ФЕДЕРАЛЬНОЕ АГЕНТСТВО

ПО ТЕХНИЧЕСКОМУ РЕГУЛИРОВАНИЮ И МЕТРОЛОГИИ

НАЦИОНАЛЬНЫЙ СТАНДАРТ РОССИЙСКОЙ ФЕДЕРАЦИИ

ГОСТ Р 53195.5— 2010

БЕЗОПАСНОСТЬ ФУНКЦИОНАЛЬНАЯ СВЯЗАННЫХ С БЕЗОПАСНОСТЬЮ ЗДАНИЙ И СООРУЖЕНИЙ СИСТЕМ

Часть 5

Меры по снижению риска, методы оценки

Издание официальное

Москва
Стандартинформ
2011

Предисловие

Цели и принципы стандартизации в Российской Федерации установлены Федеральным законом от 27 декабря 2002 г. № 184-ФЗ «О техническом регулировании», а правила применения национальных стандартов Российской Федерации — ГОСТ Р 1.0 — 2004 «Стандартизация в Российской Федерации. Основные положения»

Сведения о стандарте

1 РАЗРАБОТАН Университетом комплексных систем безопасности и инженерного обеспечения

2 ВНЕСЕН Техническим комитетом по стандартизации ТК 439 «Средства автоматизации и системы управления» при поддержке Технического комитета по стандартизации ТК 465 «Строительство»

3 УТВЕРЖДЕН И ВВЕДЕН В ДЕЙСТВИЕ Приказом Федерального агентства по техническому регулированию и метрологии от 21 декабря 2010 г. № 821-ст

4 В настоящем стандарте использованы основные нормативные положения следующих международных стандартов:

МЭК 61508-4:2010 «Функциональная безопасность систем электрических, электронных, программируемых электронных, связанных с безопасностью. Часть 4. Термины, определения, сокращения» (IEC 61508-4:2010 «Functional safety of electrical/ electronic/ programmable electronic safety-related systems — Part 4: Definitions and abbreviations»);

МЭК 61508-7:2010 Функциональная безопасность систем электрических, электронных, программируемых электронных, связанных с безопасностью. Часть 7. Обзор методов и средств (IEC 61508-7:2010 «Functional safety of electrical/ electronic/ programmable electronic safety-related systems — Part 7: Overview of techniques and measures»);

Руководство ИСО/МЭК 51:1999 Аспекты безопасности. Руководящие указания по включению их в стандарты (ISO/IEC Guide 51:1999 «Safety aspects — Guidelines for their inclusion in standards»)

5 ВВЕДЕН ВПЕРВЫЕ

Информация об изменениях к настоящему стандарту публикуется в ежегодно издаваемом информационном указателе «Национальные стандарты», а текст изменений и поправок — в ежемесячно издаваемых информационных указателях «Национальные стандарты». В случае пересмотра (замены) или отмены настоящего стандарта соответствующее уведомление будет опубликовано в ежемесячно издаваемом информационном указателе «Национальные стандарты». Соответствующая информация, уведомление и тексты размещаются также в информационной системе общего пользования — на официальном сайте Федерального агентства по техническому регулированию и метрологии в сети Интернет

Содержание

Введение

Современные здания и сооружения — объекты капитального строительства — представляют собой сложные системы, включающие в свой состав систему конструкций и ряд систем в разных сочетаниях, в том числе инженерные системы жизнеобеспечения, реализации технологических процесссов, энерго-, ресурсосбережения, безопасности и другие системы. Эти системы взаимодействуют друг с другом, с внешней и внутренней средами.

Объекты капитального строительства жестко привязаны к местности. Рабочие характеристики зданий, сооружений и входящих в них систем могут быть реализованы, проверены и использованы только в том месте, в котором объекты построены и системы установлены.

Безопасность зданий и сооружений обеспечивается применением совокупности мер, мероприятий и средств снижения риска причинения вреда до уровня приемлемого риска и поддержания его в течение периода эксплуатации или использования этих объектов. К средствам снижения риска относятся системы, связанные с безопасностью зданий и сооружений. Эти системы, состоящие из электрических и/или электронных компонентов, и/или программируемых электронных компонентов, в течение многих лет используются для выполнения функций безопасности. Для решения задач безопасности зданий и сооружений во все бóльших объемах используются программируемые электронные (компьютерные) системы.

В настоящем стандарте установлены цели основных методов/средств, рекомендованных к применению в ГОСТ Р 53195.3 и ГОСТ Р 53195.4 для анализа и снижения риска, достижения и поддержания необходимого уровня функциональной безопасности аппаратных средств (АС) и программного обеспечения (ПО) электрических, электронных, программируемых электронных (E/E/PE) связанных с безопасностью зданий и сооружений систем (СБЗС-систем) на различных стадиях их жизненного цикла, а также для оценки соответствия систем требованиям безопасности в рамках области применения ГОСТ Р 53195.1, ГОСТ Р 53195.2, ГОСТ Р 53195.3 и ГОСТ Р 53195.4. В нем приведены краткие описания указанных методов/средств, а также даны ссылки на источники, содержащие их полные описания.

Настоящий стандарт входит в комплекс стандартов с наименованием «Безопасность функциональная связанных с безопасностью зданий и сооружений систем» и является пятым стандартом этого комплекса «Часть 5. Меры по снижению риска, методы оценки». Другие стандарты, входящие в этот комплекс:

Часть 1. Основные положения;

Часть 2. Общие требования;

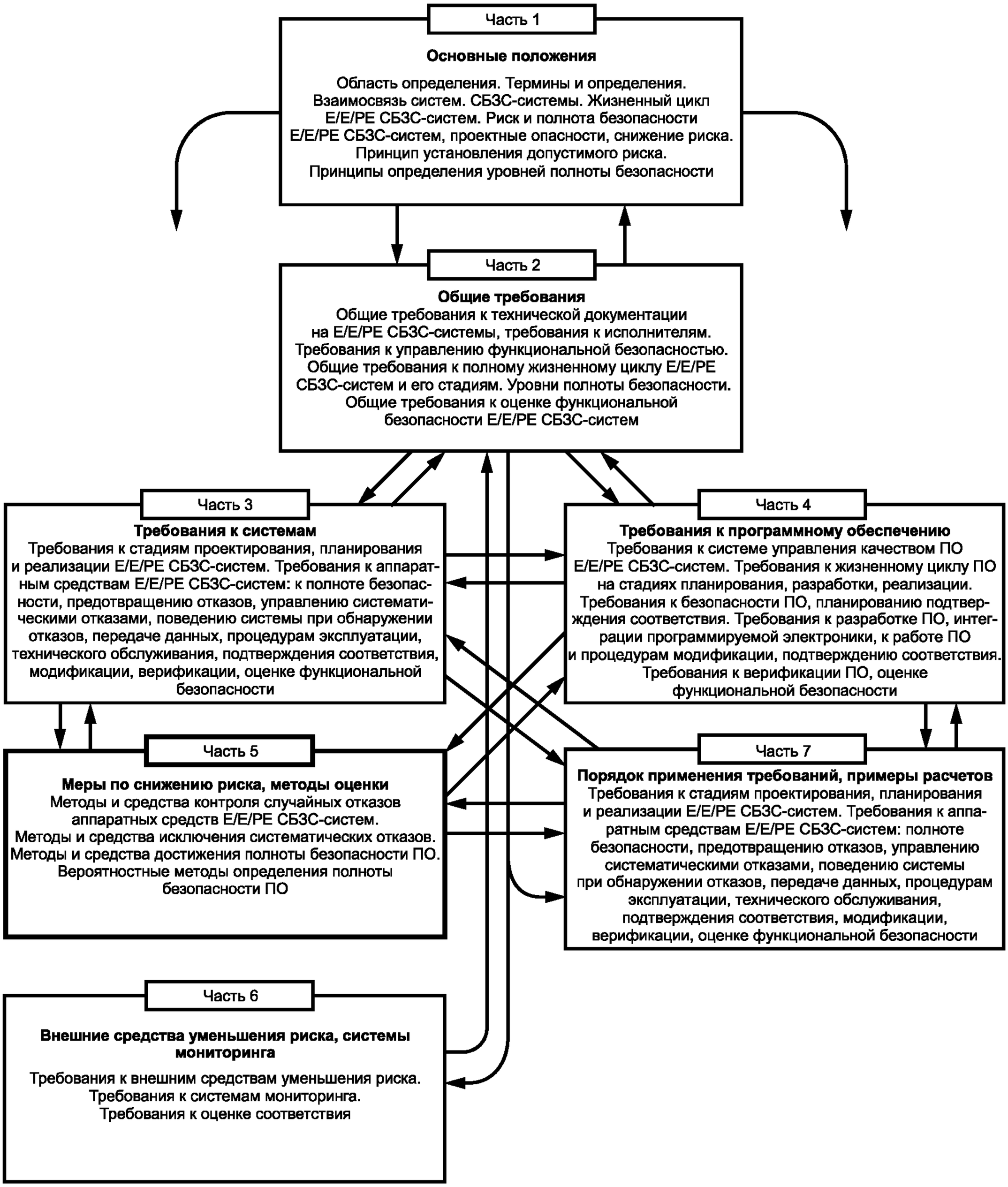
Часть 3. Требования к системам;

Часть 4. Требования к программному обеспечению;

Часть 6. Внешние средства уменьшения риска, системы мониторинга;

Часть 7. Порядок применения требований к системам и примеры расчетов.

Структура комплекса стандартов приведена ниже.


Часть 1
Основные положения
Область определения. Термины и определения. Взаимосвязь систем. СБЗС-системы. Жизненный цикл E/E/PE СБЗС-систем. Риск и полнота безопасности E/E/PE СБЗС-систем, проектные опасности, снижение риска. Принцип установления допустимого риска. Принципы определения уровней полноты безопасности
Часть 2
Общие требования
Общие требования к технической документации на E/E/PE СБЗС-системы, требования к исполнителям. Требования к управлению функциональной безопасностью. Общие требования к полному жизненному циклу E/E/PE СБЗС-систем и его стадиям. Уровни полноты безопасности. Общие требования к оценке функциональной безопасности E/E/PE СБЗС-систем
Часть 3
Требования к системам
Требования к стадиям проектирования, планирования и реализации E/E/PE СБЗС-систем. Требования к аппаратным средствам E/E/PE СБЗС-систем: к полноте безопасности, предотвращению отказов, управлению систематическими отказами, поведению системы при обнаружении отказов, передаче данных, процедурам эксплуатации, технического обслуживания, подтверждения соответствия, модификации, верификации, оценке функциональной безопасности
Часть 4
Требования к программному обеспечению
Требования к системе управления качеством ПО E/E/PE СБЗС-систем. Требования к жизненному циклу ПО на стадиях планирования, разработки, реализации. Требования к безопасности ПО, планированию подтверждения соответствия. Требования к разработке ПО, интеграции программируемой электроники, к работе ПО и процедурам модификации, подтверждению соответствия. Требования к верификации ПО, оценке функциональной безопасности
Часть 5
Меры по снижению риска, методы оценки
Методы и средства контроля случайных отказов аппаратных средств E/E/PE СБЗС-систем. Методы и средства исключения систематических отказов. Методы и средства достижения полноты безопасности ПО. Вероятностные методы определения полноты безопасности ПО
Часть 7
Порядок применения требований, примеры расчетов
Требования к стадиям проектирования, планирования и реализации E/E/PE СБЗС-систем. Требования к аппаратным средствам E/E/PE СБЗС-систем: полноте безопасности, предотвращению отказов, управлению систематическими отказами, поведению системы при обнаружении отказов, передаче данных, процедурам эксплуатации, технического обслуживания, подтверждения соответствия, модификации, верификации, оценке функциональной безопасности
Часть 6
Внешние средства уменьшения риска, системы мониторинга
Требования к внешним средствам уменьшения риска. Требования к системам мониторинга. Требования к оценке соответствия

НАЦИОНАЛЬНЫЙ СТАНДАРТ РОССИЙСКОЙ ФЕДЕРАЦИИ

БЕЗОПАСНОСТЬ ФУНКЦИОНАЛЬНАЯ СВЯЗАННЫХ С БЕЗОПАСНОСТЬЮ ЗДАНИЙ И СООРУЖЕНИЙ СИСТЕМ

Часть 5

Меры по снижению риска, методы оценки

Functional safety of building/erection safety-related systems.
Part 5. Techniques and measures on risk reduction, estimation methods

Дата введения — 2012 — 01 — 01

1 Область применения

Настоящий стандарт распространяется на связанные с безопасностью зданий и сооружений системы (далее — СБЗС-системы), аппаратные средства (далее — АС) и/или программное обеспечение (далее — ПО), являющиеся частями СБЗС-системы либо используемые для разработки СБЗС-систем в рамках областей применения ГОСТ Р 53195.1, ГОСТ Р 53195.2, ГОСТ Р 53195.3 и ГОСТ Р 53195.4.

Настоящий стандарт применяется совместно со стандартами ГОСТ Р 53195.1, ГОСТ Р 53195.2, ГОСТ Р 53195.3 и ГОСТ Р 53195.4.

Настоящий стандарт устанавливает основные методы/средства, используемые для выполнения требований ГОСТ Р 53195.3 и ГОСТ Р 53195.4, и методы оценки соответствия.

Настоящий стандарт содержит краткие описания методов/средств, рекомендуемых в ГОСТ Р 53195.3 и ГОСТ Р 53195.4 и применяемых на различных стадиях жизненных циклов СБЗС-систем, их АС и ПО для снижения рисков, а также ссылки на источники с полным описанием этих методов/средств.

Примечание — Под «методами/средствами» в настоящем стандарте понимаются методы и/или средства. В большинстве методов/средств, описанных в приложениях А, Б, В и Г, метод состоит в применении того или иного аппаратного, программного или аппаратно-программного средства или средств, в применении логических или математических действий (которые выполняются с использованием средств информатики и математики). В отдельных случаях рассматриваются методы или средства в чистом виде.

Настоящий стандарт не распространяется на одиночные СБЗС-системы, способные осуществить необходимое снижение риска и требуемая полнота безопасности которых ниже самого низкого уровня полноты безопасности (SIL1), определенного в таблицах 1 и 2 ГОСТ Р 53195.2. Он не распространяется также на здания и сооружения, оснащенные такими системами или не имеющие никаких связанных с безопасностью систем.

