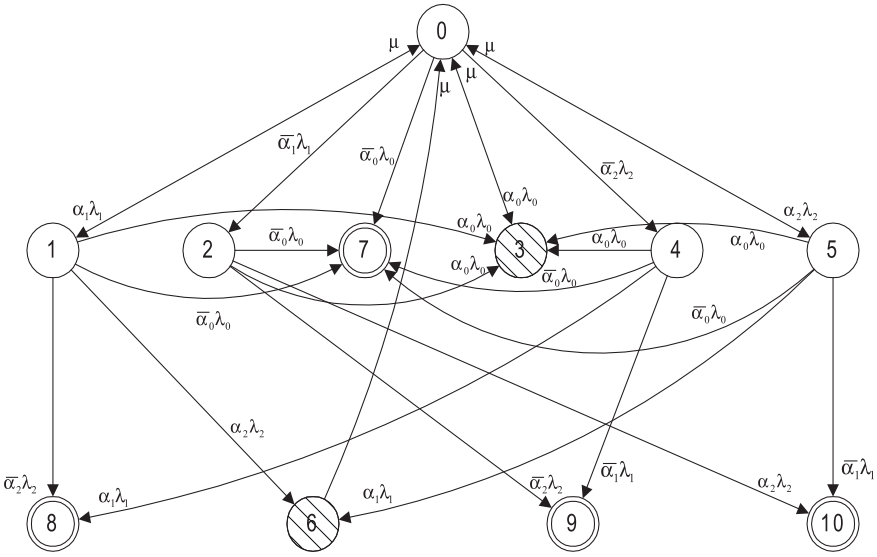


Рис. III.5.14. Граф состояний ДС с двумя небезопасными составными системами и системой управления

Модели исследования стратегии *1*

На рис. III.5.14 приведена графовая модель ДС с двумя небезопасными составными системами и системой управления.

Состояния ДС:

0 – все три системы работают без опасных отказов;

1 – отказ в первой системы с интенсивностью λ_1. Он обнаружен с вероятностью α_1 (интенсивность перехода 0-1 $\alpha_1\lambda_1$);

2 – опасный отказ ДС за счет необнаруженного отказа первой системы, отказ не обнаружен с вероятностью $1-\alpha_1=\overline{\alpha}_1$;

3 – отказ в системе управления ДС с интенсивностью λ_0. Он обнаружен с вероятностью α_0 – защитный отказ ДС;

4 – опасный отказ ДС за счет необнаруженного отказа второй составной системы;

5 – отказ во второй системе с интенсивностью λ_2. Отказ обнаружен с вероятностью α_2;

6 – возникли обнаруженные отказы двух составных систем – защитный отказ ДС;

7 – опасный отказ ДС в результате отказа системы управления с интенсивностью λ_0, отказ не обнаружен с вероятностью $1-\alpha_0=\bar{\alpha}_0$;

8 – опасный отказ ДС – отказали обе системы, обнаружен отказ первой системы и не обнаружен отказ второй системы;

9 – опасный отказ ДС – возникли необнаруженные отказы двух составных систем;

10 – опасный отказ ДС – отказали обе системы, обнаружен отказ второй системы и не обнаружен опасный отказ первой системы.

Критерии защитных отказов

Обнаружен отказ системы управления, либо отказ в любой одной системе и обнаружен отказ в системе управления (состояние 3), либо возникли и обнаружены отказы в обеих системах при исправной системе управления (состояние 6). Состояния 3 и 6 защитных отказов на рис. III.5.14. заштрихованы.

Критерии опасных отказов системы

Не обнаружен отказ в системе управления, либо опасный отказ в любой одной системе и не обнаружен отказ в системе управления (состояние 7), либо при исправной системе управления возникли опасные отказы в обеих составных системах, причем оба отказа не обнаружены (состояние 9) или обнаружен только один из двух отказов (состояния 8 и 10).

Таким образом, множество неопасных состояний в системе $S_{\text{Н}}=\{0, 1, 2, 3, 4, 5, 6\}$, множество защитных состояний $S_{\text{З}}=\{3, 6\}$, множество опасных состояний $\overline{S}_{\text{Н}}=\{7, 8, 9, 10\}$.

Определяют выбранные показатели функциональной безопасности и среднюю наработку до защитного отказа ДС. Для этого предварительно во множестве состояний $S_{\text{Н}}$ находят математические ожидания времени пребывания системы в каждом из состояний этого множества, а также вероятности переходов между его состояниями:

$$T_0 = \frac{1}{\lambda_0 + \lambda_1 + \lambda_2};\ T_1 = \frac{1}{\lambda_0 + \lambda_2 + \mu};\ T_2 = \frac{1}{\lambda_0 + \lambda_2};$$

$$T_3 = \frac{1}{\mu};\ T_4 = \frac{1}{\lambda_0 + \lambda_1};\ T_5 = \frac{1}{\lambda_0 + \lambda_1 + \mu};\ T_6 = \frac{1}{\mu};$$

$$p_{01} = \frac{\alpha_1\lambda_1}{\lambda_0 + \lambda_1 + \lambda_2};\ p_{02} = \frac{\bar{\alpha}_1\lambda_1}{\lambda_0 + \lambda_1 + \lambda_2};\ p_{03} = \frac{\alpha_0\lambda_0}{\lambda_0 + \lambda_1 + \lambda_2};$$

$$p_{04} = \frac{\bar{\alpha}_2\lambda_2}{\lambda_0 + \lambda_1 + \lambda_2};\ p_{05} = \frac{\alpha_2\lambda_2}{\lambda_0 + \lambda_1 + \lambda_2};\ p_{10} = \frac{\mu}{\lambda_0 + \lambda_2 + \mu};$$

$$p_{13} = \frac{\alpha_0\lambda_0}{\lambda_0 + \lambda_2 + \mu};\ p_{16} = \frac{\alpha_2\lambda_2}{\lambda_0 + \lambda_2 + \mu};\ p_{23} = \frac{\alpha_0\lambda_0}{\lambda_0 + \lambda_2};\ p_{30}=1;$$

$$p_{43} = \frac{\alpha_0\lambda_0}{\lambda_0 + \lambda_1};\ p_{50} = \frac{\mu}{\lambda_0 + \lambda_1 + \mu};\ p_{53} = \frac{\alpha_0\lambda_0}{\lambda_0 + \lambda_1 + \mu};$$

$$p_{56} = \frac{\alpha_1\lambda_1}{\lambda_0 + \lambda_1 + \mu};\ p_{60}=1.$$

Затем находят аналитические выражения показателей функциональной безопасности ДС. Например, показатель средней наработки до опасного отказа и до защитного состояния системы соответственно определяются в следующем виде:

$$T_{\text{ОП}_1} = \frac{\begin{pmatrix} T_0 + p_{01}T_1 + p_{02}T_2 + p_{04}T_4 + p_{05}T_5 + \\ +(p_{03} + p_{01}p_{13} + p_{02}p_{23} + p_{04}p_{43} + p_{05}p_{53})T_3 + p_{01}p_{16}T_6 \end{pmatrix}}{\begin{pmatrix} 1-(p_{01}p_{10} + p_{05}p_{50} + p_{03} + p_{01}p_{13} + \\ +p_{02}p_{23} + p_{04}p_{43} + p_{05}p_{53} + p_{01}p_{16} + p_{05}p_{56}) \end{pmatrix}}. \quad (5.27)$$

Средняя наработка до защитного отказа вычисляется в множестве состояний $\overline{S}_3=\{0, 1, 2, 4, 5\}$

$$T_{31}=\frac{T_0+p_{01}T_1+p_{02}T_2+p_{04}T_4+p_{05}T_5}{1-p_{01}p_{10}-p_{05}p_{50}}. \qquad (5.28)$$

Среднеквадратичные отклонения времен до опасного и защитного отказов находятся с помощью формул (4.26) и (4.27) [1].

Результаты исследований временных показателей функциональной безопасности ДС, а также среднеквадратичных отклонений времен до опасного и защитного отказов для следующих значений входных данных: $\lambda_1=10^{-8}$; $\lambda_2=10^{-6}$; $\lambda_0=10^{-10}$; $\alpha_1=\alpha_2=0{,}80$; 0,85; 0,90; 0,95; 0,99; $\alpha_0=0{,}99$; $\mu=0{,}05$; 1; $\gamma=1$; 50 – приведены на рис. III.5.15. Графики $T_{\text{ОП}}$, T_3 и соответственно $\sigma_{\text{ОП}}$, σ_3 практически совпадают.

Установлено, что функциональная безопасность ДС, построенной по стратегии 1, находится на уровне функциональной безопасности лучшей из двух небезопасных составных систем, а именно первой системы, у которой интенсивность отказов на два порядка ниже, чем у второй составной системы.

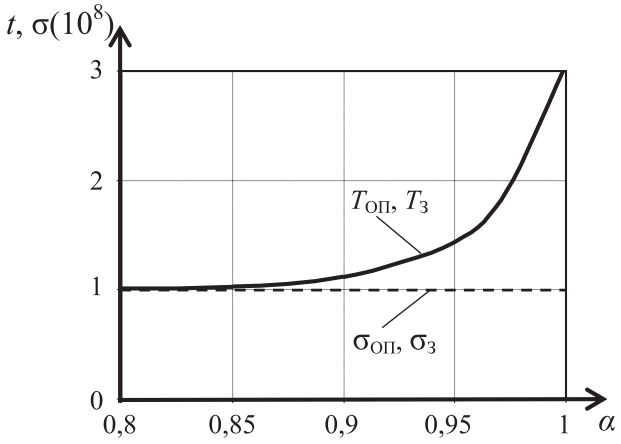


Рис. III.5.15. Результаты оценки эффективности стратегии 1 построения ДС с двумя составными небезопасными системами

При повышении эффективности собственного контроля составных систем наблюдается небольшой рост уровня безопасности ДС по сравнению с первой составной системой – в реальном диапазоне эффективности собственного контроля составных систем 0,85 – 0,95 средняя наработка до опасного отказа ДС $T_{ОП}$ и средняя наработка до защитного отказа ДС $T_З$ увеличиваются в среднем на 20%. При этом среднеквадратичные значения этих времен ($\sigma_{ОП}$, $\sigma_З$ соответственно) практически не зависят от роста эффективности контроля и несколько меньше показателей наработки до отказов, что свидетельствует о приемлемой достоверности оценок показателей $T_{ОП}$ и $T_З$.

В целом стратегия 1 не обеспечивает существенного прорыва в создании безопасной ДС.

Анализ влияния составных параметров графовой модели рис. III.5.14. показал, что связи между вершинами 1-3, 2-3, 4-3 и 5-3 практически не оказывают влияния на результаты исследования. Это обстоятельство позволяет упростить граф состояний и получить обозримую формулу средней наработки до опасного отказа ДС, построенной по стратегии 1:

$$T_{ОП1} = \frac{T_0 + p_{01}T_1 + p_{02}T_2 + p_{04}T_4 + p_{05}T_5 + p_{03}T_3 + p_{01}p_{16}T_6}{1-(p_{01}p_{10} + p_{05}p_{50} + p_{03} + p_{01}p_{16})}. \quad (5.29)$$

При этом формула расчета средней наработки до защитного отказа остается прежней (см. (5.28)).

III.5.8.2. Безопасность комбинированной системы, построенной по стратегии 2

Приняты предпосылки такие же, как и для стратегии 1. На рис. III.5.16 приведена графовая модель ДС с двумя небезопасными системами и системой управления, в которой принято в случае обнаружения отказа любой одной небезопасной системы формировать защитный отказ двухуровневой системы.

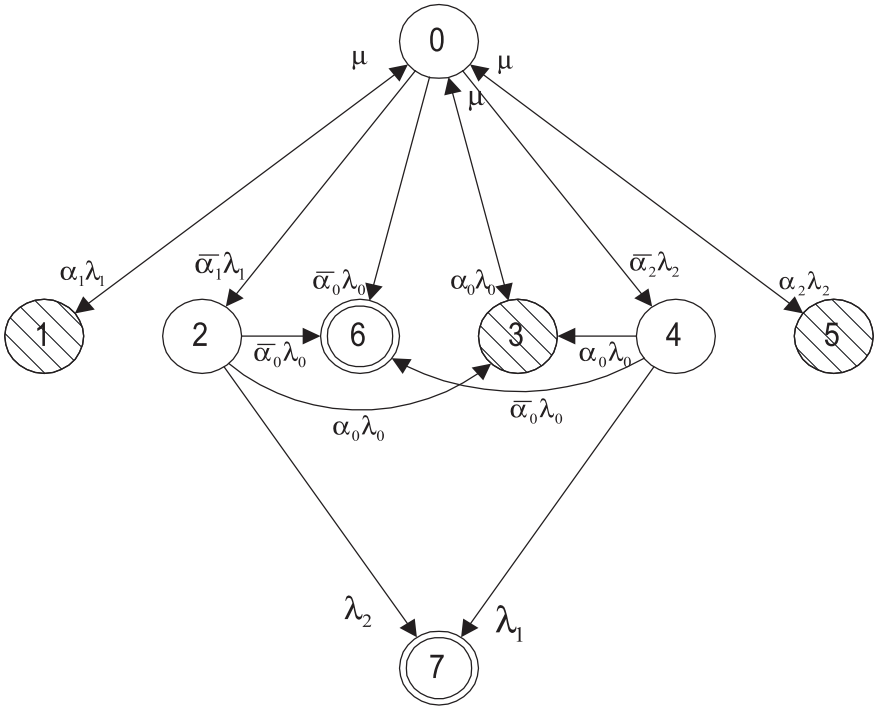


Рис. III.5.16. Графовая модель безопасности двухуровневой системы, построенной по стратегии 2

Состояния ДС:

0 – все три системы работают без опасных отказов;

1 – отказ в первой системе с интенсивностью λ_1. Он обнаружен с вероятностью α_1 (интенсивность перехода 0-1 $\alpha_1\lambda_1$);

2 – опасный отказ в первой системе не обнаружен с вероятностью $1-\alpha_1=\bar{\alpha}_1$;

3 – отказ в системе управления ДС с интенсивностью λ_0. Он обнаружен с вероятностью α_0;

4 – не обнаруженный опасный отказ во второй системе с интенсивностью λ_2;

5 – отказ во второй системе обнаружен с вероятностью α_2;

6 – опасный отказ системы управления с интенсивностью λ_0. Отказ не обнаружен с вероятностью $1-\alpha_0=\bar{\alpha}_0$;

7 – возникли необнаруженные опасные отказы двух составных систем.

Критерии защитных отказов

Обнаружен отказ любой одной системы (состояния 1, 5) или системы управления (состояние 3), либо необнаруженный отказ в любой одной системы и обнаруженный отказ в системе управления (состояние 3). Состояния 1, 3 и 5 защитных отказов на III.5.16 заштрихованы.

Критерии опасных отказов двухуровневой системы

Не обнаружен отказ в системе управления, либо опасный отказ в любой одной системе и не обнаружен отказ в системе управления (состояние 6), либо при исправной системе управления возникли отказы в обеих составных системах, причем не обнаружен отказ хотя бы в одной из двух отказавших систем (состояние 7).

Таким образом, множество неопасных состояний в системе $S_{\text{Н}} = \{0, 1, 2, 3, 4, 5\}$, множество защитных состояний $S_{\text{З}} = \{1, 3, 5\}$, множество опасных состояний $\overline{S}_{\text{Н}} = \{6, 7\}$.

Руководствуясь графовым методом и алгоритмом его реализации, определяют выбранные показатели функциональной безопасности, с учетом математических ожиданий времени пребывания системы в каждом из состояний множества $S_{\text{Н}}$, а также вероятности переходов между его состояниями:

$$T_0=1/(\lambda_0+\lambda_1+\lambda_2);\ T_1=(1/\mu);\ T_2=1/(\lambda_0+\lambda_2);\ T_3=(1/\mu);\ T_4=1/(\lambda_0+\lambda_1);$$

$$T_5=(1/\mu);\ p_{01}=\frac{\alpha_1\lambda_1}{\lambda_0+\lambda_1+\lambda_2};\ p_{02}=\frac{\overline{\alpha}_1\lambda_1}{\lambda_0+\lambda_1+\lambda_2};$$

$$p_{03}=\frac{\alpha_0\lambda_0}{\lambda_0+\lambda_1+\lambda_2};\ p_{04}=\frac{\overline{\alpha}_2\lambda_2}{\lambda_0+\lambda_1+\lambda_2};\ p_{05}=\frac{\alpha_2\lambda_2}{\lambda_0+\lambda_1+\lambda_2};$$

$$p_{10}=1;\ p_{23}=\frac{\alpha_0\lambda_0}{\lambda_0+\lambda_2};\ p_{30}=1;\ p_{43}=\frac{\alpha_0\lambda_0}{\lambda_0+\lambda_1};\ p_{50}=1.$$

Затем находят аналитические выражения показателей функциональной безопасности ДС:

$$T_{\text{ОП2}}=\frac{\left(\begin{array}{c}T_0+p_{01}T_1+p_{02}T_2+p_{04}T_4+p_{05}T_5+\\+(p_{03}+p_{02}p_{23}+p_{05}p_{53})T_3\end{array}\right)}{1-(p_{01}+p_{05}+p_{03}+p_{02}p_{23}+p_{04}p_{43})}. \qquad (5.30)$$

Средняя наработка до защитного отказа определяется в множестве состояний $\overline{S}_3=\{0, 2, 4\}$ по формуле

$$T_{32}=T_0+p_{02}T_2+p_{04}T_4. \qquad (5.31)$$

Результаты оценки установленных временных показателей функциональной безопасности ДС, а также среднеквадратичных отклонений этих времен соответственно до опасного и защитного отказов для ранее указанных диапазонов значений входных данных применительно к стратегии 2 построения ДС приведены на рис. III.5.17 а, б.

Полученные результаты свидетельствуют о том, что с ростом эффективности контроля составных систем ДС, построенной по стратегии 2, увеличивается количество обнаруженных отказов этих систем и, следовательно, уменьшается средняя наработка T_3 до защитного отказа ДС. Уровень T_3 изменяется в пределах одного порядка– от значения, примерно равного среднеарифметическому значению средних наработок до отказа составных систем, до значения средней наработки до отказа менее надежной из двух составных систем. При этом среднеквадратичное отклонение наработки времени до защитного отказа ДС существенно больше времени T_3, что свидетельствует о большом разбросе наработки до защитного отказа ДС.

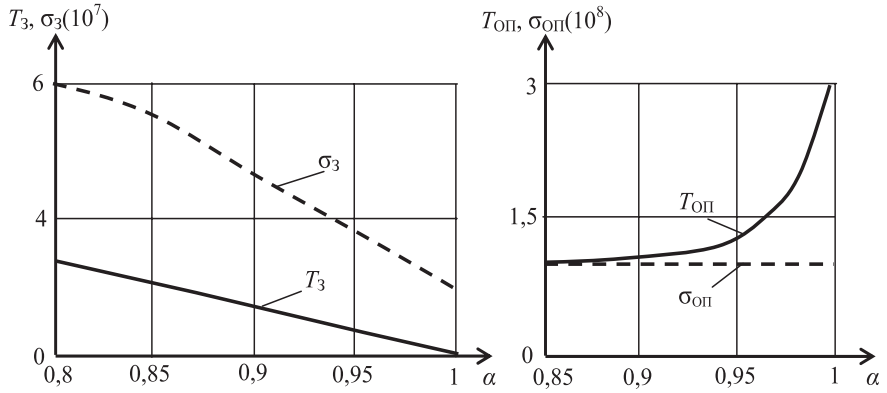


Рис. III.5.17. Результаты оценки эффективности стратегии 2 построения ДС с двумя составными небезопасными системами.

Результаты исследований средней наработки до опасного отказа ДС, построенной по стратегии 2, совпадают с результатами исследований этого показателя функциональной безопасности ДС, построенной по стратегии 1.

В целом математическое моделирование двух стратегий построения ДС показало предпочтительность стратегии 1 перед стратегией 2, поскольку в обоих случаях имеют место практически одинаковые интенсивности опасных отказов ДС, примерно равные интенсивности опасных отказов лучшей из двух составных систем. При этом существенно повышается интенсивность защитных отказов ДС в случае стратегии 2 и, следовательно, понижается эффективность работы двухуровневой системы в целом.

III.5.8.3. Безопасность комбинированной системы, построенной по стратегии 31.

На рис. III.5.18 приведена графовая модель безопасности ДС с двумя небезопасными системами, в которой устройство управления в условиях стратегии ***1*** в течение среднего времени $\overline{t}_р = 1/\gamma$ принимает достоверные решения о наличии отказа

составной системы, необнаруженного штатными средствами контроля.

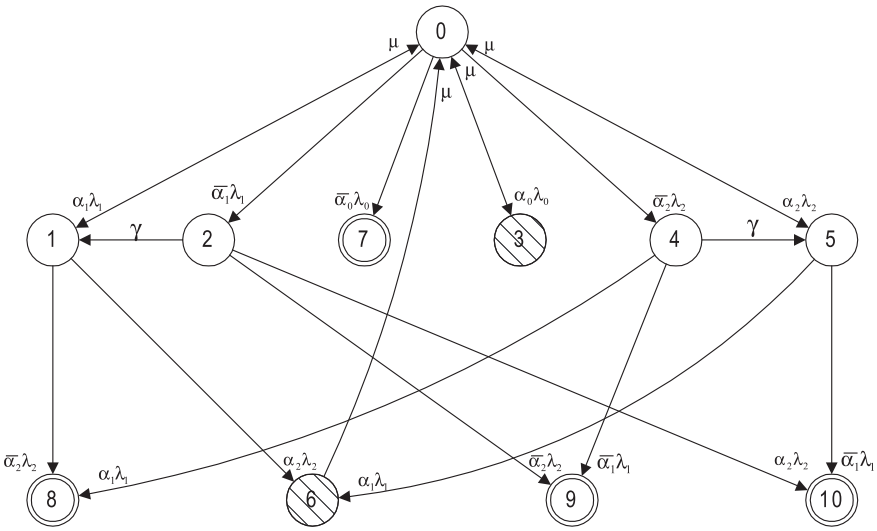


Рис. III.5.18. Графовая модель безопасности ДС, построенной по стратегии 31.

Состояния ДС:

0 – все три системы работают без опасных отказов;

1 – отказ в первой системе с интенсивностью λ_1. Он обнаружен с вероятностью α_1 штатными средствами (интенсивность перехода 0-1 $\alpha_1\lambda_1$) или с интенсивностью γ на уровне ДС в случае пропуска этого отказа штатными средствами первой системы;

2 – неопасный отказ в ДС при отказе первой системы и исправности второй системы. Отказ первой системы не обнаружен с вероятностью $1-\alpha_1=\bar{\alpha}_1$;

3 – отказ в системе управления ДС с интенсивностью λ_0. Он обнаружен с вероятностью α_0 – защитный отказ;

4 –неопасный отказ ДС за счет необнаруженного отказа во второй системе;

5 – отказ во второй системе с интенсивностью λ_2. Отказ обнаружен с вероятностью α_2;

6 – возникли обнаруженные отказы двух составных систем – защитный отказ ДС;

7 – опасный отказ системы управления с интенсивностью λ_0. Отказ не обнаружен с вероятностью $1-\alpha_0 = \overline{\alpha}_0$;

8 – опасный отказ ДС: отказали обе системы, обнаружен отказ первой системы и не обнаружен отказ второй системы;

9 – опасный отказ ДС: возникли необнаруженные отказы двух составных систем;

10 – опасный отказ ДС: отказали обе системы; обнаружен отказ второй системы и не обнаружен отказ первой системы.

Критерии защитных отказов

Обнаружен отказ системы управления, либо отказ в любой одной составной системе и обнаружен отказ в системе управления (состояние 3), либо возникли и обнаружены отказы в обеих системах при исправной системе управления (состояние 6). Состояния 3 и 6 защитных отказов на рис. III.5.18 заштрихованы.

Критерии опасных отказов системы

Не обнаружен отказ в системе управления, либо опасный отказ в любой одной составной системе и не обнаружен отказ в системе управления (состояния 7), либо при исправной системе управления возникли отказы в обеих системах, причем не обнаружен отказ хотя бы в одной из двух отказавших систем (состояния 8 или 10), либо возникли необнаруженные отказы в двух составных системах (состояние 9).