2 Нормативные ссылки

В настоящем стандарте использованы нормативные ссылки на следующие стандарты:

ГОСТ Р ИСО 9000—2005 Системы менеджмента качества. Основные положения и словарь

ГОСТ Р ИСО 9001—2008 Системы менеджмента качества. Требования

ГОСТ Р ИСО 10006—2005 Системы менеджмента качества. Руководство по менеджменту качества при проектировании

ГОСТ Р ИСО/МЭК 16085–2007 Менеджмент риска. Применение в процессах жизненного цикла систем и программного обеспечения

Издание официальное

ГОСТ Р 51700—2000 Совместимость технических средств электромагнитная. Технические средства, подключаемые к симметричным линиям. Параметры асимметрии относительно земли. Схемы измерений

ГОСТ Р 51904—2002 Программное обеспечение встроенных систем. Общие требования к разработке и документированию

ГОСТ Р 53195.1—2008 Безопасность функциональная связанных с безопасностью зданий и сооружений систем. Часть 1. Основные положения

ГОСТ Р 53195.2—2008 Безопасность функциональная связанных с безопасностью зданий и сооружений систем. Часть 2. Общие требования

ГОСТ Р 53195.3—2009 Безопасность функциональная связанных с безопасностью зданий и сооружений систем. Часть 3. Требования к системам

ГОСТ Р 53195.4—2010 Безопасность функциональная связанных с безопасностью зданий и сооружений систем. Часть 4. Требования к программному обеспечению

ГОСТ Р МЭК 61160—2006 Менеджмент риска. Формальный анализ проекта

ГОСТ 27.310—95 Надежность в технике. Анализ видов, последствий и критичности отказов. Основные положения

ГОСТ 13661—92 Совместимость технических средств электромагнитная. Пассивные помехоподавляющие фильтры и элементы. Методы измерения вносимого затухания

ГОСТ 16962.2—90 Изделия электротехнические. Методы испытаний на стойкость к механическим внешним воздействующим факторам

ГОСТ 30382—95 Совместимость технических средств электромагнитная. Дроссели помехоподавляющие. Общие технические условия

Примечание — При пользовании настоящим стандартом целесообразно проверить действие ссылочных стандартов в информационной системе общего пользования — на официальном сайте Федерального агентства по техническому регулированию и метрологии в сети Интернет или по ежегодно издаваемому информационному указателю «Национальные стандарты», который опубликован по состоянию на 1 января текущего года, и по соответствующим ежемесячно издаваемым информационным указателям, опубликованным в текущем году. Если ссылочный стандарт заменен (изменен), то при пользовании настоящим стандартом следует руководствоваться заменяющим (измененным) стандартом. Если ссылочный стандарт отменен без замены, то положение, в котором дана ссылка на него, применяется в части, не затрагивающей эту ссылку.

3 Термины и определения

В настоящем стандарте применены термины по ГОСТ Р 53195.1, ГОСТ Р 53195.2, ГОСТ Р 53195.3 и ГОСТ Р 53195.4, а также приведенные ниже термины с определениями.

3.1 **антивалентные сигналы** (antivalent signals): Два сигнала с одинаковым информационным содержанием, передаваемые по каналам связи в инверсной форме (аналоговые сигналы — в противофазе, цифровые сигналы — с инверсией 0 в 1 или наоборот).

3.2 **константная неисправность** (stuck-at fault): Неисправность аппаратного средства, вызванная переходом элемента устройства в одно из неизменяемых состояний, например, при «залипании» контактов реле.

3.3 **константный отказ** (stuck-at failure): Отказ аппаратного средства и/или программного обеспечения, приводящий к переходу аппаратного средства в одно из неизменяемых состояний и/или выдаче на выходе неизменяемых данных или неизменяемой команды.

3.4 **постепенный отказ** (drift failure): Отказ аппаратного средства из-за постепенного выхода его характеристик за допустимые пределы.

3.5 **самоустраняющийся отказ** (transient failure): Отказ, обусловленный переходными процессами, устраняющийся по их завершении.

3.6 **условная тревога** (conditional alarm): Состояние, близкое к тревожному, но еще не влекущее опасных последствий.

Примечание — Термин относится к СБЗС-системам, в которых предусмотрено ступенчатое реагирование на постепенно развивающиеся тревожные события.

3.7 **чрезвычайное действие** (emergency action): Действие, требующее выполнения при возникновении чрезвычайной ситуации для снижения риска причинения вреда.

4 Обозначения и сокращения

В настоящем стандарте приняты следующие обозначения и сокращения:

АС — аппаратное(ые) средство(а);

Е/Е/РЕ — электрический(ая, ое), электронный(ая, ое), программируемый(ая, ое) электронный(ая, ое) — в отношении модуля, устройства или системы;

ИС — интегральная(ые) микросхема(ы);

ОЗУ — оперативное запоминающее устройство (устройство памяти с произвольным доступом);

ОКС — Общероссийский классификатор стандартов;

ПЗУ — постоянное запоминающее устройство;

ПЛК — программируемый логический контроллер;

ПО — программное обеспечение;

ППЗУ — перепрограммируемое постоянное запоминающее устройство;

САПР — система автоматизированного проектирования;

СБЗС-система — система, связанная с безопасностью зданий и сооружений;

УО — управляемое оборудование;

ADA — язык программирования для встраиваемых систем, разработанный в 1979—1980 годах в США и названный в честь Ады Лавлэйс;

ADT — обозначение данных абстрактного типа (от англ. *abstract data type*);

CCS — наименование метода/средства расчета соединяющихся систем (от англ. *calculus of communicating system*);

CHAZOP — наименование метода/средства обеспечения безопасности и работоспособности систем управления (от англ. *control hazards operability*);

CHAZOPs — наименование метода/средства анализа безопасности работы компьютеров (от англ. *computer hazardous operation analysis*);

CIRCAL — наименование метода/средства расчета критических цепей (от англ. *circuit calculus*);

CORE — наименование метода/средства выражения контролируемых требований (от англ. *controlled requirements expression*);

CRC — циклический избыточный код коррекции ошибок (от англ. *cyclic redandency check*);

CSP — наименование метода/средства описания последовательных коммуникационных процессов (от англ. *communicating sequential processes*);

EDC — код обнаружения/коррекции ошибок;

E/E/PES — международное наименование электрической, электронной, программируемой электронной связанной с безопасностью системы;

FMEA — обозначение процедуры анализа типа отказа и его последствий (от англ. *procedure for failure mode and effects analysis*);

FTA — метод анализа на основе дерева отказов (от англ. *fault tree analysis*);

HAZOP — наименование метода/средства анализа безопасности и работоспособности (от англ. *hazard and operability*);

HOL — наименование языка логики высшего порядка (от англ. *higher-order logic*);

INMOS — наименование английской фирмы, специализирующейся на производстве транспьютеров;

JSD — наименование структурного метода разработки программных систем Джексона (от англ. *Jackson structured development*);

LCSAJ — обозначение последовательности линейного кода и перехода, применяемой при тестировании ПО (от англ. *linear code sequence and jump*);

LOTOS — наименование языка для описания спецификаций, упорядоченных во временно́й области (от англ. *language for temporal ordering specification*);

MASCOT — наименование модульного подхода к проектированию, работе и тестированию программного обеспечения (от англ. *modular approach to software construction, operation and test*);

MCDC — обозначение охвата решения модифицированными условиями (от англ. *modified condition decision coverage*);

MTBF — обозначение среднего времени наработки на отказ (от англ. *mean time between failures*);

OBJ — наименование языка для алгебраического описания спецификаций;

OCCAM — язык параллельного программирования высокого уровня, используемый для транспьютеров;
OMT — обозначение методологии объектного моделирования (от англ. *object modeling technique*);
PE — международное обозначение «программируемый(ая,ое) электронный(ая,ое)» — в отношении модуля, устройства или системы;
PROM — наименование программируемого постоянного запоминающего устройства;
RAID — наименование системы организации избыточного массива памяти с использованием недорогих накопителей на дисках (от англ. *redundant array of inexpensive disks);*
RAM — обозначение запоминающего устройства с произвольным доступом;
ROM — обозначение постоянного запоминающего устройства;
SA/SD — обозначение метода структурного проектирования программных систем на основе структурного анализа (от англ. *structured analysis/structured design*);
SADT — наименование метода/средства структурного анализа и проектирования (от англ. *structured analysis and design technique*);
SDL — наименование языка описаний и спецификаций (от англ. *specification-and-description language*);
SIL — международное обозначение уровня полноты безопасности (от англ. *safety integrity level*);
SOM — наименование технологии Ай-Би-Эм для компонентных архитектур (от англ. *system object model*);
VDM — наименование одного из методов разработки компьютерных систем на основе формального языка;
VDM++ — наименование расширенной версии метода VDM;
VDM-SL — обозначение формального языка для описания спецификаций, разрабатываемых с использованием метода VDM;
XOR — обозначение логической операции «исключающее ИЛИ»;
Z — обозначение нотации языка для описания спецификаций последовательных систем.

5 Меры (методы/средства) по снижению риска

5.1.1 Основными мерами по снижению риска являются:
- контроль случайных отказов АС E/E/PE СБЗС-систем;
- исключение систематических отказов на различных стадиях жизненных циклов СБЗС-систем;
- методы/средства, реализуемые на различных этапах стадий жизненного цикла для достижения полноты безопасности СБЗС ПО.

5.1.2 Методы/средства для контроля случайных отказов АС E/E/PE СБЗС-систем, их краткое описание, а также ссылки на источники с полным описанием приведены в приложении А.

5.1.3 Методы/средства для исключения систематических отказов на различных стадиях жизненных циклов СБЗС-систем, их краткие описания, а также ссылки на источники с полным описанием приведены в приложении Б.

5.1.4 Методы/средства для достижения полноты безопасности СБЗС ПО, реализуемые на различных этапах стадий жизненного цикла ПО, их краткое описание, а также ссылки на источники с полным описанием приведены в приложении В.

6 Методы оценки

6.1 Методы оценки функциональной безопасности СБЗС ПО, их краткое описание, а также ссылки на источники с полным описанием приведены в разделе В.6 приложения В.

6.2 Методы оценки полноты безопасности предварительно разработанных программных средств, применяемых для СБЗС-систем, основанные на вероятностном подходе, приведены в приложении Г.

Приложение А
(справочное)

Методы и средства для E/E/PE СБЗС-систем: контроль случайных отказов АС (см. ГОСТ Р 53195.3)

Примечание — Методы/средства, представленные в настоящем приложении, не являются исчерпывающими. Аппаратные средства E/E/PE СБЗС-систем интенсивно развиваются, и методы и средства быстро совершенствуются. При выборе конкретных методов/средств следует ориентироваться, в первую очередь, на стандартизованные методы/средства. В случае применения новых методов/средств всегда следует формировать и сохранять доказательственные материалы, демонстрирующие эффективность применяемых методов/средств по сравнению с методами/средствами, приведенными в настоящем приложении.

А.1 Электрические системы и компоненты

Глобальная цель: управление отказами в электромеханических компонентах.

А.1.1 Отказы, обнаруживаемые мониторингом в режиме с внешним управлением (он-лайн)

Примечание — На этот метод/средство дана ссылка в ГОСТ Р 53195.3 (таблицы А.2, А.3, А.7 и А.14 — А.19).

Цель: обнаружение отказов путем контроля поведения E/E/PE СБЗС-систем в процессе нормального функционирования управляемого оборудования (УО) в режиме с внешним управлением (он-лайн).

Описание: при определенных условиях отказы могут быть обнаружены с помощью информации о поведении УО во времени. Например, если коммутатор, который является частью E/E/PE СБЗС-системы, нормально активизируется со стороны УО и если при этом коммутатор не изменяет свое состояние в предполагаемое время, то отказ может быть обнаружен. Обычными способами невозможно локализовать такой отказ.

А.1.2 Мониторинг релейных контактов

Примечание — На этот метод/средство дана ссылка в ГОСТ Р 53195.3 (таблицы А.2 и А.15).

Цель: обнаружение отказов релейных контактов, например, из-за сварки («залипания»).

Описание: активизируемые контактные реле (или переключаемые контакты в реле) проектируют таким образом, чтобы их поводки контактов были механически жестко связаны между собой. Пусть имеются два набора переключаемых контактов *a u b* (рисунок А.1). Если нормально разомкнутый контакт *b* оказался приваренным («залипшим»), то нормально замкнутый контакт *a* не может замкнуться при обесточивании обмотки реле. Следовательно, контроль замыкания нормально замкнутого контакта *a* при обесточенной обмотке реле может быть использован для указания того, что нормально разомкнутый контакт *b* действительно разомкнут. Отказ замыкания нормально замыкаемого контакта *a* указывает на отказ контакта *b*. Таким образом, схема контроля должна обеспечивать надежное отключение или обеспечивать продолжение отключения при любом управлении оборудования контактом *b*.

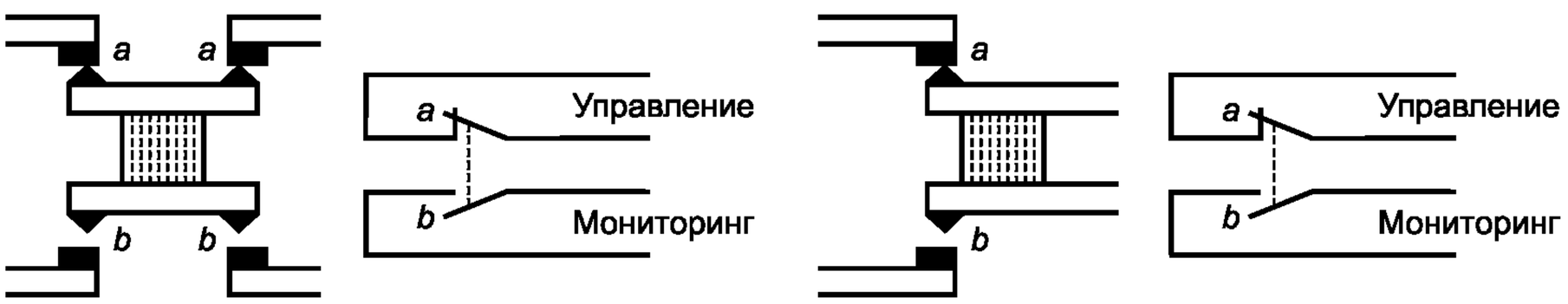


Рисунок А.1 — Контакты реле

А.1.3 Компаратор

Примечание — На этот метод/средство дана ссылка в ГОСТ Р 53195.3 (таблицы А.2 — А.4).

Цель: возможно более раннее обнаружение (не одновременное) отказов в независимом модуле обработки или в компараторе.