Таким образом, множество неопасных состояний в двухуровневой системе $S_{\text{Н}} = \{0, 1, 2, 3, 4, 5\}$, множество защитных состояний $S_{\text{З}} = \{1, 3, 6\}$, множество опасных состояний $\overline{S}_{\text{Н}} = \{7, 8, 9, 10\}$.

Математические ожидания времени пребывания системы в каждом из состояний множества $S_{\text{Н}}$, а также вероятности переходов между его состояниями имеют следующий вид:

$T_0=1/(\lambda_0+\lambda_1+\lambda_2)$; $T_0=1/(\lambda_0+\lambda_1+\lambda_2)$; $T_1=1/(\lambda_0+\lambda_2+\mu)$; $T_2=1/(\lambda_0+\lambda_2+\gamma)$;

$T_3=(1/\mu)$; $T_4=1/(\lambda_0+\lambda_1+\gamma)$; $T_5=1/(\lambda_0+\lambda_1+\mu)$; $T_6=(1/\mu)$;

$$p_{01}=\frac{\alpha_1\lambda_1}{\lambda_0+\lambda_1+\lambda_2};\ p_{02}=\frac{\bar{\alpha}_1\lambda_1}{\lambda_0+\lambda_1+\lambda_2};\ p_{03}=\frac{\alpha_0\lambda_0}{\lambda_0+\lambda_1+\lambda_2};$$

$$p_{04}=\frac{\bar{\alpha}_2\lambda_2}{\lambda_0+\lambda_1+\lambda_2};\ p_{05}=\frac{\alpha_2\lambda_2}{\lambda_0+\lambda_1+\lambda_2};\ p_{10}=\frac{\mu}{\lambda_0+\lambda_2+\mu};$$

$$p_{16}=\frac{\alpha_2\lambda_2}{\lambda_0+\lambda_2+\mu};\ p_{21}=\frac{\gamma}{\lambda_0+\lambda_2+\gamma};\ p_{30}=1;\ p_{45}=\frac{\gamma}{\lambda_0+\lambda_2+\gamma};$$

$$p_{50}=\frac{\mu}{\lambda_0+\lambda_1+\mu};\ p_{56}=\frac{\alpha_1\lambda_1}{\lambda_0+\lambda_1+\mu};\ p_{60}=1.$$

Затем находим аналитические выражения показателей функциональной безопасности ДС:

$$T_{\text{ОП31}}=\frac{\left(\begin{array}{c}T_0+p_{01}T_1+p_{02}T_2+p_{04}T_4+p_{05}T_5+(p_{03}+p_{01}p_{13}+\\+p_{02}p_{23}+p_{04}p_{43}+p_{05}p_{53})T_3+p_{01}p_{16}T_6\end{array}\right)}{\left(\begin{array}{c}1-((p_{01}+p_{02}p_{21})p_{10}+(p_{05}+p_{04}p_{45})p_{50}+\\+p_{03}+(p_{01}+p_{02}p_{21})p_{16}+(p_{05}+p_{04}p_{45})p_{56}))\end{array}\right)}. \quad (5.32)$$

Средняя наработка до защитного отказа определяется в множестве состояний $\bar{S}_3=\{0, 1, 2, 4, 5\}$ по формуле

$$T_{331}=\frac{T_0+(p_{01}+p_{02}p_{21})T_1+p_{02}T_2+p_{04}T_4+(p_{05}+p_{04}p_{45})T_5}{1-(p_{01}+p_{02}p_{21})p_{10}-(p_{05}+p_{04}p_{45})p_{50}}. \quad (5.33)$$

Результаты исследований установленных временных показателей функциональной безопасности ДС, а также среднеквадратичных отклонений этих времен соответственно до опасного и защитного отказов для ранее указанных в п. III.5.8.1. диапазо-

нов значений входных данных применительно к стратегии 31 построения ДС приведены на рис. III.5.19.

Полученные результаты исследования показывают высокую эффективность применения стратегии 31 построения ДС – средняя наработка до опасного отказа возрастет предположительно на 3 – 4 порядка, а средняя наработка до защитного отказа – на 2 порядка по сравнению с теми показателями, которые до построения ДС по данной стратегии имели место у лучшей по безопасности из двух составных информационных систем.

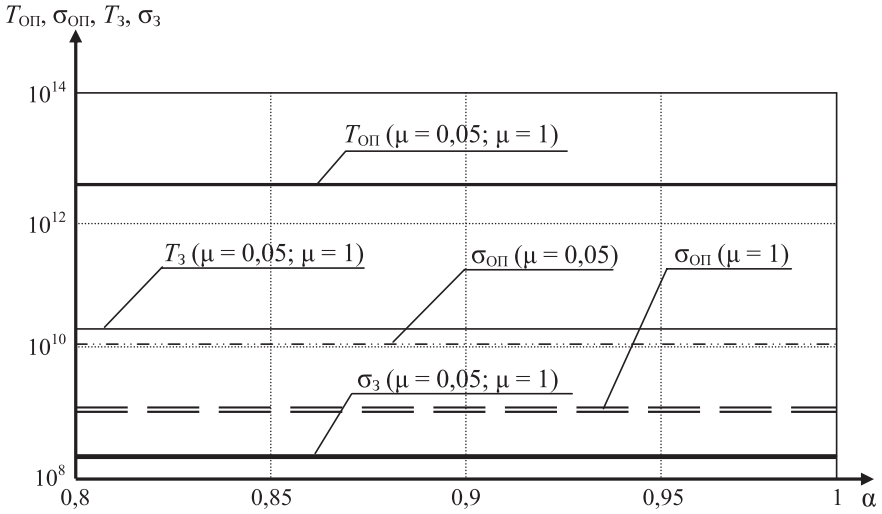


Рис. III.5.19. Результаты исследований эффективности стратегии 31 построения ДС

III.5.8.4. Безопасность комбинированной системы, построенной по стратегии 32.

На рис. III.5.20 приведена графовая модель исследования ДС с двумя небезопасными информационными системами и системой управления, в которой устройство управления в условиях стратегии 2 в течение среднего времени $\overline{t}_р = 1/\gamma$ принимает достоверные решения о наличии отказа состав-

ной информационной системы, необнаруженного штатными средствами контроля.

Состояния ДС:

0 – все три системы работают без опасных отказов;

1 – отказ в первой системе с интенсивностью λ_1 – защитный отказ ДС. Он обнаружен с вероятностью α_1 штатными средствами (интенсивность перехода 0-1 $\alpha_1\lambda_1$) или с интенсивностью γ на уровне ДС в случае пропуска этого отказа штатными средствами первой системы;

2 – неопасный отказ в ДС при отказе первой системы и исправности второй системы. Отказ первой системы не обнаружен с вероятностью $1-\alpha_1=\overline{\alpha}_1$;

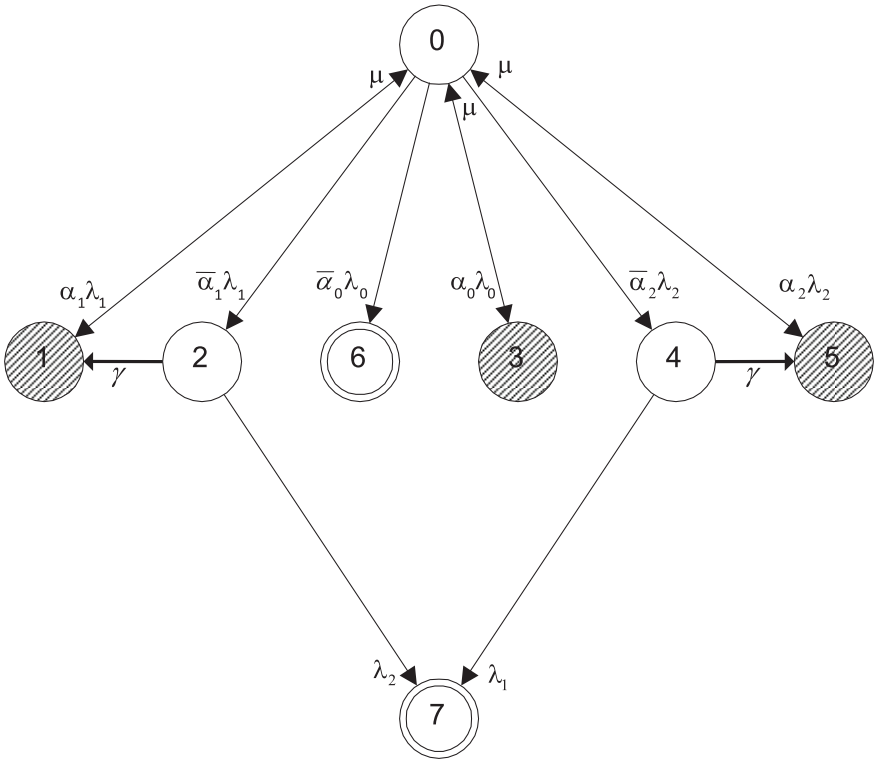


Рис. III.5.20. Графовая модель безопасности ДС, построенной по стратегии 32

3 – отказ в системе управления ДС с интенсивностью λ_0. Он обнаружен с вероятностью α_0 – защитный отказ;

4 – неопасный отказ в ДС при отказе второй системы и исправности первой системы. Отказ второй системы не обнаружен с вероятностью $1-\alpha_2=\overline{\alpha}_2$;

5 – отказ во второй системе с интенсивностью λ_2 – защитный отказ. Отказ обнаружен с вероятностью α_2 штатными средствами (интенсивность перехода 0-1 $\alpha_2\lambda_2$) или с интенсивностью γ на уровне ДС в случае пропуска этого отказа штатными средствами второй системы;

6 – опасный отказ системы управления с интенсивностью λ_0. Отказ не обнаружен с вероятностью $1-\alpha_0=\overline{\alpha}_0$;

7 – опасный отказ ДС: возникли отказы двух систем при условии, что отказ хотя бы одной системы не обнаружен.

Критерии защитных отказов

Обнаружен отказ системы управления (состояние 3, либо отказ в любой одной системе (состояния 1 и 5).

Критерии опасных отказов системы

Не обнаружен отказ в системе управления (состояние 6), либо возникли необнаруженные отказы в двух составных системах (состояние 7). Таким образом, множество неопасных состояний в системе $S_{\text{Н}} = \{0, 1, 2, 3, 4, 5\}$, множество защитных состояний $S_{\text{З}} = \{1, 3, 5\}$, множество опасных состояний $\overline{S}_{\text{Н}} = \{6, 7\}$.

Во множестве состояний $S_{\text{Н}}$ находим математические ожидания времени пребывания системы в каждом из состояний этого множества, а также вероятности переходов между его состояниями:

$$T_0=\frac{1}{\lambda_0+\lambda_1+\lambda_2};\ T_2=\frac{1}{\lambda_0+\lambda_2+\gamma};\ T_1=T_3=T_5=\frac{1}{\mu};$$

$$T_4=\frac{1}{\lambda_0+\lambda_1+\gamma};\ p_{01}=\frac{\alpha_1\lambda_1}{\lambda_0+\lambda_1+\lambda_2};\ p_{02}=\frac{\overline{\alpha}_1\lambda_1}{\lambda_0+\lambda_1+\lambda_2};$$

$$p_{03} = \frac{\alpha_0\lambda_0}{\lambda_0+\lambda_1+\lambda_2};\ p_{04} = \frac{\bar{\alpha}_2\lambda_2}{\lambda_0+\lambda_1+\lambda_2};\ p_{05} = \frac{\alpha_2\lambda_2}{\lambda_0+\lambda_1+\lambda_2};$$

$$p_{10}=1;\ p_{21} = \frac{\gamma}{\lambda_0+\lambda_2+\gamma};\ p_{30}=1;\ p_{45} = \frac{\gamma}{\lambda_0+\lambda_2+\gamma};\ p_{50}=1.$$

Затем находим аналитические выражения показателей функциональной безопасности ДС:

$$T_{\text{ОП32}} = \frac{\begin{pmatrix} T_0 + (p_{01}+p_{02}p_{21})T_1 + p_{02}T_2 + p_{04}T_4 + \\ +(p_{05}+p_{04}p_{45})T_5 + p_{03}T_3 \end{pmatrix}}{1-(p_{01}+p_{02}p_{21}+p_{05}+p_{04}p_{45}+p_{03})}. \quad (5.34)$$

Средняя наработка до защитного отказа определяется во множестве состояний $\overline{S}_3 = \{0, 2, 4\}$ по формуле

$$T_{332} = T_0 + p_{02}T_2 + p_{04}T_4. \quad (5.35)$$

Результаты исследований установленных временных показателей функциональной безопасности ДС, а также среднеквадратичных отклонений этих времен соответственно до опасного и защитного отказов для ранее указанных диапазонов значений входных данных применительно к стратегии ***32*** построения ДС приведены на рис. III.5.21.

Полученные результаты исследования показывают, что стратегия 32 построения ДС существенно лучше стратегий 1 и 2, но по показателю средней наработки до защитного отказа значительно (более чем на два порядка) уступает стратегии 31. При этом средняя наработка до опасного отказа возрастет так же, как и при стратегии 31, предположительно на 3 – 4 порядка. Таким образом, среди предложенных и исследованных стратегий построения ДС наиболее предпочтительна стратегия 31, которая за счет рационального использования имеющейся естествен-

ной дополнительной информации позволяет коренным образом улучшить ситуацию с функциональной безопасностью при наличии недостаточно безопасных составных систем.

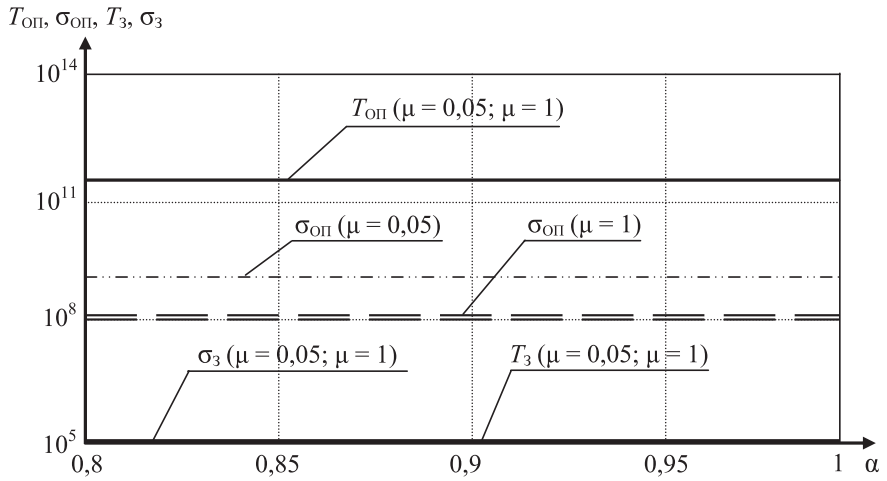


Рис. III.5.21. Результаты исследований эффективности стратегии 32 построения ДС

При этом также существенно уменьшается количество защитных отказов (повышается пропускная способность железной дороги) и повышается надежность ДС по сравнению с ДС, построенными по другим стратегиям.

Следует также отметить, что согласно проведенным исследованиям эффективность стратегий ***31*** и ***32*** построения ДС зависит не только от эффективности встроенных средств обнаружения отказов составных систем, но и (в значительной мере) от системного анализа правильности функционирования этих составных систем.

III.5.8.5. Заключение

Совместное применение различных информационных технологий построения информационных систем управления от-

ветственными и критически важными объектами создает естественные условия для построения двухуровневой системы управления.

В отличие от ранее существующих ограничений на использование в одноуровневых системах управления только безопасных устройств, в двухуровневой системе возможно применение небезопасных систем.

Математическое моделирование трех стратегий построения двухуровневых информационных систем показало, что стратегия ***3*** существенно лучше других стратегий, она позволяет за счет рационального использования имеющейся естественной дополнительной информации коренным образом улучшить ситуацию с безопасностью управления подчиненными объектами при наличии недостаточно безопасных составных систем.

III.5.9. Вопросы для самоконтроля

1. Приведите определения и поясните содержание понятий «функциональная безопасность», «функция безопасности», «полнота безопасности».

2. В чем суть применения мерила «уровней» полноты безопасности» для систем с редкой частотой возникновения опасных отказов?

3. Как зависят уровни полноты безопасности от режимов использования информационной системы и почему достижение третьего и, особенно, четвертого уровня представляет серьезные трудности?

4. Перечислите основные принципы обеспечения функциональной безопасности. Поясните смысл понятия «реактивная отказобезопасность».

5. Раскройте назначение и суть принципа «комбинированная отказобезопасность».

6. Приведите оценки В. Швира и С. Адомита допустимого времени обнаружения одиночного и двойного опасного отказа в двухканальной системе.

7. Приведите оценки допустимого времени обнаружения первого отказа в системах 2x2 и 2v3 с учетом требований по полноте безопасности.

8. Приведите оценки допустимого времени обнаружения двойного отказа в системах 2x2 и 2v3 с учетом требований по полноте безопасности.

9. Приведите и поясните известные вам варианты организации связанных с безопасностью двухканальных информационных систем.

10. Оцените достоинства и недостатки организации двухканальной системы со встроенным контролем работоспособности каналов.

11. Оцените достоинства и недостатки организации двухканальной системы с внешним контролем работоспособности каналов.

12. В чем состоит необходимость и одновременно проблематичность в перезапуске каналов в многоканальной системе?

13. Поясните назначение леммы о вероятности совпадения номеров очередного изымаемого из любой урны шара и ранее изъятого шара. Докажите лемму.

14. Сформулируйте теорему о вероятности возникновения эквивалентных отказов двух идентичных каналов в информационной системе.

15. Раскройте содержание возможных стратегий построения информационных систем с комбинированной двухуровневой организацией функциональной безопасности.

16. Оцените достоинства и недостатки стратегии ***1*** построения информационных систем с комбинированной организацией функциональной безопасности.

17. Оцените достоинства и недостатки стратегии ***2*** построения информационных систем с комбинированной организацией функциональной безопасности.

18. Оцените достоинства и недостатки стратегии ***3*** построения информационных систем с комбинированной организацией функциональной безопасности.

19. Выберите наиболее эффективный вариант построения информационных систем с комбинированной организацией функциональной безопасности.

ГЛАВА III.6. ПОДТВЕРЖДЕНИЕ СООТВЕТСТВИЯ НАДЕЖНОСТИ И БЕЗОПАСНОСТИ ИНФОРМАЦИОННЫХ СИСТЕМ

III.6.1. Основные понятия подтверждения соответствия

Информационная система должна соответствовать техническим требованиям заказчика, если разработка производилась на договорных условиях. Заказчиком может быть любая организационная структура, способная обеспечить финансовые условия разработки и, при необходимости, сбыта продукции. Возможна также инициативная разработка системы, при этом финансовые условия обеспечивает сам разработчик. В этом случае он сам формулирует технические требования к системе, основываясь на потребностях рынка продукции. Разработчик заинтересован создать такую информационную систему, характеристики которой в шкале «цена – качество» пользуются наибольшим спросом, как на рынке продукции, так и на рынке услуг. В обоих случаях созданная информационная система должна удовлетворять предъявленным к ней техническим требованиям. Это относится также и к модифицированной системе. Естественно задаться вопросом: кто определяет соответствие системы предъявленным требованиям? Разработчик полагает, что он создал именно ту систему, которую хотел получить заказчик. Однако у заказчика нет уверенности, что предъявляемая ему система отвечает его требованиям. Нужна доказательственная база того, что данная система именно та, которую хотел получить заказчик. Аналогичная ситуация имеет место и тогда, когда систе-

ма создана без участия заказчика. Теперь на рыночных условиях образуется пара продавец (разработчик) – покупатель (эксплуатирующая организация). Покупатель должен быть уверен, что он не покупает «кота в мешке»; он должен быть уверен в том, что технические характеристики приобретаемой информационной системы соответствуют тем, которые декларирует продавец (разработчик, изготовитель). Решение указанной проблемы состоит в реализации процедур подтверждения соответствия.

Подтверждение соответствия [62] – документальное удостоверение соответствия продукции (информационных систем) или оказания с их помощью услуг требованиям технических регламентов, положений стандартов, сводов правил или условий договоров на всех этапах жизненного цикла.

Подтверждение соответствия состоит из следующих процедур:

– выбор и обоснование схемы подтверждения;

– разработка и согласование перечня показателей информационной системы, по которым будет производиться подтверждение соответствие;

– отбор и идентификация образца продукции;

– разработка и согласование и программы и методики испытаний;

– проведение тестирования и испытаний образца продукции;

– аттестация производства;

– обработка результатов;

– формирование доказательственной базы заявителем с учетом результатов тестирования и испытаний, моделирования, экспертных оценок, конструкторских испытаний и других доказательственных материалов.

Здесь под *идентификацией продукции* понимается установление тождественности характеристик информационной системы ее существенным признакам. З*аявитель* – физическое или юридическое лицо, которое для подтверждения соответствия прини-

мает декларацию о соответствии или обращается за получением сертификата соответствия, получает сертификат соответствия;

Подтверждение соответствия может носить добровольный или обязательный характер. Добровольное подтверждение соответствия осуществляется в форме добровольной сертификации. Обязательное подтверждение соответствия осуществляется в формах: принятия декларации о соответствии (декларирование соответствия); обязательной сертификации.

Добровольное подтверждение соответствия осуществляется по инициативе заявителя для установления соответствия национальным стандартам, предварительным национальным стандартам, стандартам организаций, сводам правил, системам добровольной сертификации, условиям договоров. Объектами добровольного подтверждения соответствия являются информационные системы и их компоненты, программные средства, в отношении которых стандартами, системами добровольной сертификации и договорами устанавливаются требования. Орган по сертификации осуществляет подтверждение соответствия объектов добровольного подтверждения соответствия; выдает *сертификаты соответствия* на объекты, прошедшие добровольную сертификацию; предоставляет заявителям право на применение *знака соответствия*, если применение знака соответствия предусмотрено соответствующей системой добровольной сертификации; приостанавливает или прекращает действие выданных им сертификатов соответствия. Здесь знак соответствия – обозначение, служащее для информирования приобретателей, в том числе потребителей, о соответствии объекта сертификации требованиям системы добровольной сертификации или национальному стандарту.

Обязательное подтверждение соответствия проводится только в случаях, установленных соответствующим техническим регламентом, и исключительно на соответствие требованиям технического регламента. Обязательное подтверждение соответс-

твия обычно распространяется на информационные системы, их программное обеспечение и другие компоненты, если эти системы участвуют в управлении ответственными или критически важными объектами. Объектом обязательного подтверждения соответствия может быть только продукция, выпускаемая в обращение на территории Российской Федерации. Применительно к информационным технологиям применяется обязательное подтверждение соответствия в форме декларирования соответствия.

Декларация о соответствии и ***сертификат соответствия*** имеют равную юридическую силу и действуют на всей территории Российской Федерации в отношении каждой единицы продукции, выпускаемой в обращение на территории Российской Федерации во время действия декларации о соответствии или сертификата соответствия, в течение срока годности или срока службы продукции, установленных в соответствии с законодательством Российской Федерации [62].

Декларирование соответствия осуществляется по одной из следующих схем:

- принятие *декларации о соответствии* на основании собственных доказательств;
- принятие декларации о соответствии на основании собственных доказательств, доказательств, полученных с участием органа по сертификации и (или) аккредитованной испытательной лаборатории (центра) (далее – третья сторона).

При декларировании соответствия заявитель на основании собственных доказательств самостоятельно формирует доказательственные материалы в целях подтверждения соответствия продукции требованиям технического регламента. В качестве доказательственных материалов используются техническая документация, результаты собственных исследований (испытаний) и измерений и (или) другие документы, послужившие основанием для подтверждения соответствия продукции требованиям технического регламента.

Техническая документация должна содержать:

• основные параметры и характеристики продукции, а также ее описание в целях оценки соответствия продукции требованиям технического регламента;

• описание мер по обеспечению безопасности продукции на одной или нескольких стадиях проектирования (включая изыскания), производства, строительства, монтажа, наладки, эксплуатации, хранения, перевозки, реализации и утилизации;

• список документов в области стандартизации, применяемых полностью или частично и включенных в перечень документов в области стандартизации, в результате применения которых на добровольной основе обеспечивается соблюдение требований технического регламента и, если не применялись указанные документы в области стандартизации, описание решений, выбранных для реализации требований технического регламента. В случае если документы в области стандартизации, включенные в перечень документов в области стандартизации, в результате применения которых на добровольной основе обеспечивается соблюдение требований технического регламента, применялись частично, в технической документации указываются применяемые разделы указанных документов.

Техническая документация также может содержать общее описание продукции, конструкторскую и технологическую документацию на продукцию, схемы компонентов, узлов, цепей, описания и пояснения, необходимые для понимания указанных схем, а также результаты выполненных проектных расчетов, проведенного контроля, иные документы, послужившие мотивированным основанием для подтверждения соответствия продукции требованиям технического регламента.

Техническая документация, используемая в качестве доказательственного материала, также может содержать анализ риска применения (использования) продукции. Состав доказательс-

твенных материалов определяется соответствующим техническим регламентом, состав указанной технической документации может уточняться соответствующим техническим регламентом, если отсутствие третьей стороны не приводит к достижению целей подтверждения соответствия.

Декларация о соответствии должна содержать:

- наименование и местонахождение заявителя;
- наименование и местонахождение изготовителя;
- информацию об объекте подтверждения соответствия, позволяющую идентифицировать этот объект.

Декларация соответствия оформляется на русском языке и должна содержать:

- наименование и местонахождение заявителя;
- наименование и местонахождение изготовителя;
- информацию об объекте подтверждения соответствия, позволяющую идентифицировать этот объект;
- наименование технического регламента, на соответствие требованиям которого подтверждается продукция;
- указание на схему декларирования соответствия;
- заявление заявителя о безопасности продукции при ее использовании в соответствии с целевым назначением и принятии заявителем мер по обеспечению соответствия продукции требованиям технических регламентов;
- сведения о проведенных исследованиях (испытаниях) и измерениях, сертификате системы менеджмента качества, а также документах, послуживших основанием для подтверждения соответствия продукции требованиям технических регламентов;
- срок действия декларации о соответствии;
- иные предусмотренные соответствующими техническими регламентами сведения.

Срок действия декларации о соответствии определяется техническим регламентом.

Обязательное подтверждение соответствия путем декларирования соответствия должно быть выполнено и в отношении качества и функциональной безопасности программного обеспечения информационных систем. Так, на железнодорожном транспорте это обстоятельство юридически закреплено Регламентами таможенного союза [89, 90, 91].

Для проверки соответствия требованиям надежности и функциональной безопасности систему предварительно подвергают испытаниям в условиях предприятия-изготовителя для следующих категорий испытаний:

- приемо-сдаточных (ПСИ);
- периодических (ПИ);
- типовых (Т).

При *приемо-сдаточных испытаниях* система подвергается проверке при *сдаче в опытную эксплуатацию*.

Периодические испытания проводят:

- при запуске системы в производство;
- при возобновлении изготовления компонент системы после перерыва более одного года.

*Типовые испытания проводят***:**

- при внесении изменений в схемы, конструкции, технологии изготовления компонентов, замены примененных и покупных изделий.
- при изменении условий и режимов применения, замененных материалов и изделий, на которые повлияли проведенные изменения или по которым выявлено наличие неисправности по рекламации.

Внесение изменений в конструкторскую, программную и технологическую документацию системы производят после получения положительных результатов типовых испытаний.

При положительных результатах типовых испытаний система, изготовленная по измененной документации, предъявляется на испытания в целях подтверждения соответствия.

При отрицательных результатах типовых испытаний предлагаемые изменения не вносят.

Особое место в подтверждении соответствия программных средств занимают испытания на наличие или отсутствие в них недекларированных возможностей (НДВ) (см. п. III.1.7). Эти испытания базируются на анализе исходных текстов программ и нередко затрагивают коммерческие интересы разработчика. Однако вследствие большой ответственности, которую возлагает заказчик на эти испытания, обычно между интересами разработчика и испытательной лабораторией (третьей независимой аккредитованной стороной) находится компромиссное решение, направленное на обеспечение конфиденциальности испытаний исходных текстов программ. Результаты декларирования соответствия совместно с испытаниями на НДВ и аттестацией системы по требованиям государственных органов в части информационной безопасности [88] позволяют оценить состояние киберзащищенности информационной системы.

III.6.2. Проблема натурных испытаний надежности и безопасности информационных систем

Развитие информационных систем характеризуется чрезвычайным усложнением используемого технического оборудования и программного обеспечения. Усложнение аппаратуры влечет за собой усложнение ее проектирования и производства. С другой стороны, уникальность и дороговизна больших систем практически исключает традиционные эмпирические методы проектирования путем «доводки» аппаратуры на серии опытных образцов. В ряде случаев сложную систему не успевают испытать в течение всего периода эксплуатации. Вместе с тем, случайный характер явлений, происходящих в элементах сис-

тем (шумы, перекрестные искажения, отказы, сбои), сложность систем в целом затрудняют аналитические расчеты. Существующие методы анализа надежности и безопасности ориентированы на достаточно простые системы и не могут удовлетворить разработчиков современных сложных информационных систем. Кроме того, аналитический расчет показателей надежности и функциональной безопасности информационных систем осуществляется в условиях различных предположений, которые не позволяют учесть целый ряд реальных условий их функционирования. Поэтому для подтверждения аналитически полученных показателей надежности и безопасности необходимы натурные испытания всей системы в реальных условиях функционирования или условиях, близких к реальным.

Выделим два вида испытаний: *определительные* – испытания, в результате которых определяются числовые значения показателей надежности и/или безопасности, и *контрольные* – испытания, в результате которых устанавливается, что значение показателя надежности (безопасности) не ниже или не выше некоторого значения с заданной вероятностью. Поскольку в результате контрольных испытаний получают меньше информации о надежности изделий, эти испытания менее трудоемки по сравнению с определительными испытаниями.

Задача получения показателей надежности (безопасности) при определительных испытаниях может быть решена путем анализа и накопления результатов выполнения информационной системой предусмотренных задач в определенном временном интервале. При этом собираются и обрабатываются данные на всем пути от текущего ввода данных до выдачи команд управления на объект управления. Например, в информационной системе диспетчерского управления на железнодорожном транспорте по командам поездного диспетчера в течение заданного времени T сформировано и передано N команд телеуправления. Если

на выходе модулей вывода информационной системы N_T команд оказались ошибочными, то вероятность трансформации команды определяется как $P_T = N_T / N$. Аналогично определяются вероятность потери команды и другие вероятностные показатели безопасности системы. Для получения временных характеристик, например, времени до опасного отказа системы, фиксируются интервалы времени t_i между отказами системы и после n отказов вычисляется среднее время до отказа $T_{\text{ОП}} = \dfrac{\sum_{i=1}^{n} t_i}{n}$.

По характеру полученных оценок испытания могут быть организованы и проведены с целью получения усредненных показателей (средняя наработка на отказ, вероятность отказа) и испытаниями с целью получения доверительного интервала для показателя надежности или безопасности, т. е. интервала возможных значений, которым накрывается определяемый показатель с заданной доверительной вероятностью.

По характеру исходных данных определительные испытания могут быть испытаниями, основанными на использовании информации об отказах изделия (прямые испытания), информации о косвенных признаках отказов (например, изменения токов, выходящие за допустимые пределы), либо на накоплении информации об отказах (сборе информации из различных источников).

По условиям проведения испытания могут быть испытаниями в нормальных эксплуатационных режимах и в условиях, ускоряющих процесс возникновения отказов и сбоев. Ускорение испытаний имеет очень большое значение, так как главный недостаток испытаний на надежность и безопасность – их большая продолжительность.

Контрольные испытания служат средством контроля надежности и безопасности по некоторому косвенному при-

знаку. Такими признаками могут быть, например, отсутствие отказов при испытаниях на протяжении заданного времени, предельные числа допустимых или недопустимых отказов в течение некоторой последовательности интервалов времени. В первом случае испытания называются испытаниями, основанными на числе допустимых отказов, равном нулю, а во втором случае – испытаниями, основанными на последовательном анализе.

Широкое распространение на практике получили выборочные контрольные испытания.Одна из возможных процедур выборочных контрольных испытаний на надежность (безопасность) сводится к следующему: выборка размером n изделий из партии размером M ставится на испытания продолжительностью t часов. Фиксируется число появившихся отказов r за время испытаний t и это число отказов сравнивается с заранее назначенным допустимым или приемочным числом отказов R. В результате сравнения делается один из двух возможных выводов: проверяемый показатель надежности или безопасности соответствует заданным требованиям и партия принимается (при $r \le R$) или проверяемый показатель надежности или безопасности не соответствует заданным требованиям и партия бракуется (при $r > R$). Так как исход контрольных испытаний носит случайный характер, то существует определенная вероятность $p(r \le R)$ того, что партия будет принята и вероятность $p(r > R)$ того, что партия будет забракована. При этом возникают ошибки первого и второго родов. Ошибка первого рода (риск поставщика α) имеет место тогда, когда хорошая партия, изделия которой имеют уровень надежности равный или лучше заданного, бракуется по результатам испытаний. Ошибка второго рода (риск потребителя β) имеет место тогда, когда плохая партия, изделия которой имеют уровень надежности ниже заданного, принимается по результатам испытаний. Про-

стейший план проведения испытаний имеет следующий вид. Вводятся два уровня показателя безопасности: приемочный P_0 и браковочный P_1. Тогда план контроля представляет собой пару чисел N и r, которые при заданных α, β, P_0 и P_1 определяются из системы уравнений

$$\sum_{i=0}^{r-1} C_N^i P_0^{N-i} (1-P_0)^i = 1-\alpha,$$

$$\sum_{i=0}^{r-1} C_N^i P_1^{N-i} (1-P_1)^i = 1-\beta,$$

где C_N^i – биномиальный коэффициент, определяемый как $C_N^i = \dfrac{N!}{i!(N-i)!}$.

При малых значениях искомых показателей и при условии сложности и оригинальности испытываемой системы имеют место соотношения $n = N = 1$ и $R = 0$. Это означает, что проверяемая партия состоит из одного изделия, оно же и испытывается.

При проведении натурных испытаний ожидаемые численные значения вероятностных показателей безопасности настолько малы, что практически не представляется возможным в реальных условиях эксплуатации за ограниченное время получить требуемые значения показателей. Так, для получения некоторой вероятности p необходимо провести более чем $1 / p$ испытаний. Например, при оценке системы, связанной с безопасностью, к которой предъявлены требования по третьему уровню полноты безопасности (УПБ 3) с вероятностью опасного отказа в течение часа работы системы на уровне $Q_{\text{ОП}}(1)=10^{-8}$ необходимо число испытаний N задавать, по крайней мере, большим, чем 10^{+8}, т. е. время тестирования должно быть соизмеримо со временем жиз-

ненного цикла системы. Если быть более точными, то дисперсия оценки вероятности равна $(1–Q_{ОП}(1))/(NQ_{ОП}(1))$ или среднеквадратическое отклонение равно $\sqrt{\left(1-Q_{ОП}(1)\right)/\left(NQ_{ОП}(1)\right)}$. Тогда количество опытов определяется как $N=(1–Q_{ОП}(1))/(Q_{ОП}(1)\cdot\sigma^2)$. Среднеквадратическое отклонение неограниченно возрастает с уменьшением искомой вероятности. Это означает, что малую вероятность практически невозможно оценить. Такая же проблема возникает при получении временных характеристик. Например, если считать, что времена до отказов имеют экспоненциальное распределение, то необходимое число отказов определяется исходя из характеристики точности получаемых оценок (среднеквадратическое отклонение $\sigma(T_0)=T_0/\sqrt{N}$) как $N=[T_0/\sigma(T_0)]^2$. Отсюда видно, что при больших значениях средней наработки T_0 до отказа (и, особенно, $T_{ОП}$ до опасного отказа) практически невозможно за ограниченное время получить достоверные результаты испытаний. Контрольные испытания также требуют длительного времени, так как допустимые показатели безопасности имеют малые значения и для их подтверждения требуется длительное время испытаний, соизмеримое с жизненным циклом системы.

III.6.3. Методы ускорения испытаний

Для преодоления проблемы малых вероятностей разработаны методы ускоренных испытаний [92, 93, 96 и др.]. В рамках ускоренных испытаний можно выделить два подхода. Первый – это испытания в условиях, когда используются факторы, ускоряющие процесс возникновения отказов, сбоев, ошибок. Например, повышение температуры, вибрация, влажность и т. п. При этом предварительно должны быть получены зависимости показателей безопасности от изменения ускоряющего фактора в нормальных и форсированных режимах, что являет-

ся задачей не меньшей сложности, чем обычные испытания. Такие зависимости часто имеют характер корреляционных связей, а это означает, что с их помощью можно установить не строго определенное значение показателя надежности, а область его возможных значений. Испытания в форсированных условиях могут привести к разрушению изделия, при котором возникают физико-химические процессы, не характерные для нормальных условий функционирования. Кроме того, использование ускоряющих факторов может не дать значительного ускоряющего эффекта.

Поэтому целесообразен второй подход – использование методов понижения дисперсии, и, в частности, метода существенной выборки [97 – 101, 103 – 105 и др.]. Данный метод, как и другие методы понижения дисперсии, заключаются в искусственном повышении вероятностей ошибок и сбоев путем их генерации и последующим пересчетом на реальный режим функционирования. Методы понижения дисперсии получили значительное распространение в имитационном моделировании систем, когда аналитические расчеты либо затруднительны, либо просто невозможны вследствие сложности анализируемых систем.

Моделирование есть средство изучения системы путем ее замены более удобной для экспериментального исследования системой (моделью), сохраняющей существенные черты оригинала, и испытания модели методом проб. Модель воспроизводит описание с большими или меньшими упрощениями. При этом должен достигаться разумный компромисс между точностью воспроизведения и сложностью необходимых для этого средств. Методы программной имитации случайных процессов реализуют имитационное моделирование систем. При этом случайные воздействия искусственно воспроизводятся программными или физическими датчиками, включен-

ными в общую схему моделирования. Традиционный способ программной имитации случайных функций любой сложности сводится к генерации некоторых стандартных (базовых) процессов. Наиболее употребительным при цифровом моделировании базовым воздействием является последовательность чисел $v_0, \ldots, v_n$, представляющих собой в идеале реализации независимых равномерно распределенных в интервале (0, 1) случайных событий. Фактически в силу ряда причин используется псевдослучайная или квазислучайная последовательность равномерно распределенных чисел, так как она имеет циклический характер. На основе данной последовательности путем некоторых преобразований может быть получена квазислучайная последовательность случайных чисел (дискретных и непрерывных), имеющих любое распределение вероятностей. Так, для генерации непрерывных случайных воздействий наиболее распространенным методом является метод обратной функции, в соответствии с которым случайная величина w, имеющая распределение вероятностей с монотонной функцией F, генерируется из равномерно распределенной случайной величины v по формуле $w=F^{-1}(v)$. Например, случайная величина с экспоненциальным распределением имитируется по формуле $w=-\lambda^{-1}\ln(v/\lambda)$, где λ – интенсивность отказов. Существуют также и другие методы генерации случайных воздействий, включая метод исключения, метод композиции и т. д. Для некоторых распределений, например, нормального распределения вероятностей, используются специальные методы, ориентированные только на данный класс распределений. Так, при генерации нормально распределенных случайных чисел с математическим ожиданием m и среднеквадратическим отклонением σ используется свойство сходимости сумм независимых случайных величин к нормальному распределению, т.е.

$$w = a + \sigma\sqrt{\frac{12}{n}}\left[\sum_{i=1}^{n} v_i - \frac{n}{2}\right],$$

где n – количество реализаций равномерно распределенных в интервале (0, 1) случайных чисел, необходимое для получения одного нормально распределенного числа.

Таким образом, при имитационном моделировании генерируются случайные воздействия на модель системы с заданными законами распределения, в результате действия которых определяются значения случайного выходного параметра или параметров анализируемой системы.

III.6.3.1.Метод Монте-Карло

Пусть задана вероятностная модель функционирования системы с целью нахождения некоторой усредненной характеристики a [93, 94, 95, 102 и др.]. Эта модель есть представление любого показателя безопасности через некоторые случайные величины $Z=f(z_1, \ldots, z_n)$. Тогда $a=Mf(z_1, \ldots, z_n)$. Здесь M – оператор вычисления математического ожидания. Величины $z_1,\ldots,z_n$ имеют реальный смысл: это длительности безотказной работы элементов системы, количество искаженных бит информации в канале и т. п. В качестве функций $f(z_1, \ldots, z_n)$ могут выступать различные функции, зависящие от искомого показателя безопасности. Если определяется вероятность, то $f(z_1, \ldots, z_n) = I_a(z_1, \ldots, z_n)$ – индикаторная функция некоторого события, связанного с показателем a. Если определяется среднее время до отказа, то $f(z_1,...,z_n) = \overline{z}$. Зная распределения величин $z_1,\ldots,z_n$, найденные статистическим путем, и воспроизведя их реализации в точном соответствии с этими распределениями, получим реализацию показателя z. Повторяя опыт всякий раз с независимыми от предыдущих значениями z_i, получаем независимые реализации $Z_1,\ldots,Z_N$ случайной величины Z, по которым находим усредне-

нием оценку величины a. Если Ω – пространство возможных значений случайного вектора $z=(z_1, \ldots, z_n)$, а $p(x)$ – его плотность распределения в этом пространстве, то можно записать

$$a = \int_{\Omega} f(x)p(x)dx,$$

где $x=(x_1,\ldots,x_n)$; dx – элемент объема пространства Ω.

Иначе можно сказать, что $f(Z)$ есть несмещенная оценка параметра a. Если искомый показатель является вероятностью некоторого события, то функция $f(z)$ является индикаторной. В данном случае при проведении моделирования используется следующая формула для расчета параметра a:

$$a = \frac{1}{N}\sum_{i=1}^{N} I_a(z),$$

где N – число опытов или реализаций вектора z, который имеет распределение $p(x)$; $I_a(z)$ – индикаторная функция, принимающая значение 1, если событие, соответствующее показателю a, произошло, и 0 в противном случае.

Таким образом, для вычисления вероятностного показателя a необходимо заданное число раз (N) генерировать случайный вектор z в соответствии с его распределением $p(x)$. При натурных испытаниях вектор z реализуется как объективный результат ошибок, помех, отказов и сбоев. При имитационном моделировании данный вектор генерируется датчиком псевдослучайных чисел с помощью программных или аппаратных средств.

Данный подход является обычным методом имитационного моделирования Монте-Карло и требует большого количества реализаций случайного вектора при малых искомых вероятностях. Для ускорения имитационного моделирования разработан ряд методов, например, модифицированный метод Монте-Карло, метод взвешенного моделирования (значимой выборки), метод

дополняющих переменных и другие. Существуют и другие методы понижения дисперсии, например, метод расслоенной или стратифицированной выборки. Однако, многие из этих методов требуют наличия достаточно детального описания моделируемой системы и ее особенностей, что в ряде случаев затрудняет их применение. Кроме того, их достоинства уступают достоинствам метода значимой выборки, который показал свою эффективность при моделировании многих систем. Основная цель всех этих методов состоит в понижении дисперсии моделируемой на выходе системы случайной величины. Уменьшение разброса характеристик выходного параметра позволит сократить необходимое для заданной достоверности число реализаций случайного вектора и, следовательно, ускорить имитационное моделирование. Рассмотрим суть ускоренного моделирования на примере метода значимой выборки.

III.6.3.2.Метод значимой выборки

Случайная величина Y определяется значениями $Z=f(z_1, \ldots, z_n)$ в области Ω и плотностью $h(x)$. Кроме того, известно распределение $p(x)$. Тогда в силу равенства

$$a = \int_{\Omega} \big(f(x)\, p(x) / h(x) \big) h(x) dx$$

несмещенной оценкой параметра a будет величина $\hat{Z} = f(Y)$ $p(Y)/h(Y)$. Множитель $p(Y)/h(Y)$ называется весом, а данный метод – методом взвешенного моделирования или методом значимой (существенной) выборки. Случайная величина Y может не иметь определенного физического смысла – ее роль состоит в обеспечении нужных статистических свойств оценки. Если искомый показатель является вероятностью некоторого события, то при проведении моделирования используется следующая формула для расчета a:

$$a = \frac{1}{N}\sum_{i=1}^{N} I_a(z)\frac{p(x)}{h(x)}, \qquad (6.1)$$

где вектор z имеет распределение с плотностью $h(x)$.

Дисперсия оценок, получаемых методом взвешенного моделирования, определяется как

$$D = \int_{\Omega} \left(f^2(x)p^2(x)/h^2(x) \right) dx - a^2.$$

Минимум дисперсии достигается при выборе плотности $h(x)$ в виде

$$h(x) = \frac{|f(x)|\,p(x)}{\int_{\Omega} |f(y)|\,p(y)dy}.$$

К сожалению, в общем случае вычисление знаменателя является задачей, ради которой и используется ускоренное моделирование. Кроме того, при натурных испытаниях аналитическое представление функции $f(x)$ неизвестно. Поэтому, будем использовать приближенные методы определения плотности $h(x)$, близкой к оптимальной.

Таким образом, для вычисления вероятностного показателя a (см. (6.1)) необходимо заданное число N раз генерировать случайный вектор z в соответствии с распределением $h(x)$ и каждый раз осуществлять расчет весового коэффициента. Основной смысл введения новой функции $h(x)$ заключается в том, что сбои в устройствах генерируются с большей интенсивностью, а помехи – с большей вероятностью так, чтобы увеличить количество реализаций интересующих нас редких событий.

Следует отметить, что рассмотренные выше методы понижения дисперсии применимы к имитационному моделированию

сложных систем, когда при помощи вычислительных алгоритмов реализуется модель системы. При испытании реальной системы используется не модель, а сама система. Этот факт значительно усложняет использование известных методов и требуется специальный метод ускоренных натурных испытаний, который бы объединял особенности традиционных методов испытаний и методов понижения дисперсии, применяемых в имитационном моделировании.

III.6.4. Метод ускоренных натурных испытаний на надежность и функциональную безопасность информационных систем

III.6.4.1.Теоретические основы метода

Пусть необходимо провести натурные испытания информационной системы управления с целью вычисления или подтверждения некоторой усредненной характеристики ее надежности или безопасности a. Также, как и при имитационном моделировании, данный показатель безопасности выражается через некоторые случайные величины $Z=f(z_1, \ldots, z_n)$ и $a=Mf(z_1, \ldots, z_n)$. В данном случае функция $f(z_1, \ldots, z_n)$ неизвестна и не может быть представлена в виде реализуемой программно модели. Однако ее свойства аналогичны свойствам, рассмотренным выше для имитационной модели. Применительно к информационным системам управления функция $f(z_1, \ldots, z_n)$ есть результат прохождения команды через все элементы системы на ее выход. В процессе испытаний образуются независимые реализации $z_1,\ldots,z_n$ случайной величины z. Если Ω – пространство возможных значений случайного вектора $Z=f(z_1, \ldots, z_n)$, а $p(x)$ – его плотность распределения в этом пространстве, то можно записать

$$a = \int_{\Omega} f(x)\,p(x)dx,$$

где $x=(x_1, \ldots, x_n)$; dx – элемент объема пространства Ω.

При реализации ускоренных испытаний на основе метода значимой выборки нельзя заменить вектор случайных величин $z=(z_1, \ldots, z_n)$ на вектор с новой плотностью $h(x)$, как это осуществляется в имитационном моделировании. Поэтому предлагается использовать искусственное внесение случайных ошибок и искажений $y=(y_1, \ldots, y_n)$ с некоторой плотностью $g(x)$. Тогда суммарный вектор ошибок и искажений будет являться композицией исходных векторов z и y, т. е. $s=z\otimes y$, где вектор s будет иметь некоторую плотность $h(x)=\phi(p(x), g(x))$. Здесь φ – функция комбинирования двух исходных плотностей.

Таким образом, внося искусственно ошибки и искажения с функцией плотности $q(x)$, в действительности получаем общую функцию плотности $h(x)$. Отсюда

$$a = \int_{\Omega} (f(x)\frac{p(x)}{h(x)})h(x)dx = \int_{\Omega} (f(x)\frac{p(x)}{\phi(p(x), g(x)}))\phi(p(x), g(x))dx$$

и несмещенной оценкой параметра a будет величина $Z = f(Y)\dfrac{p(Y)}{\phi(p(Y), g(Y))}$.