Описание: сигналы независимых модулей обработки (процессоров) сигналов сравниваются циклически или непрерывно компаратором АС. Сам компаратор может быть внешне тестируемым, или в нем может быть использована самоконтролирующая технология. Обнаруживаемые различия в поведении модулей обработки используются как сообщения об отказах.

На рисунке А.2 показана схема компаратора в двухканальной системе.

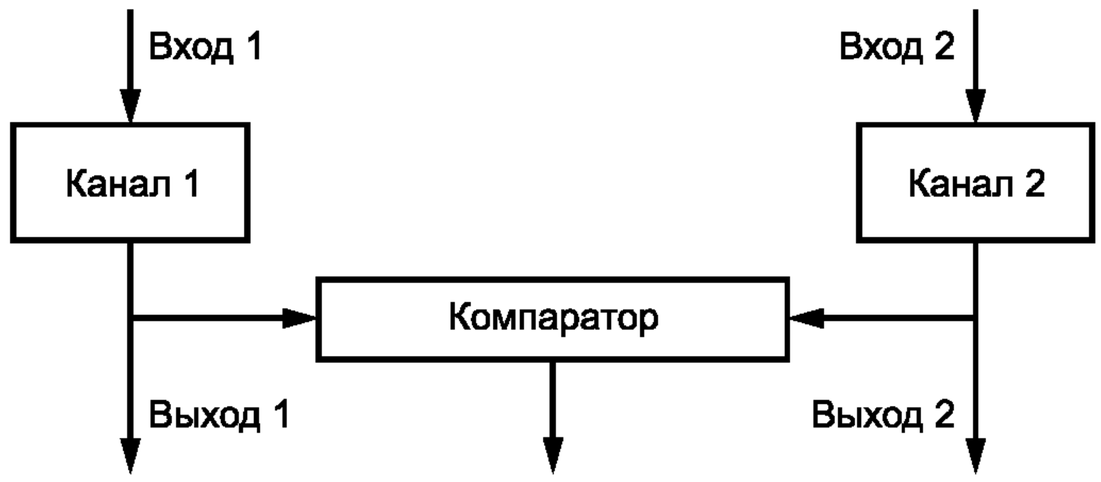


Рисунок А.2 — Компаратор в двухканальной системе

На рисунке А.3 показана схема компаратора в одноканальной системе, реализуемая с использованием ПО. Сложная обработка осуществляется с помощью двух независимых наборов данных и прикладных программ. Поскольку существует только один блок обработки, двойная обработка производится последовательно в интервалах времени *T*1, *T*2, показанных на рисунке. Независимые выходные результаты проверяются программным компаратором в интервале времени *T*3. При применении двух различных прикладных программ достигается высокий диапазон охвата отказов.

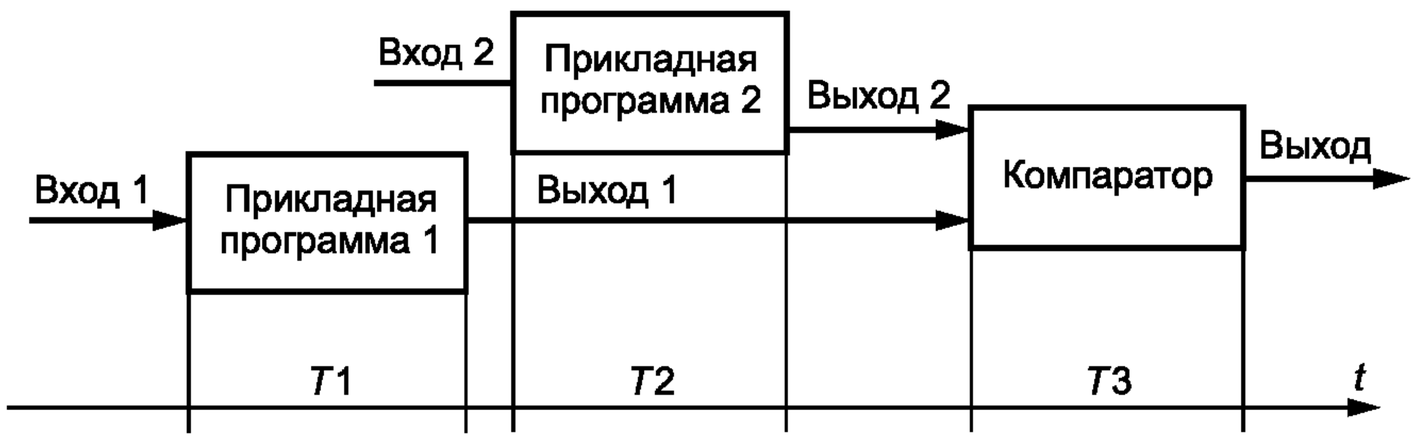


Рисунок А.3 — Компаратор одноканальной системы, реализуемый с помощью ПО

А.1.4 Схема голосования по мажоритарному принципу

Примечание — На этот метод/средство дана ссылка в ГОСТ Р 53195.3 (таблицы А.2 — А.4).

Цель: обнаружение и маскирование отказов по меньшей мере в одном из трех каналов АС.

Описание: для обнаружения и маскирования отказов применяют модуль голосования, использующий мажоритарный принцип (2 из 3, 3 из 3 или *m* из *n*). Для работы схемы голосования может быть использовано внешнее тестирование или в самой схеме могут быть использованы самоконтролирующие технологии. Схема голосования по мажоритарному принципу показана на рисунке А.4.

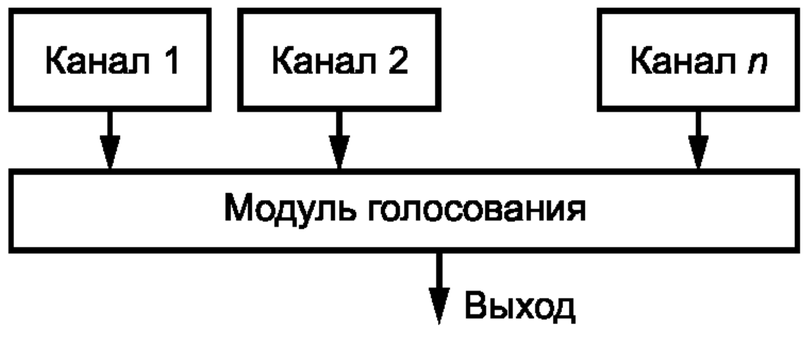


Рисунок А.4 — Голосование по мажоритарному принципу

Подробное описание метода/средства приведено в [1].

А.1.5 Отсутствие электропитания

Примечание — На этот метод/средство дана ссылка в ГОСТ Р 53195.3 (таблицы А.2, А.9, А.14 и А.15).

Цель: выполнение функции безопасности при выключении или потере электропитания.

Описание: функция безопасности выполняется, если контакты разомкнуты и ток не поступает в АС. Например, при использовании тормозов для останова опасного вращения двигателя тормоза отпускаются замыкающими контактами в СБЗС-системах и включаются размыкающими контактами.

Более подробное описание данного метода/средства приведено в [2].

А.2 Электроника

Глобальная цель: управление отказами в транзисторных компонентах.

А.2.1 Тестирование с использованием избыточных АС

Примечание — На этот метод/средство дана ссылка в ГОСТ Р 53195.3 (таблицы А.3, А.16, А.17 и А.19).

Цель: обнаружение отказов с использованием избыточных АС, то есть с использованием дополнительных АС, не требующихся для реализации выполняемых функций.

Описание: избыточные АС могут быть использованы для тестирования при соответствующей частоте запросов к заданным функциям безопасности. Такой подход обычно требуется для реализации положений подраздела А.1.1 или А.2.2 настоящего приложения.

Различные варианты методов/средств тестирования с использованием избыточных АС описаны в [3 —5, 6].

А.2.2 Динамические принципы

Примечание — На этот метод/средство дана ссылка в ГОСТ Р 53195.3 (таблица А.3).

Цель: обнаружение статических отказов путем динамической обработки сигналов.

Описание: для обнаружения статических отказов в компонентах используется принудительное изменение параметров сторонних статических сигналов (генерируемых внешними и внутренними источниками). Этот метод часто применяют в отношении электромеханических компонентов.

Более подробное описание метода приведено в [7].

А.2.3 Стандартный тестовый порт доступа и структура граничного сканирования

Примечание — На этот метод/средство дана ссылка в ГОСТ Р 53195.3 (таблицы А.3, А.16 и А.19).

Цель: управление и наблюдение за происходящим на каждом контакте интегральной схемы (ИС).

Описание: тестирование путем граничного сканирования представляет собой метод построения ИС, который повышает ее способность к тестируемости, разрешая проблему доступа к внутренним точкам тестируемой схемы. В типичной сканируемой по границам ИС, содержащей внутренние логические схемы, а также входные и выходные буферы, между внутренними логическими схемами ядра ИС и входными/выходными буферами размещают каскад сдвигового регистра, граничащий с контактами ИС. Каждый каскад сдвигового регистра находится в ячейке граничного сканирования. Ячейка граничного сканирования позволяет осуществлять контроль и наблюдать за происходящим на каждом входном и выходном контакте ИС через стандартный тестовый порт доступа. Тестирование внутренних логических схем ИС выполняется путем отделения размещенных на кристалле (чипе) внутренних логических схем от входных сигналов, получаемых от окружающих компонентов, и последующего выполнения внутреннего самотестирования. Эти тесты могут быть использованы для обнаружения отказов в ИС.

Более подробное описание этого метода/средства приведено в [8 — 10].

А.2.4 Отказоустойчивое оборудование

Примечание — На этот метод/средство дана ссылка в ГОСТ Р 53195.3 (таблица А.3).

Цель: перевод системы в безопасное состояние в случае появления отказов.

Описание: в аппаратно реализованных системах считается, что устройство работает отказоустойчиво, если:

- определенный набор отказов приводит к безопасному состоянию;
- отказы обнаруживаются.

ПРИМЕР — К определенному набору отказов могут относиться константные отказы типа «обрыв», короткие замыкания внутри и между компонентами, а также на соединениях.

Более подробное описание данного метода/средства приведено в [11 — 13].

А.2.5 Избыточный контроль

Примечание — На этот метод/средство дана ссылка в ГОСТ Р 53195.3 (таблица А.3).

Цель: обнаружение отказов путем создания нескольких функциональных модулей, контроля поведения каждого из них для обнаружения отказов и последующего инициирования перехода в безопасное состояние при обнаружении какого-либо несоответствия в поведении.

Описание: функция безопасности выполняется по меньшей мере двумя каналами АС. Выходы этих каналов контролируются. Если выходные сигналы всех каналов не идентичны, то это служит признаком отказа, и инициируется переход в безопасное состояние.

Более подробное описание вариантов данного метода/средства приведено в [13—15].

А.2.6 Электрические/электронные компоненты с автоматической проверкой

Примечание — На этот метод/средство дана ссылка в ГОСТ Р 53195.3 (таблица А.3).

Цель: обнаружение отказов путем периодической проверки функции безопасности.

Описание: АС тестируются до запуска процесса и затем тестируются повторно через определенные интервалы. УО продолжает работу только при условии успешного прохождения каждого теста.

Подробное описание данного метода/средства приведено в [16].

А.2.7 Текущий контроль аналоговых сигналов

Примечание — На этот метод/средство дана ссылка в ГОСТ Р 53195.3 (таблицы А.3 и А.14).

Цель: повышение уверенности в результатах измерения сигналов.

Описание: в случае, когда отключение или безопасные состояния представляются уровнями аналоговых сигналов, обычно контролируют устойчивость уровня этого сигнала. Этот метод обеспечивает непрерывный контроль и высокую степень доверительности в передатчике, снижает частоту необходимого тестирования функции чувствительности передатчика. Подобное тестирование также может быть применено к внешним интерфейсам, например цифровым линиям связи.

Более подробное описание данного метода/средства приведено в [17, 18].

А.2.8 Снижение номинальных характеристик

Цель: повышение надежности работы компонентов АС.

Описание: построение системы выбирают таким образом, чтобы компоненты АС работали на номинальных уровнях нагрузок и иных характеристик, ниже максимально установленных для них технических характеристик. Снижение уровня номинальных нагрузок и характеристик — это обычная практика обеспечения гарантии того, что, например, при всех нормальных условиях окружающей среды компоненты будут нормально функционировать при уровнях нагрузок меньше максимальных уровней.

А.3 Устройства обработки данных

Глобальная цель: распознавание отказов, приводящих к неправильным результатам в модулях обработки.

А.3.1 Программное самотестирование: ограниченное число комбинаций (один канал)

Примечание — На этот метод/средство дана ссылка в ГОСТ Р 53195.3 (таблица А.4).

Цель: возможно более раннее обнаружение отказов в устройствах обработки данных.

Описание: АС создают с использованием стандартных методов, в которых не учитываются какие-либо специальные требования к безопасности. Обнаружение отказов реализуется целиком дополнительными программными функциями, которые выполняют самотестирование с использованием не менее двух дополнительных комбинаций данных (например, 55hex и AAhex).

А.3.2 Программное самотестирование: блуждающий бит (один канал)

Примечание — На этот метод/средство дана ссылка в ГОСТ Р 53195.3 (таблица А.4).

Цель: возможно более раннее обнаружение отказов в устройствах памяти (например, в регистрах) и дешифраторе команд процессора.

Описание: обнаружение отказов полностью реализуется дополнительными программными функциями, выполняющими самотестирование с использованием комбинации данных (например, комбинации «блуждающих битов»), которая тестирует физическую память (регистры данных и адресные регистры) и дешифратор команд. Охват диагностикой в этом случае составляет не более 90 %.

При тестировании одиночным блуждающим битом данные записываются в каждый адрес и считываются. Затем информация сдвигается влево на один бит так, что информация начинается со второго бита, и проводится такое же тестирование снова. Процесс повторяется 32 раза до тех пор, пока тестовый бит не выйдет из испытательных данных. Аналогичное тестирование повторяется для полного диапазона тестирования (рисунок А.5).

При тестировании псевдослучайной последовательности предварительно подготовленная псевдослучайная уникальная последовательность записывается в каждое ОЗУ. После первого прохождения теста программа воспроизводит данные еще раз для подтверждения правильности размещения данных в памяти.

Содержание регистра до старта теста «блуждающий бит»

0	0	0	0

Старт тестовой последовательности

0	0	0	1

Содержание регистра после контроля

0	0	0	0

Повторение процедуры до завершения контроля всех ячеек регистра

0	0	1	0
0	1	0	0
1	0	0	0

Рисунок А.5 — Тестирование регистра блуждающим битом

А.3.3 Самотестирование, обеспечиваемое АС (один канал)

Примечание — На этот метод/средство дана ссылка в ГОСТ Р 53195.3 (таблица А.4).