Если искомый показатель является вероятностью некоторого события, то при проведении моделирования используется следующая формула для расчета a:

$$a = \frac{1}{N}\sum_{i=1}^{N} I_a(z)\frac{p(x)}{\phi(p(x), g(x))}, \qquad (6.2)$$

где вектор z имеет распределение $\phi(p(x), q(x))$.

Дисперсия оценок, получаемых методом взвешенного моделирования, определяется как

$$D = \int_{\Omega} (f^2(x) \frac{p^2(x)}{\phi^2(p(x), g(x)}) dx - a^2.$$

Минимум дисперсии достигается при выборе плотности $h(x)$ в виде

$$\phi(p(x), g(x)) = \frac{|f(x)| p(x)}{\int_a |f(y)| p(y) dy}$$

и при выборе $q(x)$ следующим образом:

$$g(x) = \phi^{-1} \left[\frac{|f(x)| p(x)}{\int_{\Omega} |f(y)| p(y) dy} \right].$$

Как уже отмечалось, вычисление последнего выражения практически невозможно. Поэтому функцию $g(x)$ следует выбирать исходя из структуры системы и вероятностных характеристик ее элементов. Для систем с высоким уровнем безопасности функция $g(x)$ должна выбираться из условия, согласно которому основной вес $g(x)$ сосредоточен на множестве точек, для которых плотность $p(x)$ имеет малые значения. Формально это можно записать как

$$\max(g(x), p(x)) \geq \varepsilon.$$

Тогда для этих точек можно считать, что с заданной точностью ε выполняется равенство $h(x) = g(x)$. Очевидно, что в действительности для любой точки справедливо неравенство $h(x) > g(x)$. Тогда

$$a = \int_{\Omega} (f(x) \frac{p(x)}{\phi(p(x), g(x))} \phi(p(x), g(x)) dx \leq$$

$$\leq \int_{\Omega} (f(x) \frac{p(x)}{g(x)} \phi(p(x), g(x)) dx = a^{\wedge},$$

где $a^{\wedge}$ – обозначение верхней границы показателя.

Другими словами, в результате испытаний вычисляется верхняя граница для показателя безопасности, т. е. истинная граница гарантированно подтверждается.

Очевидно, что не для всех элементов системы возможно искусственное внесение сбоев и ошибок. Пусть система состоит из m подсистем, в k из которых реализованы искусственные ошибки и сбои. Каждая из подсистем характеризуется некоторой плотностью случайных сбоев и ошибок $p_i(x)$. Тогда можно записать

$$a = \int_{\Omega} (f(x) \frac{p(x)}{g(x)} \phi(p(x), g(x)) dx =$$

$$= \int_{\Omega} (f(x) \frac{\prod_{i=1}^{n} p_i(x)}{\prod_{i=1}^{n} g_i(x)}) \cdot \phi(p(x), g(x)) dx =$$

$$= \int_{\Omega} \left(f(X) \frac{\prod_{i=1}^{n} p_i(x)}{\prod_{i=1}^{n} g_i(x)} \right) \phi(p(x), g(x)) dx$$

в предположении, что для всех $i > k$ имеет место равенство $p_i(x) = g_i(x)$. Другими словами, естественный поток ошибок и сбоев в оставшихся $n - k$ подсистемах является основным. Если этот поток повлияет на итоговую оценку, то, следовательно, он является преобладающим и показатель безопасности не опреде-

ляется сбоями в первых k подсистемах. Если этого не произойдет, то показатель безопасности определяется только внесенными сбоями.

Таким образом, метод ускоренного имитационного моделирования может быть использован для проведения ускоренных натурных испытаний. При этом получается пессимистическая оценка искомого показателя безопасности. Следует отметить, что для реальных высоконадежных систем соотношение плотностей $p(x)$ и $g(x)$ таково, что $a^{\wedge} \to a$, т. е. в области ненулевых значений функции $g(x)$ стремится к нулю значение вероятности $p(x) \to 0$, иначе ускорение при испытаниях не имело бы места.

III.6.4.2. Практическое приложение метода к ускоренным испытаниям информационных систем управления технологическими процессами

Любая информационная система управления технологическими процессами состоит из некоторого множества цифровых устройств (микропроцессоров, устройств памяти, контроллеров, мультиплексоров и др.) и совокупности каналов связи. Пусть количество устройств в системе равно n, а количество каналов – k. Командно-информационный поток данных в виде кадров данных циркулирует в аппаратно-информационной структуре в соответствии с заданной программой. Обозначим реальные вероятности сбойной ошибки на одной операции в каждом i-м устройстве как g_i ($i = 1 \ldots n$). Для внесения искусственных сбоев в каждое устройство создается программа-генератор сбойных ошибок (τ-генератор), генерирующая через случайные промежутки времени в соответствии с заданным распределением сбои при помощи датчика случайных чисел. Программа выполняется в фоновом режиме. Обозначим вероятности искусственно вводимой сбойной ошибки на одной операции в каждом устройстве при воздействии τ-генераторов как q_i ($i = 1 \ldots n$).

Пусть времена обработки кадра данных на каждом устройстве равны t_i, где i – номер устройства. При обработке данных на некотором устройстве выполняется заданная последовательность операций, на каждой из которых может произойти сбойная ошибка. Для большинства микропроцессорных систем количество m операций, которые необходимо выполнить до возникновения сбойной ошибки, имеет геометрическое распределение $G(m)=g(1-g)^{m-1}$. Действительно, вероятность того, что искажается первая операция, равна g; вероятность искажения второй операции определяется из условия, что первая прошла успешно и равна $g(1-g)$; вероятность искажения третьей операции определяется из условия, что первые две операции прошли успешно: $g(1-g)^2$. Продолжая данные рассуждения, получим геометрическое распределение числа операций до сбойной ошибки.

Математическое ожидание числа операций до сбойной ошибки равно $1/g$. Если среднее время выполнения одной операции равно Δt, а время обработки кадра данных на i-м устройстве – t_i, то среднее число n_i операций на один кадр данных ТУ или ТС $n_i=t_i/\Delta t$. Среднее время до сбойной ошибки равно $\Delta t/g$.

После начала обработки кадра на i-м устройстве до сбойной ошибки на m-ой операции проходит среднее время, равное $m\Delta t$. Вероятности того, что через время τ после начала обработки кадра данных произойдет реальная и искусственная сбойная ошибка равны соответственно

$$G(\tau_i) = g_i(1-g_i)^{\tau_i/\Delta t};\ H(\tau_i) = q_i(1-q_i)^{\tau_i/\Delta t}.$$

Таким образом, вероятности того, что в устройстве i ($i = 1\ldots m$), на каждом кадре обрабатываемых данных был сбой в момент времени τ_i ($i = 1\ldots m$), равны соответственно

$$G(\tau_1,...,\tau_m) = \prod_{i=1}^{m} G(\tau_i);\ H(\tau_1,...,\tau_m) = \prod_{i=1}^{m} H(\tau_i).$$

Геометрический закон достаточно точно можно аппроксимировать экспоненциальным распределением с математическим ожиданием $\Delta t/g$ или $\Delta t/q$. Поэтому непрерывный поток сбойных ошибок можно рассматривать как пуассоновский поток. Тогда вероятность того, что за время t_k прохождения кадра в i-м устройстве было w_i сбойных ошибок, определяется как

$$G(w_i) = \frac{e^{-\lambda_i t_k}(\lambda_i t_k)^{w_i}}{w_i!};\ H(w_i) = \frac{e^{-\rho_i t_k}(\rho_i t_k)^{w_i}}{w_i!},\ \text{где } \lambda_i = g_i/\Delta t;\ p_i = q_i/\Delta t.$$

Следует отметить, что в результате одного сбоя может появиться не только один искаженный бит кадра данных, а целая серия битов или даже весь кадр в целом. Количество сбоев архивируется в памяти.

Информация проходит как по устройствам, так и по трактам передачи данных между этими устройствами, действие помех в которых также необходимо учитывать. Обозначим реальные вероятности ошибки на бит в i-м информационном потоке p_i, а количество информационных потоков l. Для внесения искусственных помех в поток, в каждом устройстве на выходе потока записывается программа-генератор ошибок в потоке (z-генератор), генерирующая случайное время до появления ошибки. Так как ошибки независимы, то их количество на длине кадра данных может формироваться в соответствии с биномиальным распределением при помощи датчика случайных чисел. Однако, с точки зрения практической реализации механизма внесения ошибок в поток, необходимо перейти к непрерывному времени и осуществлять генерацию ошибок через случайные интервалы времени. Если число ошибок на длине кадра данных определяется биномиальным распределением, то случайное количество битов до очередной ошибки имеет геометрическое распределение. Переходя к непрерывному времени, данное распределение может быть аппроксимировано экспоненциаль-

ным распределением с тем же средним. В итоге, поток ошибок в трактах передачи данных является пуассоновским и число ошибок за заданный интервал времени определяется распределением Пуассона.

Обозначим вероятности искусственно вводимых ошибок в i-м потоке как h_i. Тогда вероятность возникновения m_i ошибок на длине кадра данных n_i в i-м потоке определяется для реальных и искусственных векторов ошибок как:

$$G(m_i, n_i) = C_{n_i}^{m_i} p_i^{m_i} (1 - p_i)^{n_i - m_i},\ H(m_i, n_i) = C_{n_i}^{m_i} h_i^{m_i} (1 - h_i)^{n_i - m_i}$$

где $C_{n_i}^{m_i}$ – биномиальный коэффициент, определяемый как $C_{n_i}^{m_i} = \dfrac{n_i!}{m_i!(n_i - m_i)!}$.

Обозначим скорость передачи данных в i-м потоке как V_i, тогда длительность передачи одного бита равна $1/V_i$. Так как среднее число битов до ошибки в i-м потоке равно $1/h_i$, то среднее время до ошибки равно $1/(h_i V_i)$. Отсюда вероятность того, что за некоторое время t_k в i-м потоке произошло z_i ошибок, равна

$$G(z_i) = \frac{e^{-t_k p_i V_i} (t_k p_i V_i)^z}{z!};\ H(z_i) = \frac{e^{-t_k h_i V_i} (t_k h_i V_i)^z}{z!}.$$

Вероятности того, что в трактах передачи в процессе выполнения обработки одного кадра данных имеет место $z_1,\ldots,z_l$ ошибок, определяется для реальных и искусственных векторов ошибок как

$$G(z_1,...,z_l) = \prod_{i=1}^{l} G(\tau_i);\ H(\tau_1,...,\tau_l) = \prod_{i=1}^{l} H(\tau_i).$$

В итоге, вероятности того, что в устройствах произошло $w_0,\ldots,w_k$ сбойных ошибок, а в трактах передачи данных – $z_1,\ldots,z_k$ ошибок за время t_k реакции системы определяется как

$$G(W_i, Z_i) = G(w_0, ..., w_k)G(z_1, ..., z_l);$$

$$H(W_i, Z_i) = H(w_0, ..., w_k)H(z_1, ..., z_l),$$

где W_i, Z_i – реализации векторов $w_0,\ldots,w_k$, $z_1,\ldots,z_l$ на i-м прогоне команды.

Тогда выражение для вычисления показателя безопасности методом ускоренных испытаний имеет следующий вид:

$$a = \frac{1}{N}\sum_{i=1}^{N} I_a(W_i, Z_i)\frac{G(W_i, Z_i)}{H(W_i, Z_i)} =$$

$$= \frac{1}{N}\sum_{i=1}^{N} I_a(W_i, Z_i)\frac{\prod_{j=1}^{k} e^{-t_k g_j/\Delta t} g_j^{w_i} \prod_{j=1}^{N} e^{-t_k p_j V_i} p_j^{z_i}}{\prod_{j=1}^{k} e^{-t_k q_j/\Delta t} q_i^{w_i} \prod_{j=1}^{l} e^{-t_k h_j V_i} h_j^{z_i}}, \qquad (6.2)$$

где $I_a(W_i, Z_i)$ – индикаторная функция, принимающая значение 1, если событие, соответствующее показателю a, произошло при реализации векторов W_i, Z_i, и 0 в противном случае; N – число выданных команд в процессе эксперимента, k – число устройств, l – число каналов передачи данных.

Например, если a – вероятность трансформации команды телеуправления, то функция $I_a(W_i, Z_i)$ равна 1, если в результате генерации ошибок и сбоев на i-м прогоне команды управления на выходе системы произошла трансформация команды в некоторую другую разрешенную команду, иначе функция $I_a(W_i, Z_i)$ равна 0. Таким образом, функция $I_a(W_i, Z_i)$ в формуле (6.2) не вычисляется, а определяется в результате натурных испытаний. Если a – временной показатель безопасности, например среднее время до ложного срабатывания средства напольного оборудования, то $I_a(W_i, Z_i)$ принимает значения интервалов времени между ложными срабатываниями.

Для повышения эффективности представленного подхода целесообразно использовать также *метод дополняющих переменных*. При практической реализации данного метода генерация искусственных ошибок и сбоев, осуществляемая при помощи преобразования датчика случайных чисел, распределенных равномерно в интервале (0, 1), выполняется чередованием генерируемого случайного числа v и числа $1 - v$. Монотонность функции $Z=f(z_1,\ldots,z_n)$, необходимая для реализации метода дополняющих переменных, следует из того факта, что чем больше интенсивность ошибок или сбоев в элементах системы, тем больше соответственно количество ситуаций трансформации команд на выходе системы.

Для использования разработанного метода ускоренных натурных испытаний необходимо решить две основные задачи.

Первая задача – выбор значений реальных вероятностей ошибки в результате сбоев на одну операцию в устройствах с номерами 1, …, k и вероятностей ошибок в потоках или каналах передачи данных с номерами 1, …, l.

Вторая задача – определение интенсивностей искусственных ошибок и сбоев на одну операцию в устройствах 1, …, k и вероятностей искусственных ошибок в потоках или каналах передачи данных с номерами 1, …, l.

Для решения первой задачи предлагается использовать следующие два пути, выбор одного из которых определяется наличием исходной информации о надежности и безопасности всех элементов системы:

1. Использование уже имеющихся данных или сведений о характеристиках безопасности и надежности элементов системы, а также о характеристиках помехоустойчивости каналов связи. Источником таких данных могут быть документация на соответствующее устройство, характеристики функционирующих и эксплуатируемых аналогов, оценки

независимых экспертов, данные разработчиков элементной базы и аналитические расчеты для конкретного устройства. Однако, как показывает опыт, приведенные в документации характеристики надежности и безопасности могут достаточно сильно расходиться с реальными их характеристиками. Поэтому для подтверждения данных по устройствам и каналам, а также для их получения (при отсутствии) на начальном этапе предлагается также использовать натурные испытания каналов и устройств.

2. Заметим, что основная часть устройств имеет вероятности сбоев или ошибок значительно более высокие, чем соответствующие вероятности для системы. Поэтому время, необходимое для получения данных вероятностей для отдельных устройств, намного меньше, чем время испытаний системы. Это делает возможным проведение обычных натурных испытаний по известным и широко применяемым методикам.

Следует отметить, что первая задача во многом определяет и вид многих математических выражений для реализации метода ускоренных натурных испытаний. При расчете весовых коэффициентов для каждого элемента системы были использованы предположения о независимости сбоев цифровых устройств и ошибок в каналах передачи информации, геометрический закон распределения числа операций до сбоя и т. п.), которые могут быть изменены в результате предварительного анализа функционирования элемента. Например, если известно, что канал передачи данных характеризуется наличием пакетов ошибок и может быть описан Марковскими моделями, то соответствующие выражения также изменятся.

Таким образом, испытания проводятся в два этапа:

1. Определение вероятностей ошибок в результате сбоев отдельных компонент системы и вероятностей ошибок на бит в потоках передачи данных на основе априорной информации.

2. Определение показателей безопасности методами ускоренного натурного моделирования и введением искусственных сбойных ошибок в компонентах системы и ошибок в потоках передачи данных.

Следует отметить, что описанная инженерная методика может быть использована для любого уровня детализации составных элементов информационной системы управления технологическими процессами, что позволяет определять характеристики надежности системы и безопасности управления на промежуточных этапах обработки данных и передачи команд. При этом, их определение не требует дополнительных испытаний, а может проводиться в рамках испытаний всей системы. Для этого необходима архивация результатов прохождения кадров данных каждого из этапов с соответствующим вычислением весовых коэффициентов. В данном случае для любого множества Q мощности l потоков передачи данных и множества R мощности k компонент системы показатель безопасности a вычисляется по следующей формуле:

$$A = \frac{1}{N}\sum_{i=1}^{N}\left(I_a(W_i, Z_i)\frac{\prod_{j=1}^{k} e^{-t_k g_j/\Delta t} {g_j}^{w_i} \prod_{j=1}^{l} e^{-t_k p_j V_i} {p_j}^{z_i}}{\prod_{j=1}^{k} e^{-t_k q_j/\Delta t} {q_j}^{w_i} \prod_{j=1}^{l} e^{-t_k h_j V_i} {h_j}^{z_i}} \right) \qquad (6.3)$$

III.6.4.3. Оценка продолжительности испытаний

Определим количество опытов, необходимое для подтверждения вероятностного показателя безопасности A. Чтобы показать, что $a \le A$, в обычных условиях моделирования необходимо выполнить, по крайней мере, $N \sim 1/A$ прогонов команды. Если быть более корректным, то для любых значений A неравенство Чебышева определяет объем выборки $N \ge 1/(4\Delta_A(1-P))$, где Δ_A – доверительный интервал (погреш-

ность) вычисления вероятности A при заданной доверительной вероятности P. Если выбрать доверительный интервал равным $\Delta_A=A$, а $P = 0{,}95$, то необходимый объем выборки равен $N\geq 5/A$. Заметим, что если подобрать соответствующим образом вероятности q_i и h_i внесенных ошибок и сбоев, то это значение можно уменьшить.

Если в результате N прогонов команд управления контролируемое событие не происходит, то соответствующий этому событию показатель безопасности не превышает значения A, то есть, A является его верхней границей для истинного показателя безопасности.

Пусть N^* – допустимое число испытаний. Тогда $N^*\approx N\Phi$, где Ф – «среднее» значение весового коэффициента. Отсюда $N^*=5\Phi/A$ и $\Phi=N^*A/5$. Оценим приближенно значение Ф. Среднее значение случайной величины w_i для пуассоновского процесса равно $t_k q_i/\Delta t$, а среднее значение случайной величины z_i равно $1/h_i V_i$.

Пусть $q=q_i$, $g=g_i$, $h=h_i$, $p=p_i$, $V=V_i$. Тогда

$$\Phi = \frac{g^{kt_k q/\Delta t} p^{lt_k V}}{q^{kt_k q/\Delta t} h^{lt_k V}}.$$

Предполагая, что каналы связи и устройства вносят примерно одинаковый вклад в искажение кадра данных, получаем уравнения с неизвестными q и h:

$$\left(AN^* 4(1-P)\right)^{1/2} = \frac{g^{kt_k q/\Delta t}}{q^{kt_k q/\Delta t}};\ \left(AN^* 4(1-P)\right)^{1/2} = \frac{p^{lt_k V}}{h^{lt_k V}}.$$

Здесь значения A, g, p, k, l, P, N^*, V, t_k и Δt известны. Данные уравнения могут быть решены методом половинного деления отрезка (0, 1).

Определив значения q и h, проводим испытания с числом прогонов команд N^*.

III.6.5. Пример ускоренных натурных испытаний на функциональную безопасность информационной системы управления технологическим процессом – системы диспетчерской централизации на железнодорожном транспорте

III.6.5.1. Описание объекта испытаний

Объектом испытаний является система диспетчерской централизации и диспетчерского контроля (ДЦ-ДК) в составе следующих взаимосвязанных подсистем: 1. Подсистема пункта управления (ПУ); 2. Подсистемы пунктов контроля (КП); 3. Распределенная коммуникационная подсистема.

Данная система ДЦ-ДК выполняет следующие **основные функции**: передача команд телеуправления (ТУ) с ПУ на КП; передача, прием и исполнение команд ТУ на пунктах контроля; автоматическое ведение графика исполненного движения, местное управление на пунктах контроля, сбор информации о поездном положении с контролируемых участков, передача команд телесигнализации (ТС) на пункт управления, тестирование исправности всей системы с глубиной до модулей аппаратных средств, архивирование результатов работы всех узлов системы и др.

Состав системы ДЦ – ДК:

• ***Подсистема ПУ***: программное обеспечение общего и специального назначения, автоматизированные рабочие места поездных диспетчеров, электромеханика, диспетчеров других служб, сервер сети и сетевое оборудование, программно – технические шлюзы системы и др.;

• ***Подсистема КП***: комплекс технических средств (КТС) «Тракт – ЛП» в составе (см. рис. III.6.1) центрального вычислителя (модуль МАСТЕР), модуля вывода дискретной информа-

ции (МВЫВ), модуля ввода дискретной информации (МВВОД), модуля вывода ответственных команд (МВОК), модулей сопряжения с объектами управления и контроля, общее и специальное программное обеспечение, помехоустойчивая локальная сеть на базе CAN – магистрали и др.;

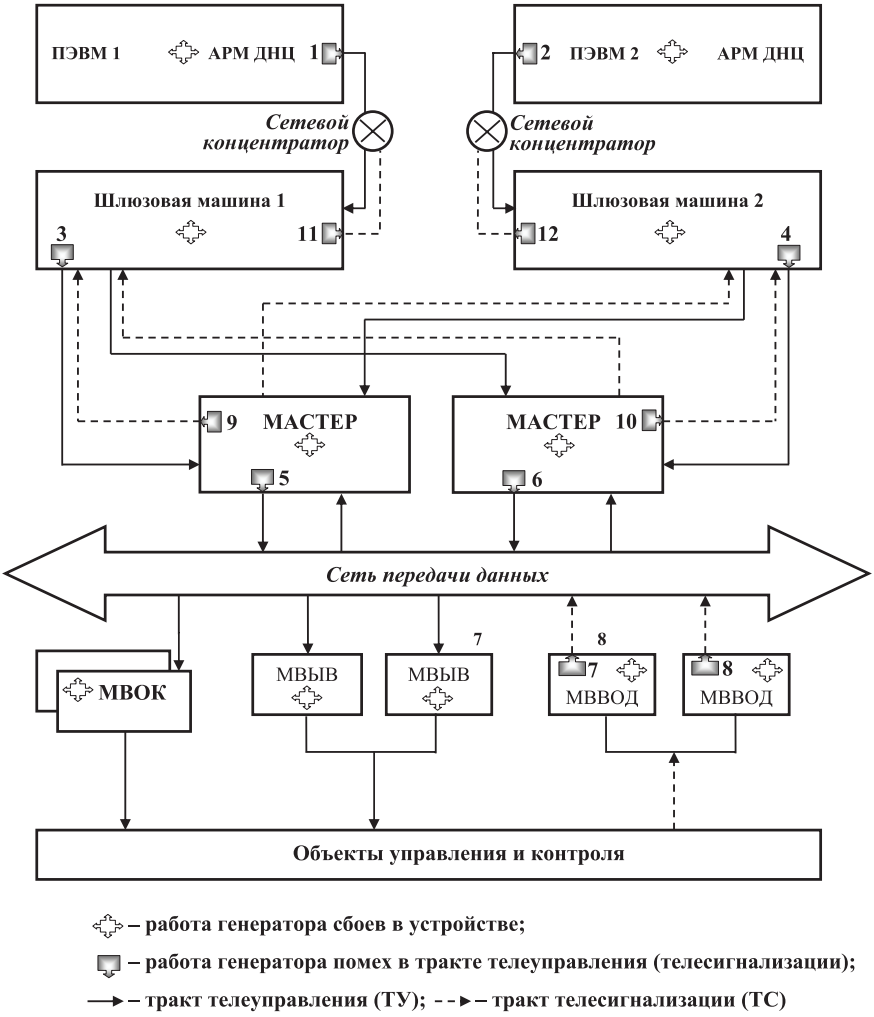


Рис. III.6.1. Структура системы диспетчерской централизации и диспетчерского контроля, представленной на ускоренные натурные испытания

• ***Коммуникационная подсистема*** обеспечивает работу с различными типами каналов связи, включая оптоволоконные.

На ускоренные натурные испытания представлена система ДЦ-ДК в упрощенном варианте: для обеспечения функциональной безопасности в подсистеме пункта управления сформированы два независимых канала в составе двух автоматизированных рабочих мест (АРМов) ДНЦ (поездных диспетчеров) и двух шлюзовых машин. В подсистеме пункта контроля (КП) для обеспечения надежности задублирован модуль МАСТЕР и модули вывода ответственных команд (МВОК), модули ввода и вывода.

Программное обеспечение комплекса технических средств «Тракт – ЛП» состоит из: технологического программного обеспечения (ПО), инструментального ПО, системного ПО, объектного ПО, прикладного ПО. *Технологическое ПО* предназначено для обеспечения загрузки программ и модулей, тестирования и отладки на этапе производства, во время проведения пуско-наладочных и ремонтных работ. *Инструментальное ПО* предназначено для разработки и корректировки прикладного и системного ПО модуля МАСТЕР и функциональных модулей. *Системное ПО* предназначено для обеспечения безопасного и надежного функционирования прикладного программного обеспечения. *Объектное ПО* отражает особенности каждой конкретной железнодорожной станции диспетчерского участка и позволяет выполнять такие функции, как исключение враждебных маршрутов, разделение команд во времени, регистрацию накопления маршрутов и др. *Прикладное ПО* состоит из программных модулей связи с верхним уровнем, с периферийными контроллерами и программного модуля анализа команд верхнего уровня.

На рис. III.6.1 знаком  показаны генераторы помех. Они привязаны к каналам передачи информации и включены в АРМы ДНЦ (под номерами 1 и 2), в шлюзовые машины (под номерами

3 и 4, 11 и 12), в модули МАСТЕР (под номерами 5 и 6, 9 и 10), а также в модули ввода (под номерами 7 и 8). Их общее количество в объекте испытаний – 12.

Знаком ✣ на схеме рис. III.6.1 обозначены генераторы сбоев. Их общее количество в объекте испытаний – 11.

III.6.5.2.Цель испытаний и критерии отказов

Цель испытаний состоит в оценке соответствия показателей безопасности системы ДЦ-ДК «Тракт» требованиям, установленным в нормативно-технической документации (см. табл. III.6.1). Испытания контрольные. *Систему считают выдержавшей испытания* на функциональную безопасность, если нормированные значения показателей, полученные после обработки результатов, *не выходят за пределы нижней границы.*

Нормируемые показатели функциональной безопасности системы ДЦ-ДК «Тракт» приведены в табл. III.6.3.

При неудовлетворительных результатах испытаний они повторяются в полном объеме с заменой оборудования. Если в процессе испытаний с замененным оборудованием получены *неудовлетворительные результаты*, система считается не выдержавшей испытания.

***Критерий отказа системы**:*

1. Трансформация сигнала ТУ или ТС.
2. Потеря сигнала ТУ или ТС.

Под трансформацией сигнала ТУ или ТС следует считать получение сигнала ТС от модуля ввода (МВВОД) по несоответствующему выданной команде каналу управления в период реализации цикла команды ТУ или между ними. При этом решение о месте трансформации сигнала (в тракте ТУ или ТС) принимается после обработки архивов информации в системе.

Потерей сигнала ТУ или ТС следует считать отсутствие отклика системы по каналам ТС на выдачу команды ТУ. При этом

решение о принадлежности сигнала (тракту ТУ или ТС) принимается после обработки архивов информации в системе.

Критерий опасного отказа систем

Опасным отказом системы считается получение сигнала от модуля ввода о срабатывании контрольного выходного узла системы:

- по любому из каналов управления модуля вывода ответственных команд (МВОК) в период времени между циклами выдачи команд ТУ;
- по несоответствующему выданной ответственной команде (ОК) каналу управления МВОК в период реализации цикла команды ТУ;
- при реализации системой не ответственной команды.

III.6.5.3. Алгоритмы генерации сбоев и помех

Общие сведения

Генерация сбоев и помех осуществлялась программно с использованием стандартного датчика случайных чисел ς, равномерно распределенных в интервале от 0 до 1.

Генерация сбойных ошибок в *i*-м устройстве выполнялась путем моделирования случайных чисел, имеющих экспоненциальное распределение с интенсивностью $\rho_i = q_i/\Delta t$. Подпрограмма на языке С в общем случае имеет следующий вид:

```
  double pokazat1(float q, float dt, int m, int*e)
  {
  if ( q/dt!=0) {
  for (int i=1; i<=m; i++) if(ravnom_01()>0,5)
e[i]=1;else e[i]=0;
  return ( -log(ravnom_01()*dt/q;
  } else return (0);
  }
```

Параметры подпрограммы:

q – вероятность сбойной ошибки,

dt – длительность исполнения одного такта ведущего процессора устройства;

m – длина искажаемого результата;

e – вектор ошибки, искажающей результат.

Подпрограмма формирует случайные числа τ_1, τ_2, …, τ_j, …, равные моментам наступления сбойных ошибок. Абсолютное системное время реализации числа τj архивируется в каждом устройстве 1…12 системы ускоренных натурных испытаний (см. рис. III.6.1). Вектор ошибок e складывается по mod 2 с искажаемым результатом.

Генерация ошибок вследствие помех в *i*-м потоке передачи данных осуществляется моделированием случайных чисел, имеющих экспоненциальное распределение со средним временем до ошибки равным $1/h_iV_i$. Подпрограмма на языке С в общем случае имеет следующий вид:

```
double pokazat2(float h, float v, int m, int*e)
{
if ( h*v!=0) {
for (int i=1; i<=m; i++) e[i]=0;
for (int i=1; i<=m; i++) if(ravnom_01()>1/m) {
e[i]=1; break;
}
return ( -log(ravnom_01()/h/v);
} else return (0);
}
```

Параметры подпрограммы:

h – вероятность ошибки на один бит в потоке,

v – скорость передачи данных в *i*-м потоке передачи данных;

m – длина искажаемого результата;

e – вектор ошибки, искажающей результат.

Подпрограмма формирует случайные числа z_1, z_2, …, z_j, …, равные моментам наступления ошибок в потоке передаваемых данных. Абсолютное системное время реализации числа z_j архивируется в восьми устройствах системы ускоренных натурных испытаний (см. рис. III.6.1) для z-генератора. Вектор ошибок **e** складывается по mod 2 с искажаемым результатом.

Генераторы модулей МАСТЕР комплекса технических средств «Тракт»

Генераторы G5, G6 формируют ошибки в потоке данных локальной сети сигналов ТУ, генераторы G9, G10 – в каналах связи со шлюзовыми машинами (телесигнализация – ТС). Кроме того, в модулях МАСТЕР основного и резервного комплекта работают генераторы сбоев.

Генераторы сбоев и помех для модулей ввода и вывода команд

Генератор сбоев для модуля МВОК работает по прерыванию от независимого таймера следующим образом. При наступлении события τ_j формируется вектор ошибки, время в виде кода записывается в таймер. По истечении времени τ_j по прерыванию от таймера на выходной код накладывается по mod 2 вектор ошибки. Абсолютное системное время наложения ошибки записывается в архив данных оперативной памяти модуля МАСТЕР. Считывание архивов данных осуществляется через инструментальный канал в сервисный компьютер специальной утилитой.

Генераторы G7, G8 формируют ошибки в потоке данных локальной сети (ТС). Кроме того, в модулях МВВОД и МВЫВ основного и резервного комплектов работают генераторы сбоев.

Генераторы шлюзовой машины

Генераторы G3, G4 формируют ошибки в потоке данных тракта ТУ канала связи с линейным постом, генераторы G11, G12

формируют ошибки в потоке данных тракта ТС сети Ethernet (шлюзовая машина – АРМ ДНЦ), доступный через системный протокол SMB (Server Message Block). Кроме того, в шлюзовых машинах основного и резервного комплектов работают генераторы сбоев.

Генераторы персональных компьютеров поездных диспетчеров

Генераторы G1, G2 формируют ошибки в потоке данных тракта ТУ сети Ethernet (АРМ ДНЦ – шлюзовая машина). Кроме того, в персональных компьютерах АРМ ДНЦ основного и резервного комплектов работают генераторы сбоев.

Функции, выполняемые генераторами сбоев и генераторами помех:

- формирование вектора ошибки, который моделирует сбой устройства;
- формирование последовательности сбоев при длительности выполнения одного такта ведущего процессора: в модуле МАСТЕР $dt = 40$ нс, в модулях ввода (МВВОД) и вывода (МВОК, МВЫВ) $dt = 62{,}5$ нс, в шлюзовой машине и в персональном компьютере ДНЦ $dt = 5$ нс;
- формирование вектора ошибок для потока передачи данных;
- формирование последовательности ошибок в потоках данных:

– сети CAN линейного поста: скорость передачи данных $v = 10^5$ бит/с.; длина битового представления кадра CAN $m = 64$; длина вектора ошибки 8 байтов;

– каналов связи с шлюзовыми машинами: скорость передачи данных $v = 57600$ бит/с, длина битового представления кадра CAN $m = 8$; длина вектора ошибки 1 байт;

– локальной сети линейного поста: скорость передачи данных $v = 10^5$ бит/с; длина битового представления кадра CAN $m = 32$; длина вектора ошибки 4 байта;

– тракта ТС сети Ethernet: скорость передачи данных $v = 10^7$ бит/с.; длина битового представления кадра CAN $m = 384$; длина вектора ошибки 48 байтов;

– тракта ТУ сети Ethernet: скорость передачи данных $v = 10^7$ бит/с.; длина битового представления кадра CAN $m = 176$; длина вектора ошибки 22 байта;

• запуск процессов, формирующих сбои и ошибки в потоках данных шлюзовой машины;

• формирование архивов данных и считывание архивов данных через инструментальный канал в сервисный компьютер специальной утилитой.

Инициализация и блокирование генераторов

Модуль МАСТЕР: возможность работы генераторов ошибок определяется установкой перемычки на плате, которая разрешает доступ к инструментальному блоку ПЗУ. В этом блоке хранятся исполняемые коды генераторов и отведено место для протоколов. Пуск генераторов осуществляется автоматически после синхронизации часов системы.

Модули ввода и вывода команд: возможность работы генераторов сбоев определяется установкой перемычек на платах модуля вывода ответственных команд и на платах ввода-вывода, обычных команд. Установленные перемычки разрешают доступ к инструментальному блоку Flash-ПЗУ, в котором хранится исполняемый код генератора. Значения векторов сбойных ошибок $\boldsymbol{q}$ и помех в каналах передачи данных $\boldsymbol{h}$ вводятся на начальном этапе испытаний через инструментальный последовательный канал RS-232 из сервисного компьютера специальной утилитой.

Шлюзовая машина и персональный компьютер поездного диспетчера: исполняемый код генераторов ошибок загружается с сервисной дискеты. Работа генератора блокируется при отсутствии дискеты. После выключения электропитания образ генераторов в оперативной памяти уничтожается. Инициализа-

ция работы генераторов осуществляется командой с подключаемой к шлюзовой машине рабочей клавиатуры или командой по рабочему каналу от верхней сети.

Считывание данных архивов

Модуль МАСТЕР: считывание результатов испытаний осуществляется через инструментальный канал в сервисный компьютер. В состав генераторного блока модуля МАСТЕР включен программный модуль обмена с сервисным компьютером, который позволяет считывать три блока протокола.

Модули ввода и вывода команд, шлюзовая машина: считывание данных архивов осуществляется через инструментальный последовательный канал RS-232 в сервисный компьютер специальной утилитой.

Персональный компьютер поездного диспетчера: во время работы независимые генераторы сбоев и ошибок вследствие помех в каналах передачи данных записывают на жесткий диск два блока протокола. Каждый блок состоит из заголовка и данных и обеспечивает идентификацию и целостность протокола. Считывание протокола испытаний осуществляется стандартными средствами копирования файлов.

III.6.5.4. Порядок проведения испытаний

Испытания проводятся в следующем порядке:

- подготовка исходных данных (определение входных параметров, расчет априорных вероятностей сбойных ошибок и помех);
- проверка комплектности аппаратуры и документации;
- проверка соединения оборудования;
- введение исходных данных для испытаний;
- запуск системы.

Остановимся более детально на этапе подготовки исходных данных.

Входными параметрами для проведения испытаний являются:

g – априорная вероятность сбойной ошибки при выполнении устройством одной операции;

q – имитируемая τ-генератором вероятность сбойной ошибки при выполнении устройством одной операции;

p – априорная вероятность ошибки в одном бите информации и в тракте передачи данных;

h – вероятность ошибки на один бит в потоке тракта передачи данных при воздействии z-генератора;

N – количество прогонов команд в системе при проведении испытаний;

A – значение показателя безопасности ;

P – доверительная вероятность;

t_k – максимальное время реакции системы на запрос диспетчера;

dt – длительность исполнения одного такта ведущего процессора устройства;

v – скорость передачи данных в потоке тракта передачи данных;

m – длина искажаемого результата;

k – количество устройств в системе;

l – количество информационных потоков в системе.

Значения векторов вероятностей ошибок h и q вычисляются по разработанным программам на двух разных компьютерах. Все результаты вычислений сравниваются. Результаты расчетов вероятностей ошибки за счет помех и сбойной ошибки и при доверительной вероятности $P = 0{,}95$, минимальном значении показателя $A = 10^{-12}$ (вероятности трансформации одного бита сообщения) и $N = 9600$ прогонов команд управления приведены в таблицах III.6.1 и III.6.2 соответственно.

Таблица III.6.1. Результаты расчета вероятности h

Номера генераторов помех	V	p	t_k	l	Результат расчета
1, 2, 11, 12	10^7	10^{-7}	2,2	4	$2{,}0\times10^{-6}$
3, 4, 9, 10	$5{,}76\times10^4$	5×10^{-4}	2,2	2	$6{,}2\times10^{-4}$
5, 6, 7, 8	10^5	10^{-7}	2,2	5	$7{,}1\times10^{-6}$

Таблица III.6.2. Результаты расчета вероятности q

Номера устройств и генераторов сбоев	dt	g	t_k	k	Результат расчета
Шлюзовые машины и ПЭВМ 1, 2, 3, 4	5×10^{-9}	5×10^{-10}	2,2	4	$3{,}6\times10^{-9}$
Модули МАСТЕР 5, 6	4×10^{-8}	5×10^{-10}	2,2	2	$2{,}8\times10^{-7}$
МВОК, МВВОД, МВЫВ 7, 8, 9, 10, 11	$6{,}25\times10^{-8}$	5×10^{-10}	2,2	5	$1{,}9\times10^{-7}$

При введении исходных данных на платах линейного пункта КТС «Тракт» устанавливают соответствующие перемычки для обеспечения работы генераторов сбоев и помех. Последовательный канал сервисного компьютера подключают к инструментальному разъему модуля МАСТЕР. Вводят исходные данные согласно таблицам III.6.1 и III.6.2. Указанные действия повторяют для всех модулей резервных комплектов.

Вставляют диски с телом соответствующих *z*-генераторов и τ-генераторов в приемные карманы накопителей персональных компьютеров поездных диспетчеров и шлюзовых машин основных и резервных комплектов.

Расчетная продолжительность испытаний – 6 часов.

III.6.5.5. Обработка и оценка результатов испытаний

Формулы для обработки результатов испытаний

Пусть L_j, $P_{\text{ТРТУ}j}$, $P_{\text{ТРТС}j}$, $P_{\text{ПТУ}j}$, $P_{\text{ПТС}j}$ – программные счетчики событий на j-м прогоне соответственно опасных отказов (L), трансформации сигналов телеуправления и телесигнализации ($P_{\text{ТРТУ}}$ и $P_{\text{ТРТС}}$), потери сигналов телеуправления и телесигнализации ($P_{\text{ПТУ}}$ и $P_{\text{ПТС}}$).

Интенсивность опасных отказов – L. При обработке архивов фиксируются опасные отказы. Если опасный отказ произошел на j-м прогоне команды, то $L_j = L + W_j$. В итоге интенсивность отказов определяется как $L_j/(t_k \cdot N \cdot 3600)$, где t_k – время реакции системы на изменения входных данных, N – количество испытаний.

Вероятность трансформации сигнала ТУ – $P_{\text{ТРТУ}}$. При обработке архивов фиксируются трансформации сигнала ТУ. Если трансформация произошла на j-м прогоне, то $P_{\text{ТРТУ}j} = P_{\text{ТРТУ}} + W_j$. В итоге вероятность трансформации сигнала ТУ после N испытаний $P^*_{\text{ТРТУ}} = P_{\text{ТРТУ}j} \big/ N$.

Вероятность трансформации сигнала ТС – $P_{\text{ТРТС}}$. При обработке архивов фиксируются трансформации сигнала ТС. Если трансформация произошла на j-м прогоне, то $P_{\text{ТРТС}j} = P_{\text{ТРТС}} + W_j$. В итоге вероятность трансформации сигнала ТС после N испытаний равна $P^*_{\text{ТРТС}} = P_{\text{ТРТС}j} \big/ N$.

Вероятность потери информации в канале ТУ – $P_{\text{ПТУ}}$. При обработке архивов фиксируются потери информации в канале ТУ. Если потеря произошла на j-ом прогоне, то $P_{\text{ПТУ}j} = P_{\text{ПТУ}} + W_j$. В итоге вероятность потери информации в канале ТУ после N испытаний равна $P^*_{\text{ПТУ}} = P_{\text{ПТУ}j} \big/ N$.

Вероятность потери информации в канале ТС – $P_{\text{ПТС}}$. При обработке архивов фиксируются потери информации в канале ТС. Если потеря произошла на j-ом прогоне, то $P_{\text{ПТС}j} = P_{\text{ПТС}} + W_j$.

В итоге вероятность потери информации в канале ТС после N прогонов равна $P^{*}_{\text{ПТС}} = P_{\text{ПТУ}j} \big/ N$.

Весовой коэффициент на j-м прогоне испытаний вычисляется по формуле

$$D_j = \frac{G(W_j, Z_j)}{H(W_j, Z_j)} = \frac{\prod_{i=1}^{C_1} e^{-(t_k - t_j) g_i / \Delta t_j} g_i^{w_j} \prod_{i=1}^{C_2} e^{-(t_k - t_j) p_i / \Delta t_j} p_i^{z_j}}{\prod_{i=1}^{C_1} e^{-(t_k - t_j) q_i / \Delta t_j} q_i^{z_j} \prod_{i=1}^{C_2} e^{-(t_k - t_j) h_i / \Delta t_j} h_i^{z_j}},$$

где $C_1 = 11$ – количество подключенных к устройствам τ-генераторов сбоев, $C_2 = 12$ – количество подключенных к каналам передачи информации z-генераторов помех.

Нормируемые значения показателей и результаты испытаний приведены в табл. III.6.3.

Если в результате N прогонов команд управления контролируемое (негативное) событие не происходит, то соответствующая этому событию интенсивность опасных отказов системы не превышает значения A (см. (6.3)) и, следовательно, значение A показателя является его нижней границей.

Таблица III.6.3 – Результаты испытаний...

п/п	Наименование показателя	Нормируемые значения	Результаты испытаний
1	Вероятность опасного отказа за 1час = (равна интенсивности опасных отказов)	УПБ-2 10^{-6}–10^{-7}	+
2	Вероятность трансформации сигнала ТУ	10^{-14}	–
3	Вероятность трансформации сигнала ТС	10^{-8}	+
4	Вероятность потери информации в канале ТУ	10^{-10}	+
5	Вероятность потери информации канале ТС	10^{-8}	+

Символ «+» в табл. III.6.3 означает, что по данному показателю (например, интенсивности опасных отказов) система выдержала контрольные испытания. Символ «–» означает что по данному показателю (вероятность трансформации команды телеуправления) система не выдержала испытания. Последующий разбор результатов показал, что задано чрезмерно завышенное требование и по существующим стандартам вероятность трансформации команд телеуправления не должна быть ниже уровня 10^{-12}. В этом случае результат испытаний положителен.

III.6.6. Подтверждение соответствия программных средств

Программные средства олицетворяют «мозг» информационной системы. От их качества, безопасности, информационной защищенности, соответствия заданным требованиям зависят функциональная надежность и безопасность всей информационной системы. Отсюда следует необходимость во всесторонних экспертизах и испытаниях программных средств. В настоящее время сформировалась практика подтверждения соответствия программных средств по следующим направлениям:

• контрольные испытания *качества и функциональной безопасности* в целях представления доказательственных материалов для декларирования соответствия программных средств. Требования обязательной сертификации путем декларирования соответствия обычно содержатся в технических регламентах соответствующей отрасли промышленности или сельского хозяйства, например в технических регламентах Таможенного союза для железнодорожного транспорта [89, 90, 91]. В этих регламентах подчеркнуто, что доказательственная база формируется заявителем с привлечением третьей аккредитованной независимой стороны;

• контрольные испытания по *требованиям безопасности информации* на отсутствие в программных средствах НДВ. Эти испытания проводятся на основе руководящего документа [110] Федеральной службы по техническому и экспортному контролю РФ (ФСТЭК России) и основываются на анализе исходных текстов программ;

• аудит защиты от *несанкционированного доступа* (НСД) и экспертиза киберзащищенности программного обеспечения.

III.6.6.1. Основные положения методики испытаний качества и функциональной безопасности программных средств

Методика устанавливает типовую номенклатуру показателей качества и функциональной безопасности ПО, метрики показателей качества и модели оценивания ПО, методы тестирования и экспертизы модификаций ПО.

Цель испытаний состоит в оценке соответствия ПО требованиям качества и функциональной безопасности согласно установленному для данного объекта испытаний уровню полноты безопасности.

Наряду со стандартными разделами (область применения, цель испытаний, нормативные документы, термины и определения, объект испытаний) в методике детализируются следующие специфические для каждого объекта испытаний разделы:

• виды и последовательность проведения испытаний, определяемые характеристики качества, надежности и функциональной безопасности;

• условия проведения испытаний функциональных возможностей ПО;

• обработка результатов испытаний.

Перед проведением испытаний заявитель (разработчик, продавец продукции) должен сформировать перечень деклариру-

емых показателей ПО в соответствии с требованиями Технических регламентов на данную продукцию и согласовать их с заказчиком (покупателем продукции или эксплуатирующей организацией). В случае добровольной сертификации заявитель формирует перечень сертификационных показателей и также согласовывает с заказчиком продукции. После этого независимая аккредитованная испытательная лаборатория проводит идентификацию испытуемого программного обеспечения. Результаты идентификации оформляются «Актом идентификации образца», в котором указываются:

– заявитель (наименование организации, юридический адрес);

– основание для испытательных работ;

– наименование программного продукта (включая вид носителя, номер и дату записи);

– цель идентификации;

– место идентификации;

– средства идентификации (например, «Фиксация и контроль исходного состояния программного комплекса (ФИКС), сертификат);

– результат идентификации в виде таблицы из двух разделов: наименование и номер носителя и контрольные суммы.

Акт идентификации подписывается экспертом и утверждается руководителем испытательной лаборатории.

Принятие решения о качестве ПО производится на основе результатов оценки соответствия показателей качества заданным требованиям [109]. Контрольные вопросы для испытаний и последующей оценки качества программного обеспечения могут быть сформированы путем анализа характера ошибок программного обеспечения, представленных в документе «Регистрация данных о качестве. Каталог дефектов продуктов программного обеспечения».

Контрольные вопросы изменяются в зависимости от представленного ПО. В качестве исходных данных для определения соответствия качества ПО заданным требованиям используются тексты программ, документация, результаты испытаний функциональных возможностей на стендах и/или на действующих объектах, а также результаты тестовых прогонов программного обеспечения и математического моделирования.

В таблице III.6.4 приведен фрагмент требований к характеристикам качества. В этом фрагменте приведены требования стандартов к свойствам характеристики качества «Функциональные возможности программного обеспечения».