Цель: возможно более раннее обнаружение отказов в процессоре с использованием специальных АС, которые увеличивают скорость и расширяют области обнаружения отказов.

Описание: применяют дополнительные специальные АС, которые обеспечивают функции самотестирования для обнаружения отказов. Например, это может быть аппаратный модуль, циклически контролирующий выход определенной битовой комбинации в соответствии с принципом действия сторожевой схемы (рисунок А.6).

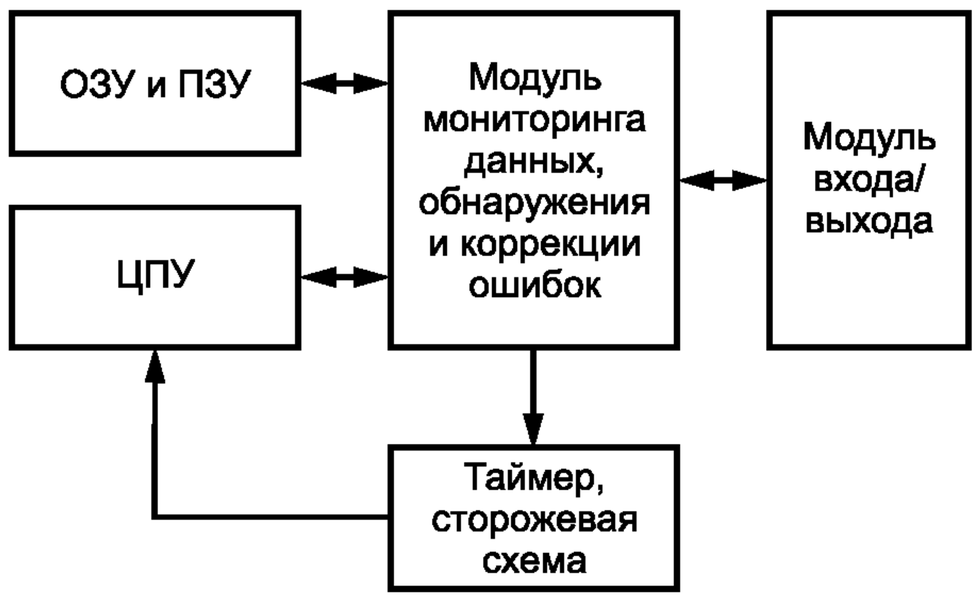


Рисунок А.6 — Структурная схема процессорного модуля с самотестированием и обнаружением отказов

Более подробное описание метода/средства приведено в [19, 20].

А.3.4 Закодированная обработка (одноканальная)

Примечание — На этот метод/средство дана ссылка в ГОСТ Р 53195.3 (таблица А.4).

Цель: возможно более раннее обнаружение отказов в процессоре.

Описание: процессоры могут быть спроектированы с использованием специальных встроенных схемных средств обнаружения отказов или исправления отказов. До недавнего времени эти методы/средства применялись только в относительно простых схемах и не получали широкого распространения; однако не следует исключать их применение в будущих разработках.

ПРИМЕР — Модули входа/выхода дублируются, и с выхода передающего на вход приемного устройства передаются два одинаковых сигнала А и В по двум отдельным линиям. Два входных сигнала поступают на логическое устройство, выполняющее операцию конъюнкции (логическое И). Если на выходе получается логическая единица, это свидетельствует о совпадении сигналов и исправности соединительных линий и передающего устройства. Если на выходе образуется логический нуль, это свидетельствует о неисправности линии (линий) и/или передающего устройства. Логическая единица используется для формирования разрешающей команды на дальнейшую передачу и обработку сигнала. Логический нуль используется для сигнализации о неисправности. На практике могут быть использованы другие логические операции, например ИСКЛЮЧАЮЩЕЕ ИЛИ, при передаче инверсии одного из сигналов («НЕ А») или («НЕ Б»), которые приводят к тем же результатам.

Более подробное описание данного метода/средства приведено в [4 и 18].

А.3.5 Программное обнаружение несовпадений

Примечание — На этот метод/средство дана ссылка в ГОСТ Р 53195.3 (таблица А.4).

Цель: возможно более раннее обнаружение отказов в процессоре путем динамического программного сравнения.

Описание: два модуля взаимно обмениваются данными (включая результаты, промежуточные результаты и тестируемые данные). Сравнение данных, выдаваемых с использованием программных средств в каждом модуле, и обнаруженные различия инициируют формирование сообщения об отказе (см. также рисунок А.3).

А.4 Постоянное запоминающее устройство

Глобальная цель: выявление изменения информации в ПЗУ.

А.4.1 Сохранение слов с многобитовой избыточностью (например, контроль ПЗУ модифицированным кодом Хэмминга)

Примечание — См. также А.5.6 и В.3.2. На этот метод/средство дана ссылка в ГОСТ Р 53195.3 (таблица А.5).

Цель: обнаружение всех однобитовых ошибок, всех двухбитовых ошибок и некоторых ошибок во всех битах 16-битового слова.

Описание: каждое слово в памяти расширяется несколькими избыточными битами для формирования модифицированного кода Хэмминга с кодовым расстоянием, по меньшей мере равным 4. При каждом считывании слова проверка избыточных битов может указывать, произошло искажение данных или нет. При обнаружении различия вырабатывается сообщение об ошибке. Эта процедура может также использоваться для обнаружения ошибок адресации путем вычисления избыточных битов для слова данных, объединенного с его адресом.

Подробное описание методов/средств обнаружения и коррекции ошибок приведено в [21—24].

А.4.2 Модифицируемая контрольная сумма

Примечание — На этот метод/средство дана ссылка в ГОСТ Р 53195.3 (таблица А.5).

Цель: обнаружение всех ошибок нечетных битов, то есть приблизительно 50 % всех возможных битовых ошибок.

Описание: контрольная сумма образуется соответствующим алгоритмом, который использует все слова в блоке памяти. Эта контрольная сумма может храниться как дополнительное слово в ПЗУ либо она может быть добавлена как дополнительное слово в блок памяти для того, чтобы алгоритм контрольной суммы выработал заранее заданное значение. В последнем тестировании памяти контрольная сумма создается снова с использованием того же алгоритма, и результат сравнивается с запомненным или с заданным значением. При обнаружении различий вырабатывается сообщение об ошибке.

Этот метод более подробно описан в [21].

А.4.3 Сигнатура одного слова (8 битов)

Примечание — На этот метод/средство дана ссылка в ГОСТ Р 53195.3 (таблица А.5).

Цель: обнаружение значительного числа однобитовых ошибок, многобитовых ошибок в слове с обнаружением приблизительно 99,6 % всех возможных битовых ошибок.

Описание: содержимое блока памяти сжимается (с использованием аппаратных или программных средств) в одно слово памяти с использованием алгоритма контроля с помощью циклического избыточного кода (CRC). Типичный алгоритм CRC рассматривает все содержимое блока памяти как побайтовый или побитовый последовательный поток данных, в котором выполняется непрерывное полиномиальное деление с использованием полиномиального генератора. Остаток от деления сохраняется и представляет собой сжатое содержимое памяти — это «сигнатура» памяти. Сигнатура вычисляется еще один раз в последующем тестировании и сравнивается с уже запомненным значением. При обнаружении различий выдается сообщение об ошибке.

Данный метод/средство более подробно описан в [25, 26].

А.4.4 Сигнатура двойного слова (16 битов)

Примечание — На этот метод/средство дана ссылка в ГОСТ Р 53195.3 (таблица А.5).

Цель: обнаружение большинства однобитовых и многобитовых ошибок в слове с обнаружением приблизительно 99,998 % всех возможных битовых ошибок.

Описание: при применении этого метода сигнатура вычисляется с использованием алгоритма контроля с помощью циклического избыточного кода (CRC), однако длина результирующего значения составляет по меньшей мере два слова. Расширенная сигнатура заносится в память, повторно вычисляется и сравнивается как одно слово. При обнаружении различий между сохраненной и повторно вычисленной сигнатурами выдается сообщение об ошибке. Метод позволяет обнаружить примерно 99, 998 % всех возможных битовых ошибок.

Данный метод/средство более подробно описан в [25, 26].

А.4.5 Повторение блока (например, дублирование ROM аппаратными и программными средствами)

Примечание — На этот метод/средство дана ссылка в ГОСТ Р 53195.3 (таблица А.5).

Цель: обнаружение всех битовых ошибок.

Описание: адресное пространство дублируется в двух областях или устройствах памяти. Первая область памяти работает в нормальном режиме. Вторая — содержит ту же информацию и доступна параллельно с первой. Их выходы сравниваются, и при обнаружении различий выдается сообщение об ошибке (см. также А.5.7). Для обнаружения некоторых видов битовых ошибок данные должны запоминаться инверсно в одной из двух областей памяти и инвертироваться обратно при чтении.

Данный метод/средство более подробно описан в [27].

А.5 Изменяемые пространства памяти

Глобальная цель: обнаружение отказов во время процессов адресации, записи и считывания.

Должны быть учтены следующие отказы:

- константные отказы ячеек памяти;
- паразитные связи между ячейками памяти;
- отказы адресации;
- изменение содержимого из-за внешних воздействий.

А.5.1 Тесты «шахматная доска» и «марш» для памяти с произвольным доступом (RAM)

Примечание — На эти методы/средства дана ссылка в ГОСТ Р 53195.3 (таблица А.6).

Цель: обнаружение преимущественно статических битовых ошибок.

Описание: распределенная в шахматном порядке битовая комбинация нулей и единиц записывается в ячейки памяти с битовой организацией. Затем эти ячейки анализируются попарно, чтобы убедиться в их одинаковости и правильности. Адрес первой ячейки такой пары является переменным, а адрес второй ячейки этой пары образуется путем битового инвертирования первого адреса. Первое прохождение диапазона адресов памяти осуществляется в направлении увеличения переменных адресов, а при втором прохождении — в направлении уменьшения адресов. После этого оба прохождения повторяются с заранее заданным инвертированием. При обнаружении какого-либо различия выдается сообщение об отказе.

При «маршевом» тестировании памяти с произвольным доступом ячейки памяти с битовой организацией инициализируются унифицированным потоком битов. При первом прохождении ячейки анализируются в нисходящей последовательности. Проверяется правильность содержимого каждой ячейки, и ее содержимое инвертируется. Базовая последовательность битов, которая создана в первом прохождении, рассматривается при втором прохождении в убывающем порядке и тем же способом. Первые прохождения повторяются с инвертируемыми предварительными значениями в третьем и четвертом прохождениях. При обнаружении различий выдается сообщение об отказе.

Данные тесты более подробно описаны в [28, 29].

А.5.2 Тест «прогулочная дорожка» для памяти с произвольным доступом

Примечание — На этот метод/средство дана ссылка в ГОСТ Р 53195.3 (таблица А.6).

Цель: обнаружение статических и динамических ошибочных битов и перекрестных помех между ячейками памяти.

Описание: тестируемая область памяти инициализируется унифицированным потоком битов. Затем первая ячейка инвертируется, и остальная часть памяти анализируется на правильность. После этого первая ячейка повторно инвертируется для возврата в исходное состояние, и вся процедура повторяется для следующей ячейки. Второе прохождение модели «блуждающего бита» осуществляется при инверсии всех первоначально назначенных значений памяти. При обнаружении различий выдается сообщение об ошибке.

Данный метод/средство более подробно описан в [28, 29].

А.5.3 Тест «бегущий код» для памяти с произвольным доступом

Примечание — На этот метод/средство дана ссылка в МЭК 61508-3 (таблица А.6).

Цель: обнаружение статических битовых ошибок и динамических ошибок из-за паразитных связей.

Описание: при тестировании памяти с произвольным доступом «попарной записью-считыванием» выбранная область памяти сначала инициализируется унифицированно (то есть все 0 или все 1). После этого первая ячейка памяти тестируется и затем инвертируется, а все остальные ячейки анализируются на правильность содержимого. После каждого доступа к чтению одной из оставшихся ячеек инвертированная ячейка также проверяется. Эта процедура повторяется для каждой ячейки в выбранной области памяти. Второе прохождение выполняется противоположно первоначальному прохождению. Любые различия приводят к выдаче сообщения об ошибке.

Тестирование «прозрачной попарной записью-считыванием» представляет собой вариацию описанной выше процедуры: вместо инициализации всех ячеек в выбранной области памяти существующее содержимое остается неизменным, а для сравнения содержимого набора ячеек используются контрольные суммы (сигнатуры). Выбирается первая тестируемая ячейка области памяти и вычисляется и сохраняется сигнатура *S*1 всех оставшихся ячеек области. Затем тестируемые ячейки инвертируются, и повторно вычисляется сигнатура *S*2. (После каждого доступа к чтению к одной из оставшихся ячеек инвертируемая ячейка также проверяется.) *S*2 сравнивается с *S*1, и при любом различии выдается сообщение об ошибке. Тестируемая ячейка повторно инвертируется для повторного установления исходного содержимого, и сигнатура *S*3 всех оставшихся ячеек повторно вычисляется и сравнивается с *S*1. Любые различия приводят к выдаче сообщения об ошибке. Все ячейки памяти в выбранной области тестируются тем же способом.

Данный метод/средство более подробно описан в [28, 29].

А.5.4 Тест «Авраам» для памяти с произвольным доступом

Примечание — На этот метод/средство дана ссылка в ГОСТ Р 53195.3 (таблица А.6).

Цель: обнаружение всех константных отказов и отказов, возникающих из-за связей между ячейками памяти.

Описание: общее число обнаруженных ошибок выше, чем при тесте «попарная запись-считывание». Число операций, необходимых для выполнения всего тестирования памяти, составляет примерно 30*n,* где *n* — число ячеек памяти. Тестирование может быть «прозрачным» при выполнении запоминания и тестирования в различных временных интервалах в периоде рабочего цикла.

Данный метод/средство более подробно описан в [30, 31].

А.5.5 Однобитовая избыточность (например, контроль памяти с произвольным доступом с помощью бита четности)

Примечание — На этот метод/средство дана ссылка в ГОСТ Р 53195.3 (таблица А.6).

Цель: обнаружение 50 % всех возможных битовых ошибок в тестируемой области памяти.