Таблица III.6.4. Фрагмент требований к характеристикам качества

№ п/п	**Требования к свойствам характеристики качества «Функциональные возможности»**	**Ссылки на нормативно-техническую документацию**
1.	**Пригодность**	
1.1	Соответствие функций требованиям	[109], [57]
2.	**Правильность**	
2.1	Соответствие программной документации согласно спецификации	[109]
2.2	Наличие всех файлов программного обеспечения	[58]
2.3	Наличие всех функций программного обеспечения	[109]
2.4	Наличие средств диагностики всех некорректных ситуаций	[109]
2.5	Наличие всех элементов интерфейса с пользователем	[61]
2.6	Отсутствие противоречий в выполнении функции (по сравнению со спецификациями требований или документацией)	[109], [57]

2.7	Отсутствие противоречий в выполнении алгоритмов	[109], [57]
3.	**Способность к взаимодействию**	
3.1	Обеспечение информационного взаимодействия отдельных частей системы	[109], [57]
3.2	Обеспечение взаимодействия с другими смежными системами и системами верхнего уровня	[109], [57]
4.	**Согласованность**	
4.1	Соответствие комплектности, содержания и оформления документации требованиям стандартов	[58] – [61]
4.2	Соответствие интерфейса ПО требованиям безопасности	[109], [57]
4.3	Лицензионная чистота заимствованного ПО	[109], [57]
5.	**Защищенность**	
5.1	Ведение списка учетных записей пользователей, допущенных к работе с ограничением их прав	[109], [57]

Испытания функциональных возможностей ПО проводятся с использованием аттестованных должным образом стендов испытательной лаборатории или стендов разработчика. В последнем случае должна быть оформлена аренда на эти стенды. Испытания проводятся с помощью сценариев реализации предусмотренных функций ПО. Рабочие протоколы испытаний функциональных возможностей должны содержать подробное описание результатов проведенных испытаний, включая графики, рисунки и выводы о надлежащем выполнении испытываемых функций. Если это представляется возможным, то испытания проводятся непосредственно на объекте.

Завершающим этапом испытаний качества ПО является этап интеграционных испытаний на работающем объекте. Для этого

ПО загружается с идентифицированного носителя информации и проводятся испытания на взаимодействие всего комплекса программных модулей.

Испытания* ПО на соответствие *требованиям стандартов по функциональной безопасности [55, 56, 57, 63 и др.] осуществляются на основе предварительной оценки риска и на этой основе определения уровня полноты безопасности выполняемых программным обеспечением функций. Уровень полноты безопасности устанавливается по результатам оценки частоты (или вероятности) возникновения опасного события при реализации каждой конкретной функции. Вследствие того, что эти события реально возникают очень редко, то для определения их частоты (вероятности) проводятся ускоренные натурные испытания (см. пп. 6.4 и 6.5). Если проведение ускоренных натурных испытаний по каким-либо причинам не представляется возможным, то для решения этой задачи можно ограничиться математическим моделированием вероятности возникновения опасного события для ПО в целом.

Испытания соответствия функциональной безопасности ПО требованиям соответствующих Технических регламентов проводятся для целей декларирования соответствия согласно [62]. До начала испытаний заявитель должен представить перечень декларируемых показателей программного обеспечения. Эти показатели должны содержать требования Технического регламента, под действие которого подпадает представленная на испытания информационная система с данным ПО. Декларируемые показатели должны содержать предварительно установленный уровень полноты безопасности в соответствие с [57], а также показатели качества по ГОСТ ИСО/МЭК 9126 и ИСО/МЭК 25010.

Для проведения испытаний ПО разрабатывается рабочая методика. Разработчик должен представить обоснованные сведения с необходимыми пояснениями о соответствии производства

испытуемого ПО установленным для него требованиям установленного уровня полноты безопасности.

III.6.6.2. Основные положения методики испытаний по требованиям безопасности информации

Целью сертификационных испытаний ПО является контроль отсутствия в нем НДВ в соответствии с требованиями руководящего документа [110], с учетом требований по функциональной безопасности ПО стандарта [57].

Перечень проверок и испытаний в соответствии с требованиями стандартов и нормативных документов приведен в таблице III.6.5

При проведении испытаний оцениваются:

- состав и содержание документации (оцениваются по показателю «соответствует – не соответствует» требованиям руководящего документа);
- полнота и отсутствие избыточности исходных текстов (полнота оценивается по показателю «полные – неполные» исходные тексты, отсутствие избыточности – по количеству избыточных файлов и функциональных объектов);
- соответствие исходного состояния ПО описанию, представленному в документации (оценивается по показателю «соответствует – не соответствует»);
- соответствие исходных текстов ПО его объектному (загрузочному) коду (оценивается по показателю «соответствуют – не соответствуют»);
- количество функциональных объектов в программном обеспечении;
- количество информационных объектов в программном обеспечении;
- количество обращений к каждому функциональному объекту из других функциональных объектов программного обеспечения;

• количество обращений к каждому информационному объекту из функциональных объектов программного обеспечения;

• соответствие фактических маршрутов выполнения функциональных объектов и маршрутов, построенных в процессе проведения статического анализа (оценивается по показателю «соответствуют – не соответствуют»);

• уровень контроля полноты и корректности реализации технических приемов в соответствии с требованиями стандартов [56, 57].

Испытания проводятся в условиях, определенных в документации на ПО.

1. Результаты контроля соответствия состава документации требованиям стандартов [58 – 61, 109 и 111] учетом специфики ПО оформляются в виде таблицы в соответствующем разделе протокола сертификационных испытаний (таблица III.6.6).

Таблица III.6.5. Перечень проверок в соответствии с требованиями нормативных документов

№ п/п	**Вид испытания**	**Ссылки на документы с требованиями к ПО [55 – 61] ,[110] и [111]**
1	***Контроль состава и содержания документации***	**[58 – 61,111]**
2	***Контроль исходного состояния ПО***	**[110]**
3	***Статический анализ исходных текстов программ:***	**[55, 56, 57, 110]**
3.1	• Контроль полноты и отсутствия избыточности исходных текстов	[56, 57, 110]
3.2	• Контроль соответствия исходных текстов ПО его объектному (загрузочному) коду	[56, 57, 110]
3.3	• Контроль связей функциональных объектов по управлению	[56, 57, 110]

№ п/п	Вид испытания	Ссылки на документы с требованиями к ПО [55 – 61] ,[110] и [111]
3.4	• Контроль связей функциональных объектов по информации	[56, 57, 110]
3.5	• Контроль информационных объектов	[56, 57, 110]
3.6	• Формирование перечня маршрутов выполнения функциональных объектов	[56, 57, 110]
4	Динамический анализ исходных текстов программ:	[55, 56, 110]
4.1	• Контроль выполнения функциональных объектов	[55, 56, 110]
4.2	• Сопоставление фактических маршрутов выполнения функциональных объектов и маршрутов, построенных в процессе проведения статического анализа	[55, 56, 110]
5	***Контроль полноты и корректности реализации технических приемов***	**[55, 56, 57]**
6	***Оформление результатов испытаний***	**[110]**

Контроль содержания документации выполняется в два этапа:

1) Проводится экспертиза по каждому документу в отдельности на соответствие требованиям стандартов Единой системы программной документации (ЕСПД), с регистрацией замечаний по характеристикам, представленным в таблице III.6.**7**.

2) На каждый документ заполняется форма, аналогичная таблице III.6.**7**. Результаты оценки в виде чисел в интервале [0…1] заносятся в правую колонку таблицы. Некоторые характеристики, отмеченные звездочкой, применимы не ко всем документам. В формах конкретных документов, к которым они неприменимы, допускается либо опускать эти характеристики, либо в графе «Оценка» проставлять «НП» («неприменима»).

Таблица III.6.6. Форма для регистрации результатов контроля состава документации

№	Требуемые документы	Результаты контроля
1.	Техническое задание [58]	
2.	Формуляр [60]	
3.	Спецификация [59]	
4.	Описание программы [111]	
5.	Тексты программ [61]	
6.	План верификации и валидации ПО [55, 56]	
7.	План интеграционных испытаний ПО [55, 56]	
8.	Отчет по верификационным и валидационным испытаниям ПО [55, 56]	
9.	Отчет по интеграционным испытаниям ПО [55, 56]	

Общая оценка каждой характеристики определяется с учетом важности документов, исходя из области применения ПО.

Таблица III.6.7. Форма для оценки программной документации

№	Требования	Оценка
	Общая оценка <Вид документа / Документации ПО >	
1.	Соответствие требованиям стандартов [55-61,109,111]	
2.	Понятность документации	
3.	*Наличие описания функций ПО	
4.	* Лицензионная чистота заимствованного ПО	
5.	Правильность документации	
6.	Приемлемость уровня технического исполнения документации	
7.	*Наличие краткой аннотации	
8.	*Наличие описания решаемых задач	
9.	*Наличие описания ограничений по применению	
10.	*Наличие описания алгоритмов	
11.	*Наличие описания входных и выходных данных	

№	Требования	Оценка
12.	*Наличие описания характеристик ПО	
13.	*Наличие описания оборудования, необходимого для функционирования ПО	
14.	*Наличие описания программной среды функционирования ПО и др. используемых программ	
15.	Наличие оглавления	
16.	Соответствие оглавления содержанию документации	
17.	Грамматическая правильность изложения в документации	
18.	Отсутствие противоречий	
19.	Отсутствие неправильных ссылок	
20.	Ясность формулировок и описаний	
21.	Отсутствие неоднозначных формулировок и описаний	
22.	Правильность использования терминов	
23.	Краткость, отсутствие лишней детализации	
24.	Единство формулировок	
25.	Единство обозначений	
26.	Отсутствие ненужных повторений	
27.	Наличие нужных объяснений	
28.	Приемлемость уровня стиля изложения	
29.	Ясность логической структуры	
30.	Наличие непрерывной нумерации страниц документов	
31.	Отсутствие незаконченных разделов, абзацев, предложений	
32.	Наличие всех рисунков, формул, таблиц	
33.	Логический порядок частей внутри главы	

2. Контроль исходного состояния ПО производится путем *идентификации* испытуемого образца программы аналогично тому, как это описано в п. III.6.6.1. В соответствии с требованиями нормативного документа [110], контроль исходного состояния заключается в фиксации состояния всего ПО и срав-

нении полученных результатов с результатами, приведенными в документации.

Результатами контроля исходного состояния должны быть рассчитанные уникальные значения контрольных сумм исходных текстов программ, исполняемых и загрузочных модулей, входящих в состав ПО. Контрольные суммы должны рассчитываться для каждого файла, входящего в состав ПО.

Результаты контрольного суммирования оформляются в виде электронных приложений к протоколу сертификационных испытаний.

3. Статический анализ исходных текстов программ предполагает проведение следующих технологических операций:

• контроль соответствия исходных текстов ПО его объектному (загрузочному) коду;

• контроль полноты и отсутствия избыточности исходных текстов ПО на уровне файлов (функциональных объектов);

• контроль связей функциональных объектов по управлению;

• контроль связей функциональных объектов по информации;

• контроль информационных объектов ПО различных типов (локальных переменных, глобальных переменных, внешних переменных и т. п.);

• формирование перечня маршрутов выполнения функциональных объектов.

Контроль соответствия исходных текстов ПО объектному (загрузочному) коду производится на основе представленных на испытания Описаний программ в присутствии представителей Заявителя проводится контрольная компиляция исходных текстов и сборка всех загрузочных модулей ПО. С помощью программы ФИКС фиксируются размер, дата записи и контрольная сумма представленных и собранных объектных (загрузочных) файлов. Оформляется Акт сборки ПО, в котором результаты сравнения представляются в виде таблицы, имеющей следующую структуру:

• **Имя файла**	имя объектного (загрузочного) модуля;
• **Размер представленного файла**	размер модуля (в байтах), представленного для проведения испытаний;
• **Размер собранного файла**	размер модуля (в байтах), собранного в процессе испытаний;
• **Дата записи представленного файла**	дата записи модуля, представленного для проведения испытаний, в формате: час, минута, секунда, день, месяц и год записи;
• **Дата записи собранного файла**	дата записи модуля, собранного в процессе испытаний, в формате: час, минута, секунда, день, месяц и год записи;
• **Контрольная сумма представленного файла**	контрольная сумма файла по алгоритму «ВКС» в шестнадцатеричном формате, представленного для проведения испытаний;
• **Контрольная сумма собранного файла**	контрольная сумма файла по алгоритму «ВКС» в шестнадцатеричном формате, собранного в процессе испытаний;
• **Результат сравнения**	проставляется отметка «одинаковы», если при сравнении контрольные суммы файлов оказались идентичными; если в результате сравнения обнаружены различия – указывается количество различий, а также предпринятое действие для установления соответствия модулей и вывод о соответствии модулей.

Затем осуществляется *контроль полноты и отсутствия избыточности исходных текстов* на основании регистрации факта успешной компиляции и сборки исполняемых файлов всех компонентов ПО. По результатам сравнения делается вывод о соответствии ПО описанию, представленному в документации. Контроль отсутствия избыточности исходных текстов произ-

водится путем сравнения перечня функциональных объектов, определенных в файлах исходных текстов, с перечнем функциональных объектов, код которых содержится в исполняемом (загрузочном) коде. Результаты оформляются в виде электронного приложения к протоколу сертификационных испытаний.

Результаты контрольной компиляции и сборки ПО оформляются в виде электронного приложения к протоколу сертификационных испытаний.

Полученные в результате компиляции исполняемые (загрузочные) модули сравниваются при помощи программы ФИКС с соответствующими модулями, представленными на испытания в составе исполняемого ПО. В случае совпадения сравниваемых модулей делается вывод о соответствии исходных текстов ПО его объектному (загрузочному) коду. В случае несовпадения – проводится анализ характера обнаруженных различий с точки зрения их влияния на логику выполнения программы. По результатам анализа делается вывод о соответствии или несоответствии представленных исходных текстов полученным программным модулям.

Полнота исходных текстов ПО подтверждается фактом успешной сборки всех компонентов типового ПО и контрольного примера с учетом установленного соответствия исходных текстов объектному (загрузочному) коду.

Контроль избыточности исходных текстов на уровне функциональных объектов осуществляется путем выявления факта наличия кода функционального объекта в исполняемом (загрузочном) файле.

Функциональные объекты, код которых не вошел в исполняемые (загрузочные) модули, полученные в результате контрольной сборки ПО, признаются избыточными.

Файлы исходных текстов, не содержащие определения ни одного функционального объекта, код которого вошел в исполняемые (загрузочные) модули, признаются избыточными.

Для *контроля связей функциональных объектов по управлению, связей функциональных объектов по информации и контроля информационных объектов* выполняется регистрация следующей информации:

- матрицы связей функциональных объектов по управлению;
- матрицы связей функциональных объектов по информации;
- матрицы обращений к идентификаторам в функциональных объектах;
- матрицы достижимости переменных из функциональных объектов.

Эти процедуры регистрации информации осуществляются для всех программных модулей ПО с помощью средств анализатора исходных текстов.

Результаты оформляются в виде электронного приложения к протоколу сертификационных испытаний.

Условиями прекращения испытаний на начальном этапе являются:

- несоответствие программного обеспечения документации на него;
- нарушение работоспособности и устойчивости функционирования программного обеспечения.

Испытания могут быть возобновлены после устранения обнаруженных несоответствий и нарушений.

4. Динамический анализ программных модулей проводится с целью подтверждения маршрутов выполнения функциональных объектов, полученных в результате статического анализа и включает:

- контроль выполнения функциональных объектов;
- сопоставление фактических маршрутов выполнения функциональных объектов и маршрутов, построенных по результатам статического анализа.

С помощью встраиваемых датчиков контроля подготавливается технологическая версия исходных текстов ПО. Датчик представляет собой вызов функции с уникальным значением фактического параметра. Датчик вставляется в файлы с исходными текстами исследуемых компонентов ПО в начало и в конец каждого функционального объекта.

На основании анализа архитектуры ПО и решаемых им задач, описанных в программной документации, составляется программа испытаний ПО на стенде, а также определяется технология формирования файлов трасс.

Для динамического анализа исполняемые (загрузочные) файлы компонентов ПО заменяются исполняемыми (загрузочными) файлами лабораторной версии, полученной путем компиляции и сборки исходных текстов с внедренными вызовами дополнительных функций-датчиков, вызываемых при входе в процедуру и всех возможных выходах из нее. Вызовы датчиков внедряются в исходные тексты компонентов ПО, являющимися критичным к наличию НДВ с помощью анализатора исходных текстов.

Код процедуры-датчика, обеспечивающей формирование записей файлов трасс в процессе функционирования лабораторной версии ПО, разрабатывается и реализуется экспертами испытательной лаборатории совместно с Разработчиком в процессе проведения сертификационных испытаний.

В результате испытаний ПО формируются файлы трасс, содержащие последовательность уникальных значений индексов внедренных датчиков. Полученные файлы трасс обрабатываются и анализируются экспертами с целью формирования перечня фактических маршрутов выполнения функциональных объектов.

Для компонентов ПО, функционирующих под управлением многозадачных ОС, полученные файлы трасс перед началом анализа разделяются по критерию принадлежности вызова функции-датчика тому или иному программному потоку. Тех-

нология формирования файлов трасс в случае использования многозадачной ОС разрабатывается и реализуется экспертами испытательной лаборатории совместно с Разработчиком в процессе проведения сертификационных испытаний.

В случае, если получение файлов трасс методом формирования лабораторной версии ПО невозможно, для проведения динамического анализа используются инструментальные средства диагностики, отладки и трассировки интегрированных сред разработки, интерактивного дизассемблера, например IDA Pro, а также аппаратно-программные комплексы стенда Разработчика, используемые при автономном тестировании, отладке и заводских испытаниях ПО.

Сборка технологической версии исполняемых модулей ПО на основе технологической версии исходных текстов должна осуществляться также, как и контрольная сборка ПО, представленного на испытания. Полученная технологическая версия исполняемых модулей ПО запускается на испытательном стенде, после чего выполняется последовательность действий, позволяющая смоделировать наиболее важные состояния системы. Результатом тестирования являются файлы трасс, содержащие уникальные индексы встроенных датчиков, к которым осуществлялось обращение. Файлы трасс каждого теста (серии тестов) обрабатываются и сопоставляются с маршрутами выполнения функциональных объектов, полученными в результате статического анализа.

Результаты испытаний представляются в виде деревьев вызовов функциональных объектов или графов состояний системы и оформляются в виде электронного приложения к протоколу сертификационных испытаний.

Фактические маршруты выполнения функциональных объектов и маршруты, построенные в процессе проведения статического анализа, сопоставляются между собой. На ос-

новании анализа этих маршрутов, построенных в процессе проведения статического и динамического анализа, делается вывод об отсутствии недекларированных возможностей в исследуемом ПО.

5. Контроль полноты и корректности реализации технических приемов подразумевает собой контроль объема технических мер обеспечения безопасности информации, реализованных при разработке ПО. Результаты контроля полноты технических приемов, используемых для разработки оцениваемого ПО в соответствии с требованиями стандартов [55, 56, 57 и др.] оформляются в виде таблиц в соответствующем разделе протокола сертификационных испытаний.

Контроль полноты и корректности реализации технических приемов заключается в определении состава использованных технических приемов при разработке ПО, проверке качества реализации данных приемов. Каждый технический прием должен быть функционально законченным, должен охватывать одно или несколько требований стандартов [55, 56, 57 и др.], быть непротиворечивым. Реализация приема должна быть логичной, рациональной, несложной для анализа. Весь объем технических приемов должен охватывать все требования определенного уровня полноты безопасности ПО.

По результатам контроля делается вывод о соответствии представленного объема технических приемов необходимому объему технических требований для соответствия системы установленной категории защищенности с учетом требований стандартов [55, 56, 57 и др.]

Ниже приведена расшифровка возможных значений требований:

«+» – этот символ означает, что технический прием или мероприятие должно быть **использовано/учтено** при разработ-

ке ПО. В случае неиспользования/замены требования другим приемом стандарта в документации программного обеспечения должно быть дано обоснование его неиспользования. Заметим, что данный символ не обязательно означает включение соответствующего технического приема в разработку, он лишь обозначает необходимость учесть соответствующее требование стандарта, например [55], которое может наоборот не рекомендовать использование приема (в этом случае около приема сделана пометка *не рекомендуется*);

«–» – этот символ означает, что технический прием или мероприятие *не рекомендовано* при разработке ПО.

Рекомендации по использованию определенных технических приемов содержатся в наборе таблиц. Каждая таблица соответствует определенному этапу разработки программного обеспечения. В таблице III.6.8 приведен пример задания технических приемов при разработке архитектуры ПО системы управления на основе ряда требований стандарта [55] .

Таблица III.6.8. Архитектура программного обеспечения

Технический прием/мероприятие	**Описание**	**Требование**
Защищенное программирование	ПО разработано таким образом, что способно обнаруживать аномальный поток команд управления или данных. Например, ПО способно проверить переменные на предмет соответствия диапазону, параметры функций на достоверность кол-ва, типа, размера; проверить собственную конфигурацию и целостность.	+

Технический прием/мероприятие	**Описание**	**Требование**
Обнаружение неисправности и диагностика	Использование специальных технических приемов, направленных на обнаружения и устранение ошибочных ситуаций в системе, как для программной (само-диагностика, подушки безопасности), так и для аппаратной части (избыточность, коды обнаружения ошибок)	+
Коды исправления ошибок	Для сохранения целостности информации используются коды исправления ошибок (Коды Хемминга, полиномиальные коды)	+
Коды обнаружения ошибок	Для сохранения целостности информации используются коды исправления ошибок (Коды Хемминга, полиномиальные коды)	+
Многоверсионное программирование	При разработке ПО используется репозиторий, содержащий предыдущие версии ПО	+
Блок восстановления	При возникновении ошибочных ситуаций происходит восстановление системы путем повторного выполнение сбойной операции в ПО. Это требование подразумевает наличие системы контроля безопасности, а также наличие системы регистрации событий.	+

Технический прием/мероприятие	**Описание**	**Требование**
Искусственный интеллект – исправление ошибок **(не рекомендуется)**	Способность ПО автоматически собирать наборы методов для восстановления системы после неисправности. Подразумевает оперативный анализ безопасности	– (этот прием очень усложняет ПО и для данной системы избыточен)
Динамическая реконфигурация программного обеспечения **(не рекомендуется)**	Способность ПО обнаруживать отказы в ресурсах системы и переконфигурировать архитектуру системы для функционирования на оставшемся ресурсе	– (этот прием избыточен)

6. Обработка, анализ и оценка результатов испытаний. Программное обеспечение считается выдержавшим все испытания при условии получения положительных оценок и выводов по результатам всех предусмотренных настоящей методикой технологических операций. Положительная оценка результатов испытаний формируется с учетом следующих критериев:

– состав и содержание информации соответствует требованиям нормативных документов [110, 58 – 61, 111]. При несоответствии состава содержания документов требованиям [110] положительный вывод по результатам испытаний может быть сделан в случае достаточности представленных сведений для проведения испытаний в соответствии с заявленным уровнем контроля;

– исходные тексты ПО представлены полностью;

– отсутствие избыточности в исходных текстах ПО. При наличии избыточности в исходных текстах положительный вывод

по результатам испытаний может быть сделан, если избыточные объекты не инициируют не описанных в документации функций ПО, приводящих к нарушению конфиденциальности, доступности или целостности обрабатываемой информации;

– исходное состояние ПО соответствует описанию, представленному в документации;

– исходные тексты ПО соответствуют его объектному коду;

– фактические маршруты выполнения функциональных объектов соответствуют маршрутам, построенным в процессе проведения статического анализа;

– полнота и корректность реализации технических приемов соответствует требованиям п. 6.3.5. При несоответствии состава технических приемов рекомендованному в данном документе положительный вывод по результатам испытаний может быть сделан в случае достаточности представленных сведений, описывающих причину неиспользования исключенного технического приема.

В процессе проведения испытаний возможно прекращение испытаний в следующих случаях:

• при установлении в процессе испытаний факта невозможности получения положительных выводов по результатам выполнения хотя бы одной из обязательных технологических операций, предусмотренных требованиями [110];

• при обнаружении в исследуемом программном обеспечении возможностей, не описанных или не соответствующих документации, при использовании которых возможно нарушение конфиденциальности, доступности или целостности обрабатываемой информации;.

Решение о прекращении сертификационных испытаний принимается по согласованию с Органом по сертификации.

В процессе проведения сертификационных испытаний допускается разработка частных Программ и методик испытаний

с целью детализации технологий проведения испытаний отдельных компонентов ПО. Решение о необходимости разработки частных Программ и методик сертификационных испытаний принимается экспертами Испытательной лаборатории и согласуется с Органом по сертификации.

По завершении испытаний оформляются протокол сертификационных испытаний и техническое заключение.

III.6.6.3. Порядок подтверждения соответствия требованиям комплексной безопасности программного обеспечения

Комплексная безопасность программного обеспечения в информационных системах управления (автоматизированных системах управления) – это сочетание в программном обеспечении следующих трех достоинств:

• *качество и функциональная безопасность;*

• *отсутствие НДВ;*

• *защищенность от несанкционированного доступа (НСД), в том числе от информационных атак, направленных на нарушение процесса управления (в частности кибератак,* см. п. 1.7.2). Подтверждение соответствия ПО требованиям защищенности от НСД производится путем аудита на основании требований нормативного документа [88].

В последнее время проявляется тенденция обсуждать комплексную безопасность ПО под углом зрения киберзащищенности информационной системы. Такой подход обедняет понимание комплексной безопасности программных средств, поскольку сосредоточен на защите от НСД, на обеспечении отсутствия НДВ и в некоторой мере учитывает требования функциональной безопасности. Комплексная безопасность охватывает более широкий спектр характеристик безопасности ПО, в том числе технологию разработки и производства ПО, технологию обеспечения

функциональной безопасности и, конечно, вопросы обеспечения стандартов качества, включая требования менеджмента качества, верификации программных модулей, документирования процессов сопровождения ПО, валидацию и интеграционные испытания ПО в составе информационной системы.

В представленных в пп. III.6.6.1 и III.6.6.2 основные положения методик подтверждения соответствия требованиям качества, функциональной безопасности и отсутствия НДВ решают задачи всесторонней оценки комплексной безопасности путем дополнения требований [88] к технологиям качественной разработки, производства и обеспечения функциональной безопасности ПО.

Требования к защите от НСД базируются на определении класса защищенности системы управления в зависимости от уровня значимости (УЗ) (критичности) обрабатываемой в ней информации.. Уровень УЗ в информационных системах управления движением поездов определяется степенью возможного ущерба от нарушения ее целостности (неправомерное уничтожение или модифицирование) и/или доступности (неправомерное блокирование) информации, в результате которого возможно нарушение штатного режима функционирования или незаконное вмешательство в процессы функционирования системы управления.

Степень возможного ущерба устанавливается заказчиком или оператором экспертным или иным методом и может быть: *высокой*, если в результате нарушения одного из свойств безопасности информации, повлекшего нарушение штатного режима функционирования системы управления, возможно возникновение чрезвычайной ситуации федерального или межрегионального характера; *средней*, если в результате нарушения одного из свойств безопасности информации (целостности, доступности, конфиденциальности), повлекшего нарушение штатного режи-

ма функционирования автоматизированной системы управления, возможно возникновение чрезвычайной ситуации регионального или межмуниципального характера; *низкой,* если в результате нарушения одного из свойств безопасности информации (целостности, доступности, конфиденциальности), повлекшего нарушение штатного режима функционирования автоматизированной системы управления, возможно возникновение чрезвычайной ситуации муниципального (локального) характера. В частности, информация, обрабатываемая в системе управления движением поездов (за исключением опасных грузов) не имеет высокого уровня значимости, связанного с ущербом федерального значения. Она имеет средний уровень значимости (критичности) (УЗ 2) или низкий уровень значимости (критичности) (УЗ 3), поскольку для одного из свойств безопасности информации (целостности, доступности) определена средняя или низкая степень ущерба и нет ни одного свойства, для которого определена высокая степень ущерба. Следовательно, информационные системы управления движением поездов следует отнести ко второму (К2) или третьему (К3) классам защищенности.

Состав мер защиты информации и их базовые наборы на примере систем класса защищенности К2 устанавливается следующим образом:

- требуются идентификация, аутентификация и управление доступом, защита машинных носителей информации, антивирусная защита, контроль защищенности информации, обеспечение целостности и доступности информации, защита системы электрической централизации в целом и ее компонентов, а также обеспечение безопасной разработки и управления обновлениями программного обеспечения системы в полном объеме требований в соответствии с [88];
- не требуется применение средств обнаружения вторжений и ограничения прав пользователей;

• решение задач комплексной безопасности информационных систем управления должно производиться на всех этапах их жизненного цикла;

• на стадии конкурса до принятия решения о создании системы заказчик совместно с разработчиком формирует концепцию системы, анализирует условия применения, оценивает возможные риски, определяет допустимые риски и с учетом этого устанавливает и распределяет требования к системе в части киберзащищенности.

На рис. III.6.2 показаны необходимые для выдачи экспертного заключения по комплексной безопасности процессы оценки

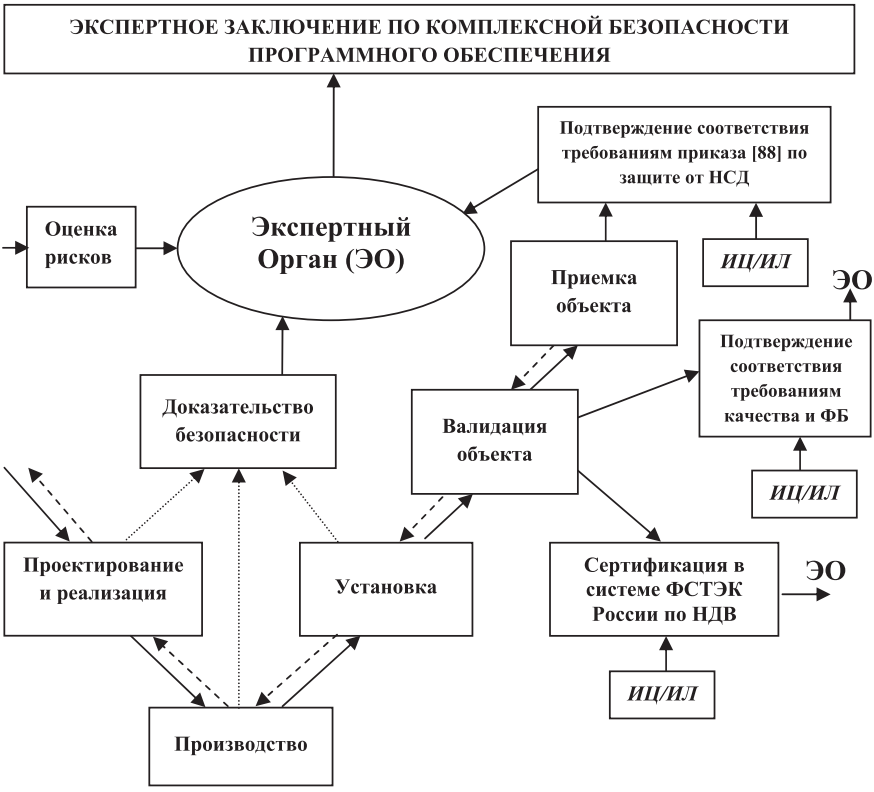


Рис. III.6.2. Подтверждение соответствия требованиям комплексной безопасности программного обеспечения

рисков, доказательства безопасности, сертификации, подтверждения соответствия ПО.

Эти процессы привязаны к соответствующим этапам жизненного цикла информационной системы. На стадии ***проектирования и реализации, производства и установки системы*** заказчик контролирует выполнение разработчиком требований нормативного документа [88]. Именно на этих этапах формируется доказательство безопасности ПО.

На ***этапах валидации*** и ***приемки*** системы должны завершиться сертификационные испытания программных средств системы на отсутствие НДВ. Для этого аккредитованным испытательным центром (лабораторией) проводятся статические, а при необходимости и динамические испытания исходных кодов программ. Положительный результат этих испытаний – сертификат, выданный ФСТЭК России. На этом же этапе должно быть подтверждено соответствие требованиям технических регламентов в части качества и функциональной безопасности программных средств системы. Подтверждение соответствия производится путем декларирования соответствия с привлечением аккредитованного в Росаккредитации испытательного центра (лаборатории).

Подтверждение соответствия в части защиты от НСД проводится на ***этапе приемки*** системы аккредитованным в системе ФСТЭК России испытательным центром (лабораторией)– ИЦ/И. Это подтверждение производится путем аудита системы на реальном объекте, который имеет определенные информационные связи и содержит конкретные подключения к корпоративной информационной среде.

Именно при таких условиях возможно установить уязвимости системы, выработать организационные и технические решения по их устранению. Возможности устранения информационных угроз и опасных отказов на стадии приемки

системы следует оценивать с помощью экспертного органа. В процессе эксплуатации системы должен осуществляться непрерывный мониторинг информационных атак. В процессе эксплуатации осуществляется техническое обслуживание и ремонт не только базовых аппаратных средств системы, но и встроенных или удаленных средств информационной защиты системы. При модификации, и, тем более, модернизации системы, повторно оцениваются риски и при необходимости повторно поэтапно обеспечивается *комплексная безопасность* системы.

III.6.7. Контрольные вопросы

1. В чем заключается подтверждение соответствия информационных систем?
2. Какие виды подтверждения соответствия вам известны?
3. В чем заключаются суть и содержание декларирования соответствия?
4. Какие виды испытаний вам известны и в чем заключается проблема натурных испытаний на надежность и функциональную безопасность информационных систем?
5. В чем заключается ускорение испытаний и какие методы ускорения испытаний вам известны?
6. Раскройте содержание метода Монте-Карло.
7. Раскройте содержание метода значимой выборки.
8. Опишите теоретические основы метода ускоренных натурных испытаний.
9. Изложите практические положения метода ускоренных натурных испытаний.
10. Опишите модель ускоренных натурных испытаний системы диспетчерской централизации на железнодорожном транспорте.

11. Раскройте алгоритмы генерации сбоев и помех в модели ускоренных натурных испытаний системы диспетчерской централизации.

12. Изложите порядок обработки результатов испытаний и принятия решения о подтверждении соответствия информационных систем требованиям стандартов.

13. Какие виды испытаний необходимы для подтверждения соответствия программных средств и в чем их назначение?

14. Опишите процедуры декларирования соответствия программных средств по требованиям стандартов качества и функциональной безопасности.

15. Опишите процедуры сертификационных испытаний программных средств на отсутствие недекларированных возможностей.

16. Опишите порядок подтверждения соответствия требованиям комплексной безопасности программного обеспечения.

ЗАКЛЮЧЕНИЕ

Весь спектр методов и способов построения надежных отказоустойчивых информационных систем разделяется на две группы, составляющих:

1. Методы анализа и синтеза структурной надежности объектов информационных систем; методы анализа и синтеза функциональной надежности выполнения информационных процессов; методы анализа и синтеза построения отказоустойчивых информационных систем реального времени, методы анализа и синтеза функциональной безопасности информационных систем управления.

2. Технология создания надежной элементной базы с учетом климатических условий, режимов работы, уровня интеграции и технологии изготовления, информационных нагрузок, гонок и состязаний сигналов и т.д.; технология разработки и сопровождения функционально надежных программных средств; способы построения качественных человеко-машинных (эргатических) информационных систем на основе теории и практики эргономики.

Данный проект, включающий книги по методам анализа структурной, функциональной надежности и методы синтеза надежности и отказоустойчивости информационных систем, ориентирован на разработку первой группы составляющих методов. Способы и приемы технологии разработки, проектирования и модернизации надежных информационных систем осталась за рамками данного проекта. На наш взгляд, их следует отдельно и детально описать на современном уровне развития

цифровой техники и средств проектирования программного обеспечения.

При проектировании надежной информационной системы целесообразно руководствоваться следующими постулатами:

- не существует абсолютной надежности информационных систем;
- введение различных видов контроля в состав информационной системы обеспечивает ее наблюдаемость, но вместе с тем приводит к снижению надежности системы;
- необходимым условием достижения приемлемой надежности системы есть введение избыточности;
- использование избыточности дает эффект только при условии наблюдаемости системы;
- надежность информационной системы должна обеспечиваться на всех этапах жизненного цикла;
- уровень надежности информационной системы ограничен экономическими рисками заказчика и эксплуатирующей организации.

Основным инструментом обеспечения структурной надежности информационных систем является структурная избыточность. Наибольший эффект достигается при применении структурного резервирования с восстановлением и разветвленной системой контроля. Однако и анализ, и синтез такого рода систем затруднен из-за необходимости строить и решать модели надежности, основанные на Марковских и полумарковских случайных процессах. В этом отношении заслуживают особого внимания читателя, на наш взгляд, графовые Марковские и полумарковские методы расчета и прогнозирования показателей надежности и функциональной безопасности систем, которые описаны в книге «Структурная надежность информационных систем». Эти методы имеют оригинальный характер и позволяют решить проблему размерности сложных систем с большим

числом состояний. Они обеспечивают строгий или приемлемый по точности приближенный расчет и прогнозирование всех стационарных или нестационарных показателей надежности или функциональной безопасности непосредственно по графу состояний системы без решения систем дифференциальных уравнений. Несомненный интерес представляют приведенные в этой книге результаты предельной оценки возможностей структурного резервирования, которые при реальных возможностях средств обнаружения отказов значительно скромнее анонсированных в широкой печати по структурному резервированию. В расчетах надежности устройств и систем автор стремился привлечь внимание специалистов к корректному решению задач обработки и представления результатов расчета и прогнозирования показателей надежности. Задачи оценки погрешности расчетов стоят в одном ряду с обработкой информации в условиях неполных и недостаточно достоверных данных.

Функциональная надежность была и остается наименее воспринимаемой специалистами областью общей теории надежности. Для перехода от надежности устройства, объекта, изделия, системы к надежности выполнения функции, процесса, к надежности услуги потребовалось уйти от стереотипов и разработать новую систему мерил – показателей применимых к функциональной надежности информационных систем управления. Речь идет, в первую очередь, об оценке достигнутого уровня надежности программного обеспечения. Свойства программ принципиально отличаются от свойств аппаратуры – программа не отказывает, не требует ремонта, не изнашивается и не стареет. Отсюда бессмысленно оценивать безотказность, ремонтопригодность, долговечность и сохраняемость программ. Наоборот, следует обратить внимание на то, что в программе, как правило, содержатся ошибки, которые проявляются при определенных наборах входных данных; програм-

ма должна правильно реализовать предусмотренный алгоритм, должно быть реализовано защитное программирование и др. Следовательно, показатель надежности программы следует формировать на основе комплексной оценки свойств правильности, безошибочности и защищенности программы. Эта комплексная оценка позволяет определять безошибочность выполнения операций, функций, процессов, совокупности процессов, выполняемых системой задач.

Краеугольные камни построения надежного программного обеспечения (ПО) закладываются при разработке архитектуры ПО. Технология разработки архитектуры надежной программы включает в себя четыре группы методов и способов: предупреж¬дение ошибок, обнаружение ошибок, исправление ошибок и обеспечение устойчивости к ошибкам. При проектировании надежных программ рекомендуется применять широкий спектр хорошо апробированных правил и рекомендаций. К ним относятся, например, модульный подход, когда программы создают на основе существующих стандартов проектирования и кодирования, жестко типизированные языки программирования, технология структурного, а также объектно-ориентированного программирования и т.д.

Ограниченные возможности резервирования, средств оперативного обнаружения отказов, сбоев, ошибок в выполнении информационных процессов, ограниченные возможности комплекса «аппаратура – программы» – все это вызывает необходимость в развитии нестандартных технологий обеспечения надежности информационных систем. Одной из них является предложенная в данной книге технология адаптивной отказоустойчивости. Суть ее состоит в активном использовании естественной временной и структурной избыточности и в активном (и автоматическом) переназначении имеющихся вычислительных ресурсов не только для оперативной обработки

информации, но и для реализации наблюдаемости системы при ограниченных средствах контроля. Понятие «адаптивная отказоустойчивость информационных систем реального времени» заменено сокращенным понятием «активная защита». Активная защита (АЗ) предназначена для достижения требуемых уровней отказоустойчивости информационных систем (ИС) реального времени в условиях незначительного резерва времени, ограниченной эффективности средств обнаружения неисправностей составных вычислительных модулей, а также при условии, что объем резервного оборудования не должен превышать объема основного оборудования. Приведенные в книге результаты исследований показали, что с помощью активной защиты имеются реальные возможности добиться гораздо более высокого уровня надежности информационных систем, чем с помощью известных методов структурного и временного резервирования. Так, при количестве избыточных вычислительных модулей не более числа основных есть возможность в десятки и даже в сотни раз превысить уровень наработки на отказ системы, который оценивается при сколь угодно большом количестве резервных модулей, но при этом реальной эффективности средств обнаружения отказов. Технология адаптивной отказоустойчивости позволяет в ограниченных временных условиях при решении реальных задач осуществлять своевременное автоматическое обнаружение и устранение отказов и сбойных ошибок путем оперативной локализации неисправных вычислительных модулей и последующей автоматической реконфигурации системы с выводом из процесса функционирования отказавших модулей.

Предметом теории надежности, также как и теории функциональной безопасности, являются случайные процессы отказов и восстановлений, сбойные, программные ошибки, ошибки операторов. Однако при этом имеются следующие основные принципиальные различия:

Теория функциональной безопасности оперирует с очень редкими событиями (опасными функциональными отказами), частота возникновения которых в соответствии с оценками международных стандартов на 2 – 4 порядка меньше частоты отказов и ошибок, исследуемых теорией надежности. Эти редкие события остаются за рамками оценок теории надежности.

Теория надежности изучает влияние внутренних возмущающих факторов (отказов и ошибок) только на работу системы, без оценки их воздействия на внешнюю среду. Теория функциональной безопасности изучает воздействие отказов и ошибок в работе системы на объекты управления и в целом на внешнюю среду. При этом центральная задача состоит в исследовании вероятности и возможности парирования предусмотренными функциями безопасности опасных функциональных отказов.

Функции безопасности предназначены для оперативного обнаружения и устранения или блокирования отказов. Решение задач устранения или блокирования обнаруженных отказов не представляет особых трудностей. Проблема состоит в том, чтобы любым возможным способом своевременно и достоверно обнаружить очень редкие события опасных отказов. Философия теории надежности имеет принципиальное отличие – возможности средств обнаружения отказов объекта должны быть достаточными для сохранения приемлемого уровня готовности (технического использования) объекта. Эта позиция объясняется тем, что при введении средств контроля в объект без избыточности безотказность этого объекта снижается за счет отказов средств контроля. При этом, чем меньше объем средств контроля, тем ниже его эффективность. Поэтому стремятся оптимизировать объем (или время контроля) таким образом, чтобы обеспечить приемлемый уровень готовности объекта при допустимых потерях его безотказности.

При наличии избыточных ресурсов (структурных, временных, информационных, функциональных) налицо два диаметрально противоположных подхода к архитектуре объекта: с позиций теории надежности эти ресурсы используются в качества резерва для повышения безотказности и готовности объекта; с позиций теории функциональной безопасности эти избыточные ресурсы используются для обеспечения обнаружения опасных функциональных отказов путем построения многоканальных объектов с параллельной обработкой информации и аппаратными или программными средствами анализа или/и сравнения результатов. При этом допускается снижение уровня надежности объекта за счет отъема в интересах функциональной безопасности избыточных ресурсов. Каждое из этих научных направлений решают свои блоки задач, хотя и базируются на общем исходном материале – отказах и ошибках составных средств. Естественно, что уменьшение интенсивности отказов и ошибок объекта влечет за собой повышение надежности и в некоторой мере функциональной безопасности. Однако, учитывая, что кардинальных достижений в этом вопросе не следует ожидать, главным и определяющим инструментом обеспечения функциональной безопасности программно-аппаратного объекта должны быть функции безопасности, сформированные по результатам оценки рисков.

Совместное применение разных информационных технологий построения информационных систем управления ответственными и критически важными объектами создает естественные условия для построения двухуровневой системы управления. В отличие от ранее существующих ограничений на использование в одноуровневых системах управления только безопасных устройств, в двухуровневой системе возможно применение небезопасных систем.

Подтверждение соответствия информационной системы требованиям нормативных документов осуществляется на основе

обоснования безопасности. Этот результирующий документ включает оценки рисков и доказательство безопасности. В свою очередь, доказательство безопасности – это так называемый «паспорт» системы, содержащий политику построения надежной и безопасной информационной системы, программу работ по обеспечению этих свойств в системе и главное – это методы и способы практического доказательства безопасности. К ним относятся результаты стендовых и конструкторских испытаний, математического моделирования, экспертные оценки и, что особенно важно, результаты натурных испытаний. Практика – основа истины. Однако при существующих высоких требованиях к надежности и, особенно, безопасности информационных систем набрать достоверные данные натурных испытаний в течение даже жизненного цикла системы не представляется возможным. Кардинальное решение проблемы состоит в применении предложенного метода ускоренных натурных испытаний. Этот метод основывается на методе имитационного моделирования Монте-Карло. Для ускорения испытаний в нем также использован метод значимой выборки. Путем генерации сбойных ошибок и помех и привнесения их соответственно на объекты и каналы передачи информации реальной информационной системы создаются условия для проведения натурных испытаний этой системы на надежность и функциональную безопасность в короткие отрезки времени, которые адекватны длительному времени функционирования системы. Приведен пример ускоренных натурных испытаний на функциональную безопасность информационной системы управления технологическим процессом – системы диспетчерской централизации на железнодорожном транспорте.

Подтверждение соответствия программных средств информационных систем должно охватывать направления качества, функциональной безопасности, безопасности информации (включая

недекларированные возможности в программах), комплексной безопасности программ (в том числе защищенности от несанкционированного доступа). Приведенные в книге III основные положения методики испытаний качества и функциональной безопасности программных средств могут быть использованы при декларировании их соответствия требованиям Технических регламентов. Методика устанавливает типовую номенклатуру показателей качества и функциональной безопасности ПО, метрики показателей качества и модели оценивания ПО, методы тестирования и экспертизы модификаций ПО. В методике подтверждения соответствия программных средств по требованиям безопасности информации учтены не только требования ФСТЭК России [111], но и требования стандартов [8, 9, 55, 56, 57] в части обеспечения целостности и доступности информации. Подтверждение соответствия программного средства должно носить комплексный характер и должно убедить и пользователя и надзорные органы в том, что это программное средство выполнено не только качественно и функционально безопасно, в том, что оно не содержит недекларированных возможностей, но и в том, что оно защищено от несанкционированного доступа (в том числе от кибератак). С этой целью в составе процедур подтверждения соответствия должен быть предусмотрен аудит выполнения требований ФСТЭК России [88], а также анализ представленных документов по оценке рисков и доказательству безопасности программ. Именно совокупность этих доказательств и оценок позволяет получить полное и достаточно объективное представление о надежности и безопасности программных средств информационной системы.

Список литературы

1. **Шубинский, И. Б.** Структурная надежность информационных систем. Методы анализа [Текст] / И. Б. Шубинский. – М. : Журнал Надежность, 2012. – 296 с.

2. **Шубинский, И. Б.** Функциональная надежность информационных систем. Методы анализа [Текст] / И. Б. Шубинский. – М. : Журнал Надежность, 2012. – 296 с.

3. **Солодкая, М. С.** К единству социального и технического: проблемы и тенденции развития научных подходов к управлению [Текст] / М. С. Солодкая. – Оренбург : ДиМур, 1997. – 208 с.

4. **Мишарин, А. С.** Функциональная надежность информационно-управляющих систем на федеральном железнодорожном транспорте [Текст] / А. С. Мишарин, И. Б. Шубинский // Известия Российской академии наук. Теория и системы управления. – 2004. – №1. – С. 155–162.

5. **Кругликов, Р. И.** Избыточность как принцип программирующей деятельности головного мозга [Текст] /Р. И. Кругликов. // Вопросы философии. – 1984. – №9. – С. 86–94.

6. **IEC 62278:2002.** Railway applications – the specification and demonstration of reliability, availability, maintainability and safety (RAMS) [Text].

7. **DIN 40041:1990-12.** Zuverlassigkeit: Begriffe / Deutsche Elektrotechnische Kommission, Ausschuss Qualitatssicherung und angewandte Statistik [Text]. – Berlin: Deutsches Institut fur Normung e.V., 1990. – 19 s.

8. **EN 50129:2003.** Railway applications – Communication, signalling and processing systems – Safety related electronic systems for signaling [Text].

9. **EN 61511-1:2005.** Functional safety – Safety instrumented systems for the process industry sector – Part 1: Framework, definitions, system, hardware and software requirements [Text].