Описание: каждое слово в памяти расширяется на один бит (бит четности), который дополняет каждое слово до четного или нечетного числа логических единиц. Четность слова данных проверяется при каждом считывании. При обнаружении ложного числа единиц выдается сообщение об ошибке. Выбор четности или нечетности должен осуществляться таким образом, чтобы всякий раз, когда в случае ошибки не выдавалось ничего, кроме нулевого (ничего, кроме 0) и единичного (ничего, кроме 1) слова, вырабатывалось уведомление о том, что это слово неправильно закодировано. Контроль четности также может быть использован для обнаружения ошибок адресации, когда четность определяется при объединении слова данных с его адресом.

Данный метод/средство более подробно описан в [32—35].

А.5.6 Контроль памяти с произвольным доступом с помощью модифицированного кода Хэмминга или обнаружение ошибок данных с кодами обнаружения и исправления ошибок (EDC)

Примечание — См. также А.4.1 и В.3.2. На этот метод/средство дана ссылка в ГОСТ Р 53195.3 (таблица А.6).

Цель: обнаружение всех нечетных битовых ошибок, всех двухбитовых ошибок, некоторых трехбитовых ошибок и некоторых многобитовых ошибок.

Описание: каждое слово в памяти расширяется несколькими избыточными битами для формирования модифицированного кода Хэмминга с кодовым расстоянием по меньшей мере равным 4. При каждом считывании слова проверка избыточных битов может указывать, произошло искажение информации или нет. При обнаружении различий выдается сообщение об ошибке. Эта процедура может быть также использована для обнаружения ошибок адресации при вычислении избыточных битов при объединении слова данных с его адресом.

Данный метод/средство более подробно описан в [34, 35].

А.5.7 Дублирование со сравнением памяти с произвольным доступом аппаратными или программными средствами и тестирование путем записи/считывания

Примечание — На этот метод/средство дана ссылка в ГОСТ Р 53195.3 (таблица А.6).

Цель: обнаружение всех битовых ошибок.

Описание: адресное пространство памяти дублируется в двух устройствах (или областях) памяти. Первое устройство памяти функционирует в нормальном режиме. Второе устройство (вторая область) памяти содержит ту же информацию и доступно параллельно с первым(ой). Их выходные данные сравниваются, и при обнаруже-

нии различий выдается сообщение об ошибке. Для обнаружения некоторых видов битовых ошибок данные должны сохраняться инверсно в одном(ой) из двух устройств (областей) памяти и обратно инвертироваться при считывании.

Для памяти, распределенной на нескольких устройствах (например, на массивах накопителей на жестких дисках), применяют так называемые системы RAID, позволяющие обнаруживать и/или корректировать одиночные и кратные ошибки при считывании, в зависимости от уровня RAID:

- для RAID уровня 0 требуется минимально два диска и обеспечивается наивысшая производительность, но без защиты от потери и/или повреждения данных. Алгоритм работы основан на разделении данных на сегменты («полоски» — striping). В том случае, если от дисковой системы требуется наивысшая производительность, но при этом также требуется защита от отказов жестких дисков, и нет недостатка в средствах, устанавливаются зеркально два RAID-контроллера, и каждый конфигурируется под RAID уровня 0;

- для RAID уровня 1 требуется два диска и фактически осуществляется только зеркальная (mirroring) запись-считывание. Такая организация памяти не снижает производительность при считывании, но скорость записи снижается, так как приходится выполнять последовательную запись на два диска — сначала на один, затем на другой;

- для RAID уровня 5 требуется минимально три диска и обеспечивается как защита данных при выходе из строя жестких дисков, так и приемлемая производительность. Применяется запись как с разделением диска на полоски (striping), так и с избыточностью (parity). Избыточность составляет один диск в одном массиве, т. е. при установке трех дисков по 9 ГБ операционная система различит только 18 ГБ. Установив 6 дисков по 9 ГБ, можно использовать для работы 45 ГБ и т. д. Для контроля данных в RAID уровня 5 используется один избыточный диск массива данных;

- для RAID уровня 6 также минимально требуется три диска, но для контроля данных используется два диска. В алгоритме RAID 6 используются два независимых механизма вычисления контрольных значений и два интеллектуальных метода восстановления данных — 2D-XOR и P+Q, что позволяет восстанавливать данные в случае отказа дисков и/или блоков данных (рисунок А.7).

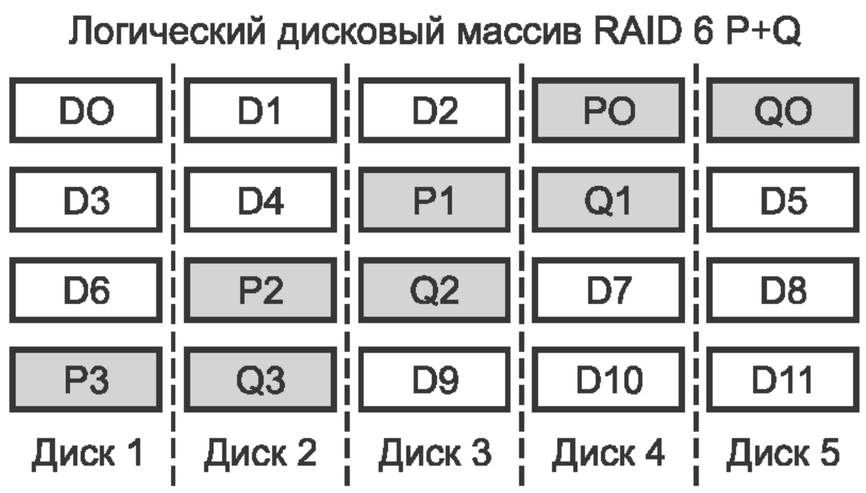


Рисунок А.7 — Структура размещения блоков данных (D) и блоков проверки (P и Q) в системе RAID 6

- для RAID уровня 7 требуются хотя бы один диск и обычное независимое подключение дисков к RAID-контроллеру. Разбиение на полоски и введение избыточности отсутствуют. Сами диски могут быть отформатированы и разбиты на логические диски в необходимой для использования операционной системе. При использовании RAID других уровней это невозможно. Применение RAID 7 фактически представляет собой использование RAID-контроллера в качестве обычного, но очень высокопроизводительного SCSI-контроллера с кэш-памятью;

- для RAID уровня 0 + 1 используется разделение диска на полоски, как в RAID уровня 0, и зеркальная запись-считывание, как в RAID уровня 1. Отличается повышенной, по сравнению с обычным RAID уровня 1, производительностью, хотя избыточность по-прежнему 100 %;

- для RAID уровня 10 применяется та же архитектура, что и в RAID уровня 0 + 1, но использованная для дискового массива из нескольких групп дисков. Избыточность соответственно 100 %;

- для RAID уровня 30 используется разделение диска на полоски, но полоска данных распределяется по большим группам дисков с использованием контроля четности для контроля целостности данных;

- для RAID уровня 50 применяются те же архитектура и принцип действия, что и в RAID уровня 30, но с использованием операции «исключающее ИЛИ» (XOR) для контроля целостности данных.

Различные варианты данного метода/средства более подробно описаны в [36—39].

А.6 Устройства ввода-вывода и интерфейсы (внешний обмен)

Глобальная цель: обнаружение отказов на устройствах ввода и вывода (цифровых, аналоговых, последовательных или параллельных) и предотвраще-ние передачи недопустимых выходных данных для обработки.

А.6.1 Тестирующая комбинация

Примечание — На этот метод/средство дана ссылка в ГОСТ Р 53195.3 (таблицы А.7, А.14 и А.15).

Цель: обнаружение статических (константных) отказов и отказов из-за перекрестных помех.

Описание: осуществляется независимое от потока данных циклическое тестирование входных и выходных элементов. В нем используются определенные тестирующие комбинации для сравнения с соответствующими предполагаемыми значениями (рисунок А.8). Информация, восприятие и оценка тестирующей комбинации должны быть независимыми друг от друга. Тестирующие комбинации не должны неблагоприятно влиять на операции, выполняемые УО.

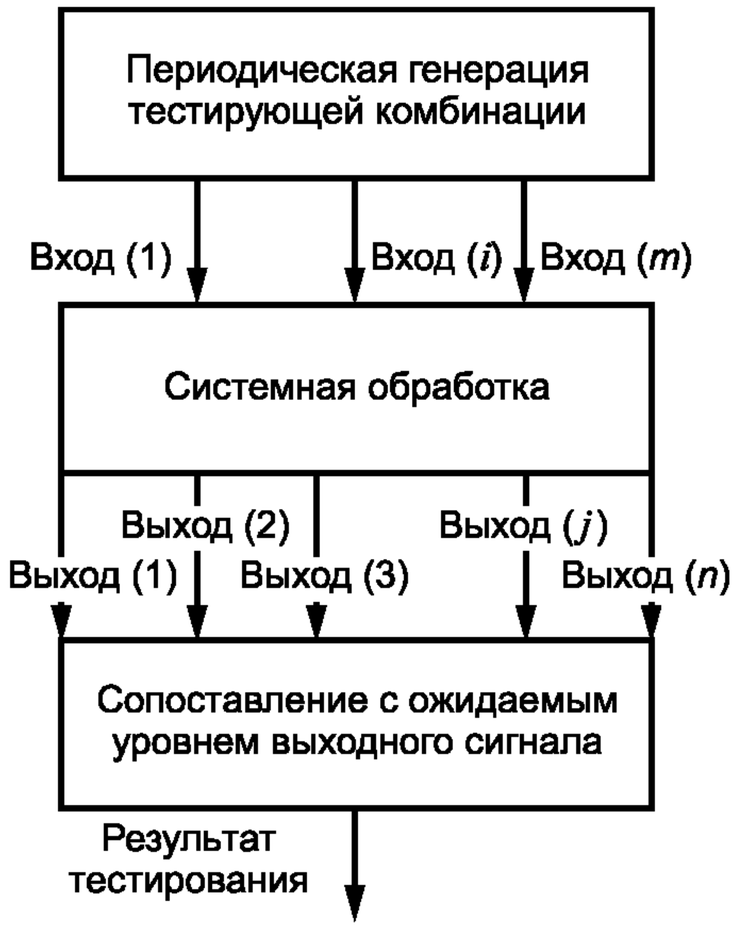


Рисунок А.8 — Применение тестирующей комбинации для обнаружения отказов

Данный метод/средство более подробно описан в [40—43].

А.6.2 Кодовая защита

Примечание — На этот метод/средство дана ссылка в ГОСТ Р 53195.3 (таблицы А.7, А.16, А.17 и А.19).

Цель: обнаружение случайных отказов АС и систематических ошибок в потоке входных/выходных данных.

Описание: метод предназначен для защиты входных и выходных данных от систематических и случайных отказов АС. Кодовая защита обеспечивает зависимое от потока данных обнаружение отказов входных и выходных модулей на основе использования избыточности информации и/или временно́й избыточности. Обычно избыточная информация накладывается на входные и/или выходные данные. Этим самым обеспечиваются средства для мониторинга правильности операций входных и выходных схем. Возможно применение многих вариантов метода, например с модуляцией несущей частоты выходным сигналом датчика либо с добавлением на выходе канала избыточных битов или кодовых слов для контроля истинности прохождения сигнала между логическим модулем и оконечным исполнительным устройством.

Данный метод/средство более подробно описан в [30, 33—35].

А.6.3 Многоканальное параллельное выходное устройство

Примечание — На этот метод/средство дана ссылка в ГОСТ Р 53195.3 (таблица А.7).

Цель: обнаружение случайных (константных) отказов АС, отказов, обусловленных внешними воздействиями, временных сбоев, отказов адресации, постепенных отказов и самоустраняющихся отказов.

Описание: метод/средство состоит в применении зависимого от потока данных многоканального параллельного выходного устройства с независимыми выходами и внешних компараторов для обнаружения случайных аппаратных отказов. Обнаружение отказов осуществляется с помощью внешних компараторов при несовпадении информации на выходах устройств, которые формируют команду отключения УО (см. рисунки А.2, А.4). Этот метод/средство действует только в случае, если поток данных изменяется в интервале диагностического тестирования.

Данный метод/средство более подробно описан в [36].

А.6.4 Средство контроля выходов

Примечание — На этот метод/средство дана ссылка в МЭК ГОСТ Р 53195.3 (таблица А.7).

Цель: обнаружение отдельных отказов — отказов, обусловленных внешними воздействиями, временных сбоев, отказов адресации, постепенных отказов (для аналоговых сигналов) и самоустраняющихся отказов.

Описание: зависимое от потока данных устройство осуществляет сравнение выходных данных с независимыми входными данными, определяя, насколько они соответствуют области допустимых значений (по времени, величине). Обнаруженный отказ не всегда относится к неправильному выходному сигналу. Этот метод/средство действует только в том случае, если поток данных изменяется в интервале диагностического тестирования.

ПРИМЕР — На рисунке А.9 показана возможная схема устройства сравнения независимых входных данных с фактическими выходными данными.

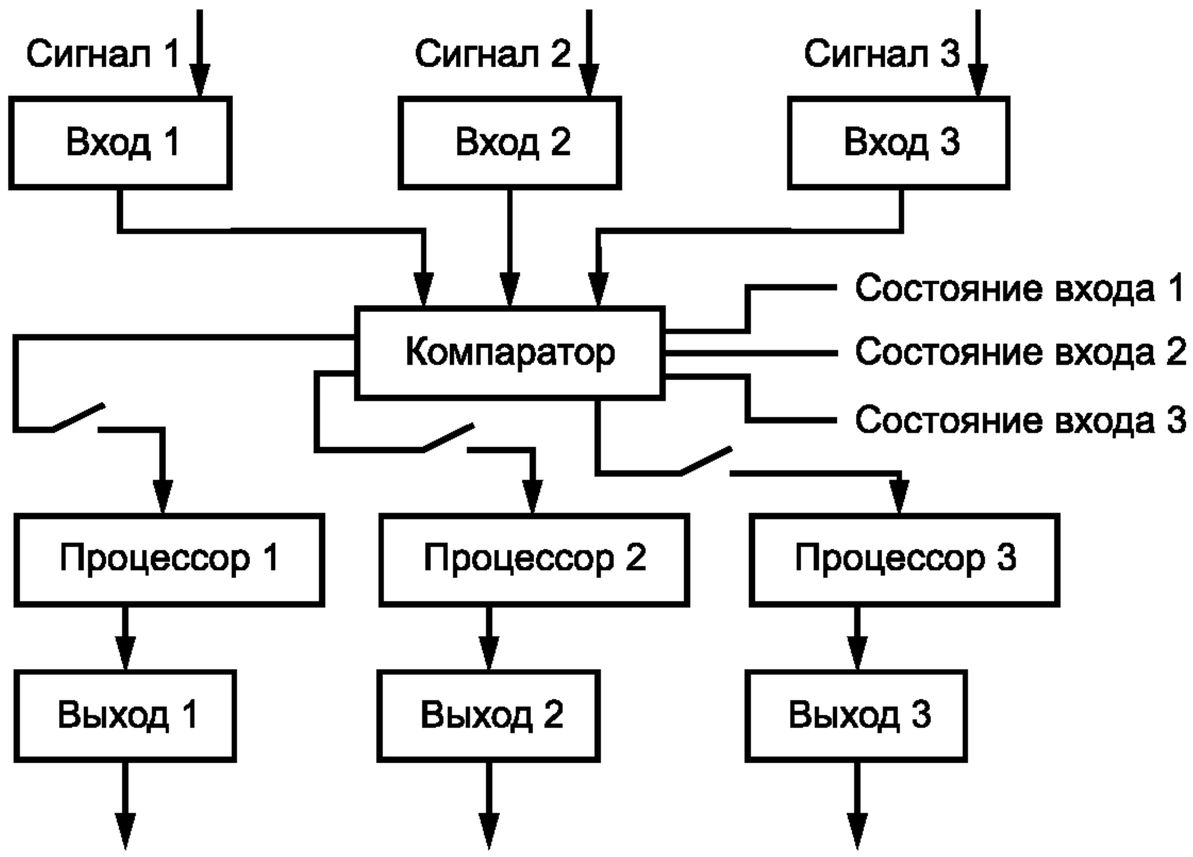


Рисунок А.9 — Устройство сравнения выходных данных с независимыми данными

Данный метод/средство более подробно описан в [36].

А.6.5 Сравнение/голосование входных данных

Примечание — На этот метод/средство дана ссылка в ГОСТ Р 53195.3 (таблицы А.7 и А.14).

Цель: обнаружение отдельных отказов — отказов, обусловленных внешними воздействиями, временных сбоев, отказов адресации, постепенных отказов (для аналоговых сигналов) и самоустраняющихся отказов.

Описание: зависимое от потока данных устройство осуществляет сравнение выходных данных с независимыми входными данными, определяя, насколько они соответствуют области допустимых значений (по времени, величине), — см. рисунок А.9. Обнаруженный отказ не всегда относится к неправильному выходному сигналу. Реализуемая избыточность может быть 1 из 2, 2 из 3 или более высокая. Этот метод/средство действует только в том случае, если поток данных изменяется в интервале диагностического тестирования.

Данный метод/средство более подробно описан в [36].

А.7 Маршруты данных (внутренний обмен)

Глобальная цель: обнаружение отказов, обусловленных процессом обмена информацией.

А.7.1 Однобитовая аппаратная избыточность

Примечание — На этот метод/средство дана ссылка в МЭК ГОСТ Р 53195.3 (таблица А.8).

Цель: обнаружение всех нечетных однобитовых ошибок, то есть 50 % всех возможных одиночных ошибок в потоке данных.

Описание: число проводников шины расширяется на один проводник (бит в параллельном коде), и этот дополнительный проводник (бит) используется для обнаружения отказов путем проверки на четность.

А.7.2 Многобитовая аппаратная избыточность

Примечание — На этот метод/средство дана ссылка в ГОСТ Р 53195.3 (таблица А.8).

Цель: обнаружение отказов в процессе передачи данных по шине и в последовательных каналах связи.

Описание:

вариант 1: число проводников шины расширяется на два или более проводников (битов), и эти дополнительные проводники (биты) используются для обнаружения отказов с применением кода Хэмминга;

вариант 2: применяются два канала связи, по которым передаются антивалентные сигналы с одними и теми же данными. На приемном конце сигналы сравниваются с использованием логической цепи «исключающее ИЛИ». В случае несовпадения данных формируется сигнал обнаружения отказа (ошибки).

Данный метод/средство более подробно описан в [4—6, 18].

А.7.3 Полная аппаратная избыточность

Примечание — На этот метод/средство дана ссылка в ГОСТ Р 53195.3 (таблица А.8).

Цель: обнаружение отказов в процессе передачи путем сравнения сигналов двух шин.

Описание: шина дублируется, и дополнительные проводники (биты) используются для обнаружения отказов.

Данный метод/средство более подробно описан в [4—6, 18].

А.7.4 Анализ с использованием тестирующих комбинаций

Примечание — На этот метод/средство дана ссылка в ГОСТ Р 53195.3 (таблица А.8).

Цель: обнаружение статических отказов (константных отказов) и перекрестных помех.

Описание: осуществляется независимое от потока данных циклическое тестирование маршрутов данных. Используется определенная тестирующая комбинация для сравнения наблюдаемых значений с соответствующими ожидаемыми значениями.

Информация тестирующей комбинации, ее получение и оценка тестирующей комбинации должны быть независимыми друг от друга. Тестирующие комбинации не должны неблагоприятно влиять на операции УО. См. также А.6.1 и рисунок А.8.

Данный метод/средство более подробно описан в [42—45].

А.7.5 Избыточность при передаче

Примечание — На этот метод дана ссылка в ГОСТ Р 53195.3 (таблица А.8).

Цель: обнаружение самоустраняющихся отказов при обмене данными по шине.

Описание: информация передается последовательно несколько раз. Метод эффективен только для самоустраняющихся отказов.

А.7.6 Информационная избыточность

Примечание — На этот метод/средство дана ссылка в ГОСТ Р 53195.3 (таблица А.8).

Цель: обнаружение отказов при передаче данных по шине.

Описание: данные передаются блоками вместе с вычисленной контрольной суммой для каждого блока. Затем приемник повторно вычисляет контрольную сумму полученных данных, и результат сравнивается с принятой контрольной суммой.

Данный метод/средство более подробно описан в [33—35].

А.8 Устройства обеспечения (например, электропитания, синхронизации и т.п.)

Глобальная цель: обнаружение отказов или управление отказами источников электропитания.

А.8.1 Защита от падения напряжения

Примечание — На этот метод/средство дана ссылка в ГОСТ Р 53195.3 (таблица А.9).

Цель: защита связанных с безопасностью систем от падения напряжения.

Описание: осуществление безопасного отключения системы устройством защитного отключения, реагирующего на предельные отклонения напряжения. Контролирующая цепь контролирует напряжение от источника электропитания. В случае падения напряжения ниже установленных пределов она формирует сигнал тревоги, который инициирует действия микроконтроллера, снижающие риск (безопасное отключение потребляющего устройства, переход на резервное электропитание и т. п.). В случае восстановления заданного напряжения логическое устройство инициирует перезапуск (перезагрузку) микроконтроллера в стандартный режим работы. На рисунке А.10 показана структурная схема одноканального модуля защиты потребляющего устройства от пониженного напряжения, а на рисунке А.11 — структурная схема многоканального модуля защиты.

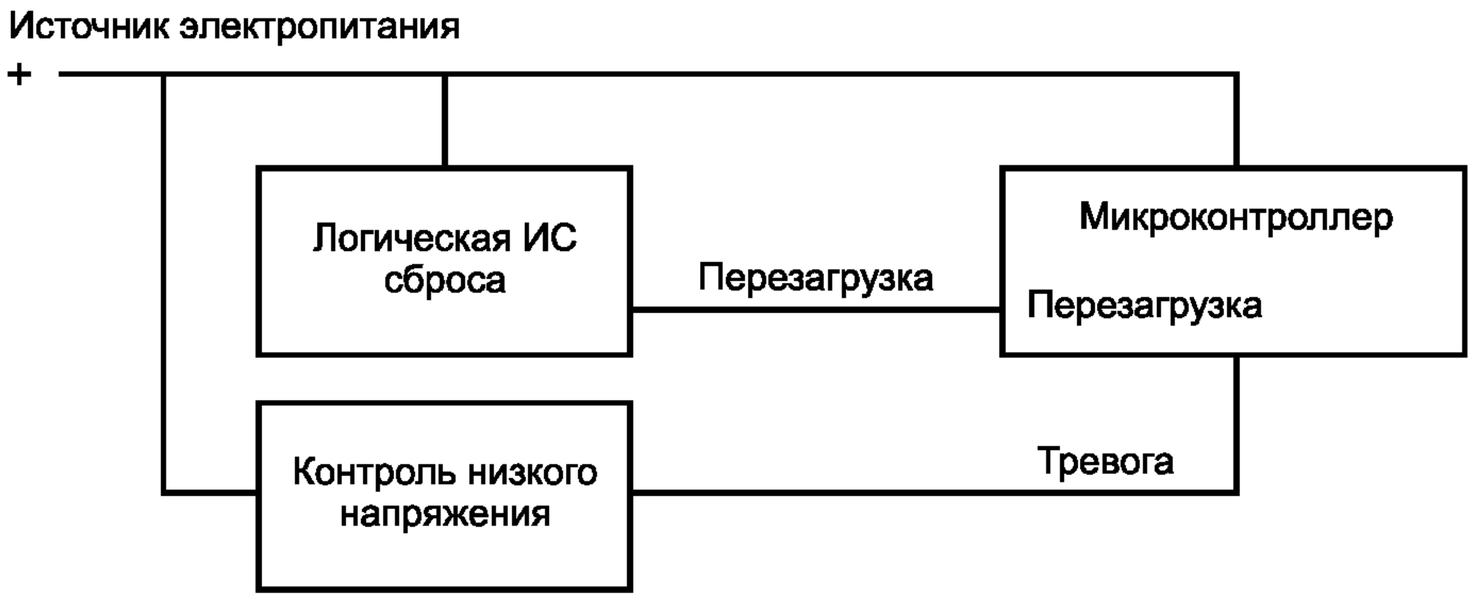


Рисунок А.10 — Структурная схема одноканального модуля защиты от пониженного напряжения

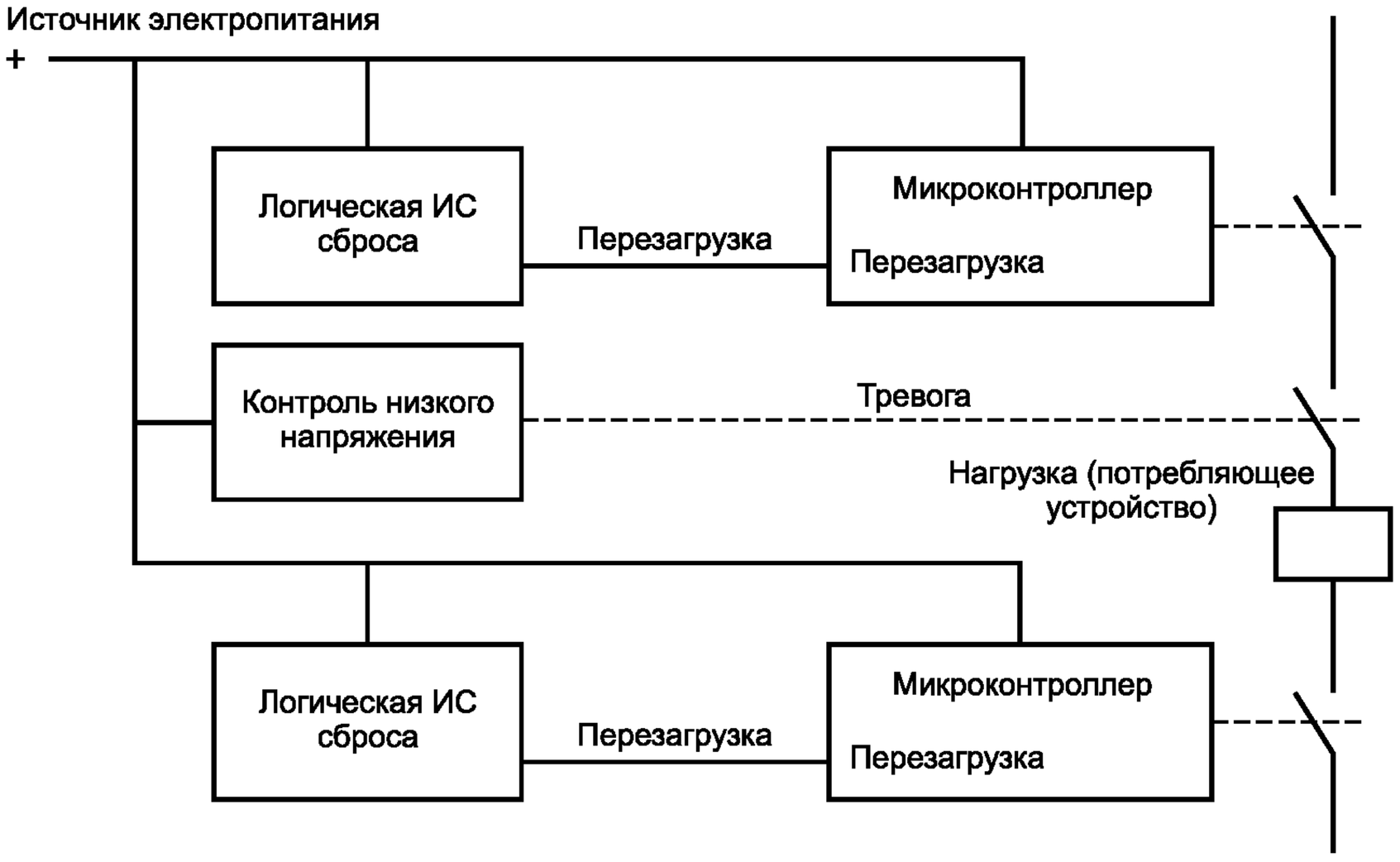


Рисунок А.11 — Структурная схема многоканального модуля защиты от пониженного напряжения

А.8.2 Защита от броска напряжения с помощью безопасного выключения

Примечание — На этот метод/средство дана ссылка в ГОСТ Р 53195.3 (таблица А.9).

Цель: защита связанных с безопасностью систем от броска напряжения.

Описание: раннее обнаружение повышения (броска) напряжения и переключение всех выходов системы в безопасное состояние с использованием процедуры отключения электропитания или переключения на резервный блок электропитания. Структура и принцип действия модуля защиты от повышения (броска) напряжения аналогичны действию модуля защиты от пониженного напряжения (см. А.8.1).

А.8.3 Управление напряжением вторичного источника электропитания

Примечание — На этот метод/средство дана ссылка в ГОСТ Р 53195.3 (таблица А.9).

Цель: контроль напряжений вторичного источника электропитания и инициализация перехода в безопасное состояние, если напряжение не находится в заданном диапазоне.

Описание: осуществление контроля напряжений вторичного источника электропитания и, если напряжение не находится в заданном диапазоне, отключение электропитания системы либо переключение ее на резервный блок электропитания. Структура и принцип действия системы (модуля) контроля вторичного источника электропитания аналогичны структуре и принципу действия модулей, описанных в А.8.1 и А.8.2.