10. **ГОСТ Р 54505-2011.** Безопасность функциональная. Управление рисками на железнодорожном транспорте [Текст]. – Введ. 2012–08–01. – М. : Стандартинформ, 2012. – III, 40 с.

11. **ГОСТ Р 54504-2011.** Безопасность функциональная. Политика, Программа обеспечения безопасности. Доказательство безопасности объектов железнодорожного транспорта [Текст]. – Введ. 2012–08–01. – М. : Стандартинформ, 2013. – III, 28 с.

12. **Макиавелли, Н.** Рассуждения о первой декаде Тита Ливия [Текст] / Н. Макиавелли // Государь. Рассуждения о первой декаде Тита Ливия. О военном искусстве : сборник Н. Макиавелли ; пер. с ит. – Минск : Попурри, 2005. – 672 с.

13. **Макиавелли, Н.** О военном искусстве [Текст] / Н. Макиавелли // Государь. Рассуждения о первой декаде Тита Ливия. О военном искусстве : сборник Н. Макиавелли ; пер. с ит. – Минск : Попурри, 2005. – 672 с.

14. **Кузьмин, С. З.** Основы проектирования систем цифровой обработки радиолокационной информации [Текст] / С. З. Кузьмин. – М. : Радио и связь, 1986. – 352 с.

15. **Иыуду, К. А.** Надёжность, контроль и диагностика вычислительных машин и систем [Текст] / К. А. Иуыду. – М. : Высшая Школа, 1989. – 215 с.

16. **Уткин, Л. В.** Нетрадиционные методы оценки надежности информационных систем [Текст] /Л. В. Уткин, И. Б. Шубинский ; под ред. проф. И. Б. Шубинского. – С-Пб. : Любавич, 2000. – 192 с.

17. **Авиженис, А. А.** Отказоустойчивость – свойство, обеспечивающее работоспособность цифровых систем [Текст] / А. А. Авиженис ; пер. с англ. // Труды ИИЭР : сб. науч. тр. – 1978. – Т. 66. – № 10. – С. 5–25.

18. **Avizienis, A.** Dependability of computer systems [Text] / A.Avizienis, J-C Laprie, B. Randell / Fundamental concepts, terminology and examples. Technical report, LAAS – CNRS, October, 2000.

19. Активная защита от отказов управляющих модульных вычислительных систем [Текст] / И. Б. Шубинский [и др.] ; под ред. проф. И. Б. Шубинского. – С.Пб: Наука, 1993. – 284 с.

20. **Lamport, L.** The byzantine generals problem [Text] / L. Lamport, R. Shostak, M. Pease // ACM Trans. Prog. Lang. Sys. – 1982. – Vol. 4. – №3. – Pp. 382–401.

21. **Ammar, H.** A Comparative Analysis of Hardware and Software Reliability Engineering [Text] / H. Ammar, B. Cukis, A. Mili. // Annals of Software Engineering. – 2000. – Vol. 10. – № 1-4. – Pp. 103-150.

22. **Thorhuus, R.** Software Fault Injection Testing [Text] / R. Thorhuus. // Master of Science Thesis in Electronic System Design. – Stockholm, Sweden: KungligaTekniskaHцgskolan, 2000.

23. **Bieman, J.** Using Fault Injection to Test Software Recovery Code [Text] / J. Bieman. // Final report of a CASI FY95 Technology Transfer Grant. / Fort Collins, Colorado, USA: Colorado Advanced Software Institute, 1996.

24. **Han, S.** An Integrated Software Fault Injection Environment for Distributed Real-time Systems [Text] /S. Han, K. G. Shin, H. A. Rosenberg. // In Proc. 2nd Annual IEEE Int. Computer Performance and Dependability Symp. (IPDS'95). – Erlangen, Germany, 1995. – Pp. 204-213.

25. **Каган, Б. М.** Основы эксплуатации ЭВМ [Текст] / Б. М. Каган, И. Б. Мкртумян. – М. : Энергоатомиздат, 1988. – 432 с.

26. **Турута, Е. Н.** Организация распределения задач в вычислительных системах, обеспечивающая их отказоустойчивость [Текст] / Е. Н. Турута. // Автоматика и вычислительная техника. – 1985. – №1. – С. 5–14.

27. **Катсуки, Д.** Pluribus – отказоустойчивый операционный мультипроцессор [Текст] / Д. Катсуки, Э. С. Элсманн, У. Ф. Манн. // Труды ИИЭР : сб. науч. тр. – 1978. – Т. 66. – № 10. – С.49–58.

28. **Шубинский, И. Б.** Расчет надежности ЭВМ [Текст] / И. Б. Шубинский, Е. Н. Пивень. – Киев: Техника, 1979. – 232 с.

29. Bell System Technical Journal [Text]. – 1977. – Vol. 56. – № 2.

30. **ISO/IEC 27032:2012.** Информационные технологии. Методы обеспечения безопасности. Руководящие указания по кибербезопасности [Текст].

31. Проект ГОСТ Р МЭК 62443-1-1 Промышленные коммуникационные сети. Защищенность (кибербезопасность) сети и системы. Часть 1-1. Терминология, концептуальные положения и модели [Текст].

32. **ГОСТ 27.002-89.** Надежность в технике. Термины и определения [Текст]. – Введ. 1990–07–01. – М. : ИПК Издательство стандартов, 1990. – 37 с.

33. **Козлов, Б. А.** Справочник по расчету надежности аппаратуры радиоэлектроники и автоматики [Текст] / Б. А. Козлов, И. А. Ушаков. – М. : Сов. радио, 1975. – 472 с.

34. Надежность технических систем : Справочник [Текст] / Ю.К. Беляев [и др.] ; под ред. И. А. Ушакова. – М. : Радио и связь, 1985. – 608 с.

35. **Shubinsky, I. B.** On the Definition of Functional Reliability [Text] / I. B. Shubinsky, X. Sheabe // Electronic Journal Reliability: Theory & Applications. – 2012, Dec. – Vol. 7. – № 4. – P. 18.

36. **Shubinsky, I. B.** On the definition of functional reliability [Text] / I. B. Shubinsky, X. Sheabe // Proceedings of the ESREL 2013, Safety, Reliability and Risk Analysis: Beyond the Horizon. – Steenbergen et al. (Eds), 2014. – Taylor & Francis Group, London, ISBN 978-1-138-00123-7. – Pp. 3021-3027.

37. **Шубинский, И. Б.** Систематический подход к защите программного обеспечения от сбоев аппаратуры [Текст] / И. Б. Шубинский, Х. Шебе. // Надежность. – 2014. – № 3. – С. 96–106.

38. **Shubinsky, I. B.** A systematic approach for defense against transient failures [Text] / I. B. Shubinsky, X. Sheabe // Proceedings of ESREL 2014, Safety and Reliability: Methodology and Applications. – Nowakowski et al. (Eds), 2015. – ISBN 978-1-138-02681-0.

39. **Shubinsky, I. B.** Was sind die Folgen des Rebootenssicherer Computer? [Text] / I. B. Shubinsky, X. Sheabe, E. N. Rosenberg. // TechnischeZuverlдssigkeit. – 2009. – EntwicklungzuverlдssigerPro dukte, VDI-Berichte 2065. – S. 325–334.

40. **Shubinsky, I. B.** A short study on rebooting safe computers and the impact on safety [Text] / I. B. Shubinsky, X. Sheabe, E. N. Rosenberg // ESREL 2009, Proceedings Reliability, Risk and Safety. – Vol. 1. – Pp. 175-178.

41. **Зыль, С.** Штатные механизмы QNX Neutrino для обеспечения отказоустойчивости вычислительных систем жесткого реального времени [Электронный ресурс] / С. Зыль. – СТА. – 2009(3). – 118 с. – Режим доступа: http://www.cta.ru/cms/f/389405.pdf.

42. **Демьянов, А. В.** VxWorks операционная система внутри Интернет [Электронный ресурс] / А. В. Демьянов. – Режим доступа: http://www.asutp.ru/?p=600027.

43. **Фогелин, Д.** Реализация высокой готовности во встраиваемых системах [Электронный ресурс] / Д. Фогелин, Х. Квинг. – Режим доступа: http://www.asutp.ru/?p=600410.

44. **Жданов, А. А.** Операционные системы реального времени [Электронный ресурс] / А. А. Жданов // PCWeek. – 1999(8). 8/1999. – Режим доступа: http://www.asutp.ru/?p=600591.

45. **Древс, Ю. Г.** Системы реального времени: технические и программные средства [Текст] : учеб. пособие / Ю. Г. Древс. – М. : МИФИ, 2010. – 320 с.

46. **Авен, О. И.** Управление вычислительным процессом в ЭВМ (Алгоритмы и модели) [Текст] / О. И. Авен, Я. А. Коган. – М. : Энергия, 1978. – 240 с.

47. **Шубинский, И. Б.** Активная защита от отказов микропроцессорных вычислительных систем [Текст] / И. Б. Шубинский. – М. : Знание, 1987. – 60 с.

48. **Григорьев, В. Л.** Программное обеспечение микропроцессорных систем [Текст] / В. Л. Григорьев. – М. : Энергоатомиздат, 1983. – 92 с.

49. **Шубинский, И. Б.** Основы анализа сложных систем [Текст] : учеб. пособие / И. Б. Шубинский. – Л. : Министерство обороны СССР, 1986. – 256 с.

50. **Гнеденко, Б. В.** Введение в теорию массового обслуживания [Текст] / Б. В. Гнеденко, И. Н. Коваленко. – Киев : Наука, 1963.

51. **Додонов, А.Г.** Введение в теорию живучести вычислительных систем [Текст] / А. Г. Додонов, М. Г. Кузнецова, А. С. Горбачик. – Киев : Наукова думка, 1990. – 184 с.

52. **Гуляев, В.А.** Организация живучих вычислительных структур [Текст] / В. А. Гуляев, А. Г. Додонов, С. П. Пелехов. – Киев : Наукова думка, 1982. – 140 с.

53. **Пархоменко, П. П.** Основы технической диагностики: Оптимизация алгоритмов диагностирования, аппаратурные средства [Текст] / П. П. Пархоменко, Е. С. Согомонян. – М. : Энергия, 1981. – 319 с.

54. **Липаев, В. В.** Надежность программного обеспечения АСУ [Текст] / В. В. Липаев. – М. : Энергоиздат, 1989.

55. **EN 50128:2000.** Railway Applications – Communications, signaling and processing systems – Software for Railway Control and Protection Systems = Применения на железнодорожном транспорте – Программное обеспечение для систем управления и обеспечения безопасности на железнодорожном транспорте [Текст].

56. **IEC 62279:2002.** Railway Applications – Dependability for Guided Transport Systems. Part 2: Safety = Применения на железнодорожном транспорте – Согласованность для управляющих транспортных систем. Часть 2. Безопасность [Текст].

57. **ГОСТ Р МЭК 61508-2012.** Функциональная безопасность систем электрических, электронных, программируемых электронных связанных с безопасностью [Текст]. – Введ. 2013–08–01. – М. : Стандартинформ, 2014. – 98 с.

58. **ГОСТ 19.201-78.** Единая система программной документации. Техническое задание. Требования к содержанию и оформлению [Текст]. – Введ. 1980–01–01. – М. : ИПК Издательство стандартов, 1978. – 4 с.

59. **ГОСТ 19.202-78.** Единая система программной документации. Спецификация. Требования к содержанию и оформлению [Текст]. – Введ. 1980–01–01. – М. : ИПК Издательство стандартов, 1978. – 4 с.

60. **ГОСТ 19.401-78.** Единая система программной документации. Текст программы. Требования к содержанию и оформлению [Текст]. – Введ. 1980–01–01. – М. : ИПК Издательство стандартов, 1978. – 1 с.

61. **ГОСТ 19.501-78.** Единая система программной документации. Формуляр. Требования к содержанию и оформлению [Текст]. – Введ. 1980–01–01. – М. : ИПК Издательство стандартов, 1978. – 5 с.

62. Федеральный закон от 27 декабря 2002 г. № 184-ФЗ «О техническом регулировании» [Электронный ресурс] // СПС КонсультантПлюс.

63. **ГОСТ Р МЭК 61513-2011.** Атомные станции. Системы контроля и управления, важные для безопасности. Общие требования [Текст]. – Введ. 2012–01–01. – М. : Стандартинформ, 2012. – 82 с.

64. **Сапожников, В.В.** Сертификация и доказательство безопасности систем железнодорожной автоматики [Текст] /

В. В. Сапожников, Вл. В. Сапожников, В. И. Талалаев [и др.] ; под ред. Вл. В. Сапожникова. – М. : Транспорт, 1997. – 288 с.

65. **Cullyer, W. J.** Safety-Critical Control Systems [Text] / W. J. Cullyer. // Computer and Control Engineering Journal. – 1991. – Vol. 2. – № 5. – Pp. 202–210.

66. **Сапожников, В.В.** Методы построения безопасных микроэлектронных систем железнодорожной автоматики [Текст] / В. В. Сапожников, Вл. В. Сапожников, Х. А. Христов, Д. В. Гавзов ; под ред. Вл. В. Сапожникова. – М. : Транспорт, 1995. – 272 с.

67. **Швир, В.** Надежность электронных схем в устройствах СЦБ [Текст] / В. Швир // Железные дороги мира. – 1986. – № 1. – С. 59-67.

68. **Коваленко, И. Н.** Анализ редких событий при оценке эффективности и надежности систем [Текст] / И. Н. Коваленко. – М. : Сов. радио, 1980. – 208 с.

69. **Коваленко, И. Н.** Методы расчета высоконадежных систем [Текст] / И. Н. Коваленко, Н. Ю. Кузнецов. – М. : Радио и связь, 1988. – 186 с.

70. **Braband, J.** A practical guide to safety analysis methods [Text] / J. Braband // SIGNAL + DRAHT. – 2001. – Vol. 93. – № 9. – Pp. 41–44.

71. **Braband, J.** Systematic Process for the Definition of Safety Targets for Railway Signalling Applications [Text] / J. Braband, A. Lennartz // SIGNAL + DRAHT. – 1999. – № 9. – Pp. 53–57.

72. **Def Stan 00-55.** Safety Management Requirements for Defence System [Text] / Ministry of Defence (UK). – 1996.

73. **Def Stan 00-56.** Requirements for Safety Related Software in Defence Equipment [Text] / Ministry of Defence (UK). – 1997.

74. **Hirao, Y.** Safety technologies and management of railway signalling in Japan [Text] / Y. Hiaro, I. Watanabe // SIGNAL + DRAHT. – 2000. – № 5.

75. Federal German Railways Office (EBA), Мь 8004. Anweisungzu den technischen Anforderungenfьr die Zulassung von Sicherungsanlagen = Федеральное управление Германских железных дорог (EBA). Документ Мь 8004. Руководящие указания по техническим требованиям к сертификации систем обеспечения безопасности [Текст].

76. **Шалягин, Д. В.** Надежность и безопасность железнодорожной автоматики и телемеханики [Текст] / Д. В. Шалягин, И. Б. Шубинский // Автоматика, связь, информатика. – 2005. – № 2. – С. 23-26.

77. **Абрамов, В.М.** Характеристики надежности и функциональной безопасности структур железнодорожной автоматики [Текст] /В. М. Абрамов, Б. Д. Никифоров, Д. В. Шалягин // Вестник ВНИИЖТ. – 2006. – №1. – С. 32–38.

78. **Шубинский, И. Б.** Методы и модели функциональной безопасности технических систем [Текст] / И. Б. Шубинский, Е. Н. Розенберг // Сборник трудов ВНИИАС (2005 г.). – М. : ВНИИАС, 2005. – 202 с.

79. **ГОСТ Р МЭК 60880-2010.** Атомные электростанции. Системы контроля и управления, важные для безопасности. Программное обеспечение компьютерных систем, выполняющих функции категории А [Текст]. – Введ. 2012–01–01. – М. : Стандартинформ, 2011. – 90 с.

80. **ГОСТ Р 51904-2002.** Программное обеспечение встроенных систем. Общие требования к разработке и документированию [Текст]. – Введ. 2003–07–01. – М. : ИПК Издательство стандартов, 2002. – 67 с.

81. **Schabe, H.** The Safety Philosophy behind CENELEC Rails Standards [Text] / H. Shabe // Proceedings ESREL 2002, Lyon, March 19-21. – 2002. – Pp. 788-790.

82. **Гайен, И.-Т.** Неправильное и правильное понимание принципов обеспечения функциональной безопасности [Текст] / И.-Т. Гайен, Х. Шебе // Надежность. – 2009. – № 4(32). – С. 63–74.

83. **Gulker, J.** Physical Principles of Safety [Text] / J. Gulker, H. Shaebe // Safety and Reliability for Management Risk : Proceedings of the ESREL 2006, Estoril, Portugal, 18–22 Sept. 2006. – 2006. – Vol. 2. – Pp. 1045–1050.

84. **ГОСТ Р 51901-2002.** Управление надежностью. Анализ риска технологических систем [Текст]. – Введ. 2003–09–01. – М. : ИПК Издательство стандартов, 2002. – 28 с.

85. **Shubinsky, I. B.** Topological semi-Markov method for calculation of stationary parameters of Reliability and Functional Safety of technical systems [Text] / I. B. Shubinsky, A. M. Zamyshlyaev // Reliability: Theory & Applications. – 2012. – Pp. 12-22.

86. **Шубинский, И. Б.** Топологический метод и алгоритм определения стационарных показателей надежности технических систем [Текст] / И. Б. Шубинский // Надежность и контроль качества. – 1984. – №5. – С. 3–10.

87. **Месарович, М.** Теория иерархических многоуровневых систем [Текст] / М. Месарович, Д. Мако, И. Тахакара. – М. : Мир, 1973. – 344 стр.

88. Приказ ФСТЭК России от 14 марта 2014 г. № 31 «Об утверждении требований к обеспечению защиты информации в автоматизированных системах управления производственными и технологическими процессами на критически важных объектах, потенциально опасных объектах, а также объектах, представляющих повышенную опасность для жизни и здоровья людей и для окружающей природной среды» [Электронный ресурс] // СПС КонсультантПлюс.

89. **ТР ТС 001/2011.** Технический регламент ТС. О безопасности железнодорожного подвижного состава [Электронный ресурс] // СПС КонсультантПлюс.

90. **ТР ТС 002/2011.** Технический регламент ТС. О безопасности высокоскоростного железнодорожного транспорта [Электронный ресурс] // СПС КонсультантПлюс.

91. **ТР ТС 003/2011.** Технический регламент ТС. О безопасности инфраструктуры железнодорожного транспорта [Электронный ресурс] // СПС КонсультантПлюс.

92. **Васильев, Д. В.** Ускоренное статистическое моделирование систем управления [Текст] / Д. В. Васильев, О. Ю. Сабинин. – Л.: Энергоатомиздат. Ленинградское отд-ние, 1987. – 136 с.

93. **Kumamoto, H.** Efficient evaluation of system reliability by Monte Carlo method [Text] / H. Kumamoto, T. Tanaka, K. Inoue // IEEE Trans. Reliab. – 1977. – Vol. R-26. – № 12. – Pp. 311–315.

94. **Уткин, Л. В.** Разработка системы ускоренного моделирования для исследования систем передачи информации [Текст] / Л. В. Уткин // Известия ЛЭТИ, Вып.418. – Л. : ЛЭТИ, 1989. – С. 58–62.

95. **Дмитриев, О. Ф.** Ускоренные испытания по проверке основных характеристик СПД [Текст] / О. Ф. Дмитриев // Построение и анализ передачи информации. – М. : Наука, 1980. – С. 102–106.

96. **Ермаков, С. М.** Метод Монте-Карло и смежные вопросы [Текст] / С. М. Ермаков. – М. : Наука, 1975. – 472 с.

97. **Hahn, P. M.** Developments in the theory and application of importance sampling [Text] / P. M. Hahn, M. C. Jeruchim // IEEE Trans. Commun. – 1987. – Vol. COM-35. – № 7. – Pp. 706–714.

98. **Jeruchim, M. C.** An experimental investigation of conventional and efficient importance sampling [Text] / M. C. Jeruchim, P. M. Hahn // IEEE Trans. Commun. – 1989. – Vol. COM-37. – № 6. – Pp. 578–587.

99. **Шеховцов, О. И.** Применение метода значимой выборки для вычисления распределения кратностей ошибок в дискретном канале связи [Текст] / О. И. Шеховцов, А. А. Шитов, Л. В. Уткин // Специальная техника средств связи : Серия Общетехническая. – 1988. – № 7.

100. **Beaulieu, N. C.** A composite importance sampling technique for digital communication system simulation [Text] / N. C. Beaulieu // IEEE Trans. Commun. – 1990. – Vol. COM-38. – № 4. – Pp. 393–396.

101. **Davis, B. R.** An improved importance sampling method for digital communication system simulations [Text] / B. R. Davis // IEEE Trans. Commun. – 1986. – Vol. COM-34. – № 7. – Pp. 715–719.

102. **Blue, J.L.** Faster Monte Carlo simulations [Text] / J.L. Blue, I.M. Beichl, F. Sullivan // Phys. Rev. – 1995. – E. 51(2). – Pp. 867-868.

103. **Lank, G. W.** Theoretical aspects of importance sampling applied to false alarms [Text] / G. W. Lank // IEEE Trans. Inform. Theory. – 1983. – Vol. IT-29. – № 1. – Pp. 73–82.

104. **Orsak, G.** On the theory of importance sampling applied to the analysis of detection systems [Text] / G. Orsak, B. Aazhang // IEEE Trans. Commun. – 1989. – Vol. COM-37. – № 4. – Pp. 332–339.

105. **Wang, Q.** On the application of importance sampling to BER estimation in the simulation of digital communication systems [Text] / Q. Wang, V. K. Bhargava // IEEE Trans. Commun. – 1987. – Vol. COM-35. – № 11. – Pp. 1231–1234.

106. **ГОСТ Р ИСО/МЭК 15408-2012.** Методы и средства обеспечения безопасности. Критерии оценки безопасности информационных технологий [Текст] : Часть 1. Введение и общая модель ; Часть 2. Защита функциональных требований ; Часть 3. Защита требований к качеству. – Введ. 2013–12–01. – М. : Стандартинформ, 2014. – 340 с.

107. **ГОСТ Р ИСО/МЭК 13335-2006.** Информационная технология. Методы и средства обеспечения безопасности [Текст] : Часть 1. Концепция и модели менеджмента безопасности информационных и телекоммуникационных технологий ; Часть 3. Методы менеджмента безопасности информационных техноло-

гий ; Часть 4. Выбор защитных мер. Часть 5. Руководство по менеджменту безопасности сети. – Введ. 2007–06–01. – М. : Стандартинформ, 2007. – 163 с.

108. **ISO 10181:1996.** Информационные технологии. Взаимодействие открытых систем. Основы безопасности для открытых систем [Текст] : Часть 1. Обзор ; Часть 2. Основы аутентификации ; Часть 3. Структура обеспечения контроля за доступом ; Часть 4. Основы подтверждения авторства ; Часть 5. Основы конфиденциальности ; Часть 6. Основы целостности ; Часть 7. Аудит защиты и основы аварийной сигнализации.

109. Проект ГОСТ Р МЭК 62443-2-1-2013 Промышленные коммуникационные сети. Защищенность (кибербезопасность) сети и системы [Текст] : Часть 2-1. Составление программы обеспечения защищенности (кибербезопасности) системы управления и промышленной автоматизации.

110. Руководящий документ. Защита от несанкционированного доступа к информации [Текст] : Часть 1. Программное обеспечение средств защиты информации. Классификация по уровню контроля отсутствия недекларированных возможностей. Утв. решением председателя Государственной технической комиссии при Президенте Российской Федерации от 4 июня 1999 г. № 114.

111. **ГОСТ 19.301-79.** Единая система программной документации. Программа и методика испытаний. Требования к содержанию и оформлению [Текст]. – Введ. 1981–01–01. – М. : ИПК Издательство стандартов, 1998. – 3 с.

112. Проект ГОСТ Р МЭК 62443-3-2013. Промышленные коммуникационные сети. Защищенность (кибербезопасность) сети и системы [Текст] : Часть 3. Защищенность (кибербезопасность) промышленного процесса измерения и управления.

Содержание