А.8.4 Безопасное выключение

Примечание — На этот метод/средство дана ссылка в ГОСТ Р 53195.3 (таблица А.9).

Цель: выключение электропитания с безопасным сохранением запомненной информации.

Описание: осуществляется раннее обнаружение повышения (броска) или понижения напряжения с тем чтобы внутреннее состояние СБЗС-системы (подсистемы) при необходимости могло быть сохранено в энергонезависимой памяти и чтобы все выходы могли быть установлены в безопасное состояние процедурой отключения электропитания, или все выходы могли быть переключены в безопасное состояние процедурой отключения электропитания, либо потребляющие устройства могли быть подключены к резервному источнику электропитания.

А.9 Временно́й и логический контроль последовательности выполнения программ

Примечание — На эту группу методов или средств даны ссылки в ГОСТ Р 53195.3 (таблицы А.16, А.17 и А.19).

Глобальная цель: обнаружение искаженных программных последовательностей, возникающих в случаях, когда отдельные элементы программы (например, программные модули, подпрограммы или команды) обрабатываются в неправильной последовательности, или в непредусмотренный период времени, либо при сбое тактовой частоты процессора, во избежание неверного выполнения или невыполнения функции безопасности (см. рисунок А.12).

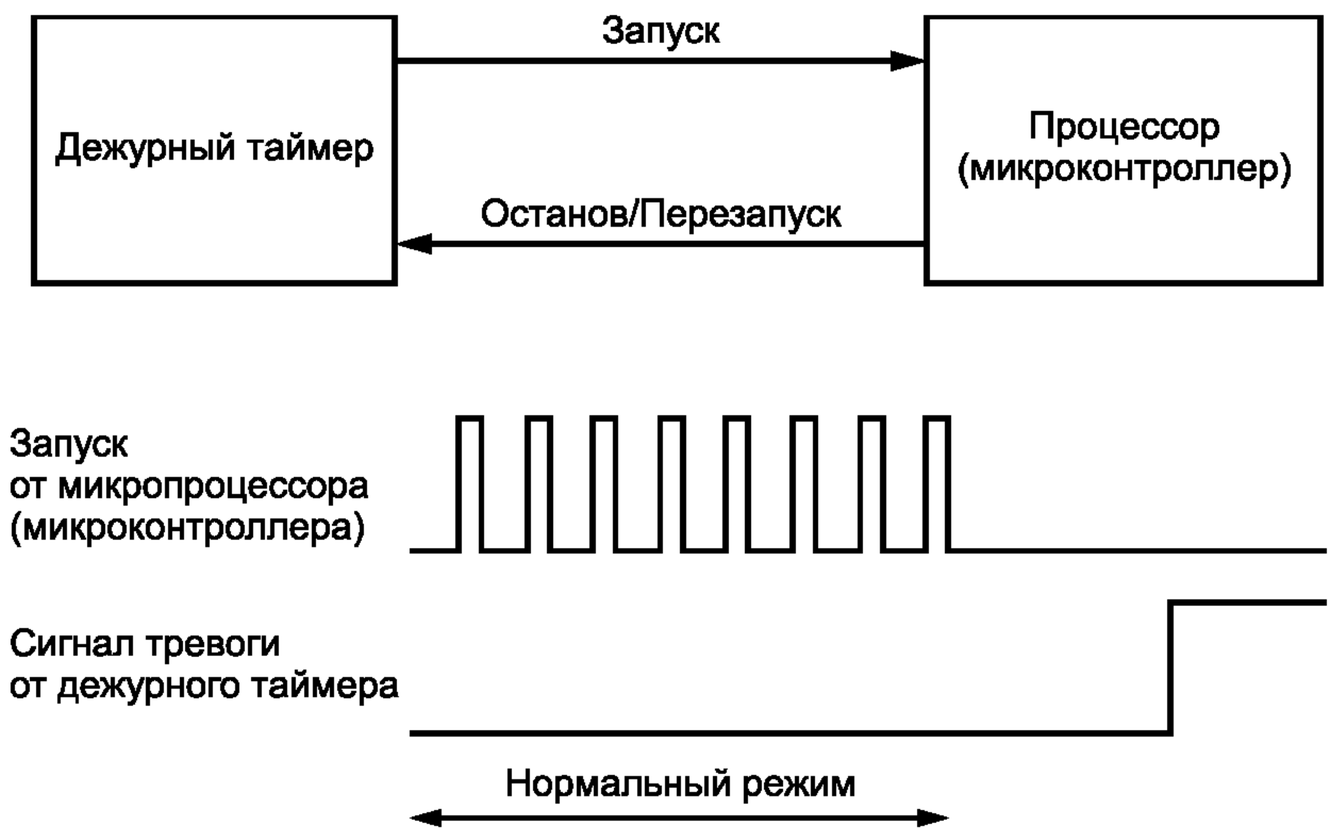


Рисунок А.12 — Структурная схема и временны́е диаграммы модуля временно́го и логического контролей последовательности выполнения программ

А.9.1 Контрольный датчик времени с отдельной временно́й базой без временно́го окна

Примечание — На этот метод/средство дана ссылка в ГОСТ Р 53195.3 (таблицы А.10 и А.12).

Цель: контроль поведения и последовательности выполнения программ.

Описание: внешний элемент (например, контрольный датчик времени), задающий интервалы времени с отдельной временно́й базой, периодически переключается для контроля поведения компьютера и последовательности выполнения программ. Осуществляется слежение за тем, чтобы моменты переключения были правильно расположены в программе. Контрольный датчик времени не переключается с некоторым фиксированным периодом, однако для него задается максимальный временно́й интервал.

А.9.2 Контрольный датчик времени с отдельной временно́й базой и временны́м окном

Примечание — На этот метод/средство дана ссылка в ГОСТ Р 53195.3 (таблицы А.10 и А.12).

Цель: контроль поведения и последовательности выполнения программ.

Описание: внешний элемент (например, контрольный датчик времени), задающий интервалы времени с отдельной временно́й базой, периодически переключается для контроля поведения компьютера и последовательности выполнения программ. Производится слежение за тем, чтобы моменты переключения были правильно расположены в программе. Если выполнение программы занимает более длительное или более короткое время, чем ожидается, выполняется чрезвычайное действие.

А.9.3 Логический контроль последовательности выполнения программ

Примечание — На этот метод дана ссылка в ГОСТ Р 53195.3 (таблицы А.10 и А.12).

Цель: контроль правильной последовательности выполнения отдельных частей программы.

Описание: правильная последовательность выполнения отдельных частей программы контролируется с помощью программных средств (процедур учета, ключевых процедур) или с использованием внешних средств контроля. Осуществляется контроль за тем, чтобы точки проверки располагались в программе правильно.

Более подробное описание данного метода приведено в [45].

А.9.4 Комбинация временно́го и логического контролей последовательности программ

Примечание — На этот метод/средство дана ссылка в ГОСТ Р 53195.3 (таблицы А.10 и А.12).

Цель: контроль поведения и правильной последовательности выполнения отдельных частей программы.

Описание: устройство, задающее интервалы времени (например, контрольный датчик времени), контролирующее последовательность программ, вновь запускается только в том случае, если последовательность частей программы выполняется правильно.

Данный метод/средство более подробно описан в [45].

А.9.5 Проверка временно́го контроля в режиме он-лайн

Примечание — На этот метод/средство дана ссылка в ГОСТ Р 53195.3 (таблицы А.10 и А.12).

Цель: обнаружение отказов при временно́м контроле.

Описание: временно́й контроль проверяется при запуске, а запуск возможен только в том случае, если временно́й контроль успешно завершен. Например, датчик температуры может быть проверен нагретым резистором при запуске.

А.10 Средства охлаждения и подогрева

Примечание — На эту группу средств дана ссылка в ГОСТ Р 53195.3 (таблицы А.17 и А.19).

Глобальная цель: управление отказами в средствах охлаждения (вентиляции) и подогрева и/или их контроль, если они связаны с безопасностью.

Данные средства более подробно описаны в [46—49].

А.10.1 Датчик температуры

Примечание — На это средство дана ссылка в ГОСТ Р 53195.3 (таблица А.11).

Цель: обнаружение температурного перегрева или недогрева до того, как система начнет действовать вне заданных требований.

Описание: применяется датчик температуры, который контролирует температуру в наиболее критических точках E/E/PE СБЗС-системы (подсистемы) или составляющих. Чрезвычайное действие осуществляется до того, как температура в критических точках выйдет за установленные в требованиях пределы.

А.10.2 Управление вентиляцией

Примечание — На этот метод/средство дана ссылка в ГОСТ Р 53195.3 (таблица А.11).

Цель: обнаружение отказов в работе вентилятора.

Описание: контролируется правильность работы вентилятора. Если вентилятор работает не должным образом, предпринимаются действия по обслуживанию или аварийному отключению.

А.10.3 Безопасное отключение с использованием плавкого предохранителя

Примечание — На этот метод/средство дана ссылка в ГОСТ Р 53195.3 (таблица А.11).

Цель: отключение СБЗС-системы безопасности до того, как система выйдет за пределы заданных температурных режимов.

Описание: для отключения СБЗС-системы используется плавкий предохранитель. В PE-системе выключение осуществляется процедурой отключения электропитания, при которой сохраняется вся информация, необходимая при чрезвычайных действиях.

А.10.4 Ступенчатые сообщения от термодатчиков и условная тревога

Примечание — На этот метод/средство дана ссылка в ГОСТ Р 53195.3 (таблица А.11).

Цель: сообщение о том, что СБЗС-система (подсистема, модуль) вышла за установленные требованиями пределы температурных режимов.

Описание: осуществляется контроль температуры, и при ее выходе за установленные пределы выдается тревожное сообщение.

А.10.5 Соединение устройства принудительного охлаждения воздуха и индикатора состояний

Примечание — На этот метод/средство дана ссылка в ГОСТ Р 53195.3 (таблица А.11).

Цель: недопущение перегрева системы (подсистемы, модуля, среды) путем использования искусственного воздушного охлаждения.

Описание: осуществляется контроль температуры системы (подсистемы, модуля, среды), и, если температура превысила заданный предел, включается искусственное воздушное охлаждение. Пользователь информируется о состоянии температуры.

А.11 Обмен данными и запоминающее устройство большой емкости

Глобальная цель: контроль отказов в процессе обмена данными между внешними источниками и запоминающим устройством большой емкости.

А.11.1 Отделение линий электропитания от линий передачи информации

Примечание — На этот метод/средство дана ссылка в ГОСТ Р 53195.3 (таблица А.13).

Цель: минимизация перекрестных помех в линиях передачи информации, индуцируемых сильными токами системы электропитания.

Описание: линии, обеспечивающие электропитание СБЗС-систем, пространственно отделяют от линий, несущих информацию. Электромагнитное поле, которое может вызывать в линиях передачи информации всплески напряжений, уменьшается с расстоянием.

Данный метод/средство более подробно описан в [18].

А.11.2 Пространственное разделение групповых линий

Примечание — На этот метод/средство дана ссылка в ГОСТ Р 53195.3 (таблицы А.13 и А.17).

Цель: минимизация перекрестных помех, индуцируемых сильными токами в групповых линиях.

Описание: групповые линии, несущие информационные сигналы, отделяются одна от другой. Электромагнитное поле, которое может вызывать всплески напряжений в групповых линиях, уменьшается с расстоянием. Этот метод/средство снижает также отказы по общей причине.

Данный метод/средство более подробно описан в [18].

А.11.3 Повышение помехоустойчивости

Примечание — На этот метод/средство дана ссылка в ГОСТ Р 53195.3 (таблицы А.13, А.17 и А.19).

Цель: минимизация электромагнитных воздействий на СБЗС-систему.

Описание: применение методов/средств, уменьшающих восприимчивость СБЗС-системы к электромагнитным воздействиям, вызванным излучениями либо наводками на линии электропитания или информационные линии, а также возникающим из-за электростатических разрядов. К методам/средствам относятся: экранирование, заземление и фильтрация. Выбор применяемых методов/средств зависит от видов систем, конкретных областей применения и требований к системам. Следует ориентироваться на стандартизованные методы/средства, области применения и требования, которые описаны, например, в ГОСТ 13661, ГОСТ 30382, ГОСТ Р 51700 и других документах по стандартизации.

А.11.4 Передача сигнала без наводок

Примечание — На этот метод/средство дана ссылка в ГОСТ Р 53195.3 (таблицы А.13 и А.17).

Цель: обнаружение одних и тех же индуцированных напряжений в групповых линиях передачи сигналов.

Описание: вся дублируемая информация передается антивалентными сигналами (например, логическими 1 и 0 и их инверсиями). Ошибки по общей причине (например, вызванные электромагнитными излучениями) могут быть обнаружены антивалентным компаратором.

А.12 Датчики

Глобальная цель: контроль отказов в датчиках, связанных с безопасностью систем.

А.12.1 Эталонный датчик

Примечание — На это средство дана ссылка в ГОСТ Р 53195.3 (таблица А.14).

Цель: обнаружение неправильной работы датчика.

Описание: для контроля выполнения процессов датчика используется независимый эталонный датчик. Все входные сигналы в подходящие временные интервалы проверяются эталонным датчиком для обнаружения отказов в работе проверяемого датчика. Продуктивным является применение датчиков со встроенным эталоном или его цифровым образом.

Данное средство более подробно описано в [50—53].

А.12.2 Непосредственно управляемый переключатель

Примечание — На этот метод/средство дана ссылка в ГОСТ Р 53195.3 (таблица А.14).

Цель: размыкание контактов с помощью непосредственного механического соединения между кулачком переключателя и поводком контакта.

Описание: непосредственно управляемый переключатель размыкает свои обычно замкнутые контакты непосредственным механическим разъединителем между кулачком переключателя и поводком контакта. Этим обеспечивается размыкание контактов переключателя всякий раз, когда кулачок переключателя находится в рабочем положении. Принцип действия аналогичен принципу, описанному в А.1.2.

А.13 Оконечные элементы (приводы)

Глобальная цель: контроль отказов в оконечных элементах СБЗС-систем.

А.13.1 Мониторинг

Примечание — На этот метод/средство дана ссылка в ГОСТ Р 53195.3 (таблица А.15).

Цель: обнаружение неправильной работы привода.

Описание: работа привода контролируется (например, управляемыми контактами реле, см. мониторинг релейных контактов в А.1.2). Избыточность, вносимая этим контролем, может быть использована для переключения системы в аварийный режим.

А.13.2 Перекрестный контроль групповых приводов

Примечание — На этот метод/средство дана ссылка в ГОСТ Р 53195.3 (таблица А.15).

Цель: обнаружение отказов в приводах путем сравнения результатов контроля их работы.

Описание: каждый групповой привод контролируется своим аппаратным каналом. При обнаружении различий вырабатывается аварийное действие.

Приложение Б
(справочное)

Методы/средства для исключения систематических отказов СБЗС-систем (см. ГОСТ Р 53195.3 и ГОСТ Р 53195.4)

Примечания

1 Часть мер (методов/средств) данного приложения относятся к методам/средствам для программного обеспечения, но они не дублируют методы/средства, приведенные в приложении В.

2 Методы/средства, представленные в настоящем приложении, не являются исчерпывающими. СБЗС-системы непрерывно развиваются, и методы/средства исключения систематических отказов совершенствуются. При выборе конкретных методов/средств следует ориентироваться, в первую очередь, на стандартизованные методы/средства. В случае применения новых методов/средств всегда следует формировать и сохранять доказательственные материалы, демонстрирующие эффективность новых методов/средств по сравнению с методами/средствами, приведенными в настоящем приложении.

Б.1 Общие методы и средства

Б.1.1 Управление проектами

Примечание — На этот метод/средство дана ссылка в ГОСТ Р 53195.3 (таблицы Б.1—Б.6).

Цель: устранение отказов путем совершенствования организационной модели, правил и средств по разработке и тестированию СБЗС-систем.

Описание: наиболее значимыми являются методы/средства, направленные:

- на создание организационной модели, в основном для обеспечения действия систем менеджмента качества и менеджмента риска в соответствии с действующими стандартами;
- на установление в руководствах по взаимосвязанным проектам и конкретным проектам регулирующих правил, мер и мероприятий для создания и оценки соответствия СБЗС-систем;
- на применение следующих наиболее важных базовых принципов:
- выбор проектной организации и установление:
  - задач и ответственности подразделений организаций;
  - уполномочивающих подразделений по обеспечению качества;
  - порядка обеспечения независимого от разработчика подтверждения качества (внутреннее инспектирование);
- на определение и принятие плана последовательных действий (модели действий):
- определение всех существенных действий, необходимых во время выполнения проекта, включая внутреннее инспектирование проверки и график его проведения;
- дополнение (обновление) проекта;
- на определение стандартной последовательности для внутреннего инспектирования:
- планирование, выполнение и проверка инспектирования (теория инспектирования);
- разделение механизмов для комплектующей продукции;
- сохранение результатов повторных проверок;
- на управление конфигурацией:
- администрирование и проверка версий;
- обнаружение влияний модификаций;
- согласование инспектирования после модификаций;
- на введение количественной оценки мер по обеспечению качества:
- установление требований;
- статистика отказов;
- на введение автоматизированных универсальных методов, инструментов и средств обучения персонала.

Стандартизованные методы/средства по аспектам менеджмента качества, менеджмента риска и управления проектами подробно описаны в ГОСТ Р ИСО 9000, ГОСТ Р ИСО 9001, ГОСТ Р ИСО 10006, ГОСТ Р ИСО/МЭК 16085.

Б.1.2 Документация

Примечания

1 На этот метод/средство дана ссылка в ГОСТ Р 53195.3 (таблицы Б.1—Б.6).

2 См. также ГОСТ Р 53195.2 (раздел 5 и приложение А).

Цель: устранение отказов и упрощение оценки соответствия СБЗС-систем требованиям безопасности с помощью систем документирования каждого шага процесса разработки.

Описание: в процессе оценки соответствия все стороны, вовлеченные в процесс проектирования, должны демонстрировать (представлять доказательства) соответствие систем эксплуатационным характеристикам, требованиям безопасности и всем составляющим, включенным в проект. В обеспечении способности к тщательной разработке и гарантированию проверки доказательств безопасности в любой период времени особое значение придается документации.

Важными общими средствами устранения отказов и упрощения оценки соответствия являются введение руководств и автоматизация, в том числе:

- руководств, которые:
- устанавливают требования к групповому плану;
- имеют контрольный список содержания;
- устанавливают формат документа;
- администрирования документирования с помощью автоматизированной и организованной библиотеки проекта.

К индивидуальным средствам относятся:

- разделение в документации:
- требований к системе;
- описания системы (документации пользователя);
- описания разработки (включая внутреннюю инспекцию);
- группирование проектной документации в соответствии с жизненным циклом СБЗС-системы;
- установление стандартных модулей документации, из которых могут быть скомпилированы документы;
- ясная идентификация составляющих частей документаций;
- формализованное обновление версий;
- выбор ясных и понятных средств описания:
- формализованная нотация определений;
- естественный язык для введений, обоснований и представления намерений;
- графическое представление примеров;
- семантическое определение графических элементов;
- директории специальных слов.

Следует ориентироваться на стандартизованные требования к документации. Эти требования подробно описаны в [55], а также устанавливаются в документах по стандартизации системы программной документации.

Б.1.3 Разделение систем, связанных с безопасностью, и систем, не связанных с безопасностью

Примечание — На этот метод/средство дана ссылка в МЭК ГОСТ Р 53195.3 (таблицы Б.1 и Б.6).

Цель: предотвращение влияния связанных с безопасностью систем на системы, не связанные с безопасностью, в непредвиденных ситуациях.

Описание: в спецификации требований к СБЗС-системам должно быть определено, возможно ли разделение систем, связанных с безопасностью, и систем, не связанных с безопасностью. Должны быть установлены соответствующие четкие требования по этому вопросу. Четкое разделение требований снижает затраты на тестирование СБЗС-систем.

Более подробное описание этих требований приведено в [18, 54].

Б.1.4 Разнообразие аппаратных средств

Примечание — На этот метод/средство дана ссылка в ГОСТ Р 53195.3 (таблицы А.16, А.17 и А.19).

Цель: обнаружение систематических отказов при выполнении действий УО, с использованием разнообразных компонентов с различными частотами и типами отказов.

Описание: применение различных типов компонентов в разнообразных каналах СБЗС-систем для снижения вероятности отказов по общей причине (например, из-за перенапряжений, электромагнитных влияний) и повышения вероятности обнаружения таких отказов. Для выполнения требуемой функции применяют различные средства, основанные на различных физических принципах (например, электрических, гидравлических, пневматических) и различных способах решения одной и той же задачи. Существуют различные типы разнообразий. Для получения функционального разнообразия используют различные подходы для достижения одного и того же результата.

Б.2 Спецификация требований к безопасности СБЗС-систем

Глобальная цель: разработка спецификации требований к СБЗС-системе, которая, по возможности, должна быть полной, не содержащей ошибок, свободной от противоречий и простой для проверки.

Б.2.1 Структурирование спецификации

Примечание — На этот метод/средство дана ссылка в ГОСТ Р 53195.3 (таблицы Б.1 и Б.6).

Цель: уменьшение сложности спецификации требований к СБЗС-системе, а также исключение ошибок интерфейсов между требованиями путем создания иерархической структуры частичных требований.

Описание: метод/средство предусматривает структурирование функциональной спецификации требований с разбиением на частичные требования таким образом, чтобы между ними существовали, по возможности, простейшие наблюдаемые отношения. Анализ требований последовательно уточняют до тех пор, пока не будут получены различимые небольшие четкие частичные требования. Результатом последнего уточнения является иерархическая структура частичных требований, которая создает основу для спецификации полных требований. Этот метод/средство позволяет выделить интерфейсы частичных требований и особенно эффективен при использовании для исключения ошибок интерфейсов.

Более подробные описания стандартизованных структурируемых требований приведены в [55—58].

Б.2.2 Формальные методы

Примечание — На эти методы/средства дана ссылка в ГОСТ Р 53195.3 (таблицы Б.1 и Б.6).

Цель: выражать спецификацию требований однозначно и последовательно таким образом, чтобы оказалось возможным обнаружить ошибки и упущения.

Описание: формальные методы представляют собой методы/средства разработки и описания системы, применяемые на определенном этапе разработки спецификации или проекта. Результирующее описание принимает математическую форму и может быть подвергнуто математическому анализу для обнаружения различных классов несогласованностей или некорректностей. Такое описание может быть в некоторых случаях проанализировано на ЭВМ со строгостью, аналогичной строгости проверки компилятором синтаксиса исходной программы, или поддержано средствами анимации для изображения различных аспектов поведения описанной системы. Анимация улучшает восприятие человеком специфицированного поведения.

Формальные методы/средства могут в общем случае предоставлять нотацию (в основном некоторые методы дискретной математики), средства для вывода описания в этой нотации и различные виды анализа для проверки на корректность различных свойств описаний.

Начиная с математически формальной спецификации проектирование может быть сведено до последовательности пошаговых уточнений к проектированию логической схемы.

Более подробное описание формальных методов/средств приведено в ГОСТ Р МЭК 61160.

Б.2.3 Полуформальные методы

Цель: создание однозначных и согласованных частей спецификации с возможностью обнаружения ошибок и пропусков.

Примечание — На эти методы/средства дана ссылка в ГОСТ Р 53195.3 (таблицы Б.1, Б.2 и Б.6) и в ГОСТ Р 53195.4 (таблицы А.1, А.2 и А.4).

Б.2.3.1 Общие положения

Цель: удостовериться в том, что проект удовлетворяет своей спецификации.

Описание: полуформальные методы представляют собой методы/средства создания описания системы на стадиях ее разработки, например при разработке спецификации, при проектировании или кодировании. Описание может быть в некоторых случаях проанализировано на ЭВМ или поддержано средствами анимации для отображения различных аспектов поведения системы. Анимация позволяет получить дополнительную уверенность в том, что система удовлетворяет как реальным требованиям, так и специфицированным требованиям.

Два полуформализованных метода описаны ниже.

Б.2.3.2 Метод конечных автоматов/диаграммы переходов состояний

Примечание — На этот метод/средство дана ссылка в ГОСТ Р 53195.4 (таблицы Б.5 и Б.7).

Цель: моделирование, подготовка спецификации или реализация структуры управления системы.

Описание: системы могут быть описаны в выражениях, отображающих их состояния, данные на их входах и их действия. Находясь в состоянии С1 и при получении на входе данных, система может выполнить действие А и перейти в состояние С2. Путем описания системных действий для каждого входа в каждом состоянии можно описать систему полностью. Образуемая в результате модель системы является автоматом конечных состояний. Она может быть изображена в виде так называемой диаграммы переходов состояний, которая показывает, каким образом система переходит из одного состояния в другое, или в виде матрицы, в которой по осям задаются состояния и входы, а ячейки матрицы содержат действия по переходу в новое состояние.

Когда система усложняется или имеет естественную структуру, это может быть отражено в уровневой структуре автомата конечных состояний.

Спецификация или проект, выраженный в виде автомата конечных состояний, может быть проверен:

- на полноту (система должна иметь действие и новое состояние для каждого входа в каждом состоянии);
- на согласованность (только одно изменение состояния описывается для каждой пары состояние/вход);
- на достижимость (можно или нельзя перейти из одного состояния в другое при любой последовательности входов).

Это важные свойства для критических систем. Инструменты для обеспечения этих проверок легко разработать. Существуют также алгоритмы, которые позволяют автоматически генерировать тестовые примеры для верификации реализаций автомата конечных состояний или для анимации модели автомата конечных состояний.

Подробное описание данных методов/средств приведено в [57—59].

Б.2.3.3 Метод сетей Петри

Примечание — На этот метод/средство дана ссылка в ГОСТ Р 53195.4 (таблицы Б.5 и Б.7).

Цель: моделирование соответствующих аспектов поведения системы, оценка и, возможно, повышение ее безопасности и рабочих характеристик путем анализа и повторного проектирования.

Описание: сети Петри относятся к классу теоретических графовых моделей, используемых для представления информации и управления потоками в системах, процессы в которых конкурентны и асинхронны.

Сеть Петри — это сеть позиций и переходов. Позиции могут быть «маркированными» или «немаркированными». Переход «активизирован», когда все его входы маркированы. В активизированном состоянии позиции разрешается (но не требуется) быть «возбужденной». Если позиция «возбуждена», вход, поступающий на переход, становится немаркированным, а вместо этого каждый выход из перехода оказывается маркированным.

Потенциальные опасности могут быть представлены в виде конкретных состояний (маркировок) в модели. Модель сетей Петри может быть расширена для обеспечения возможности синхронизации системы. И хотя «классические» сети Петри концентрируются на аспектах управления потоками, имеются некоторые расширения для включения потока данных в модель.

Подробное описание сетей Петри приведено в [60—65].

Б.2.4 Автоматизированные средства разработки спецификации

Примечание — На этот метод/средство дана ссылка в ГОСТ Р 53195.3 (таблицы Б.1 и Б.6) и в ГОСТ Р 53195.4 (таблицы А.1 и А.2).

Б.2.4.1 Общие положения

Цель: использование формальных методов/средств разработки спецификации для упрощения автоматического обнаружения неоднозначностей и полноты системы.

Описание: метод/средство предусматривает создание спецификации в форме базы данных, которая может автоматически анализироваться для оценки согласованности и полноты. Инструмент созданной таким образом спецификации предоставляет пользователю возможность применить анимацию различных аспектов специфицированной системы. В общем случае метод/средство поддерживает не только создание спецификаций, но и этап проектирования, а также и другие этапы жизненного цикла. Инструменты спецификаций могут быть классифицированы в соответствии с пунктами, приведенными ниже.

Подробное описание этих методов/средств приведено в [57—59].

Б.2.4.2 Инструменты, не ориентированные на конкретный метод

Цель: предоставление пользователю с помощью подсказок и формирования связей между соответствующими частями возможности составления правильной спецификации.

Описание: инструменты, не ориентированные на конкретный метод, освобождают пользователя от рутинной процедуры и поддерживают управление проектом. Они не формируют какую-либо конкретную методологию разработки спецификаций. Относительная независимость от метода позволяет пользователям быть более свободными и одновременно дает им некоторую специализированную поддержку, необходимую при создании спецификаций. При этом усложняется освоение метода.

Подробное описание этих методов/средств приведено в [57—59].

Б.2.4.3 Процедура, ориентированная на модель с иерархическим анализом

Цель: предоставление пользователю возможности создания правильной спецификации, обеспечив согласованность между описаниями действий и данных на различных уровнях абстрагирования.

Описание: метод/средство дает функциональное представление о необходимой системе (структурный анализ) на различных уровнях абстрагирования (степень точности). Анализ проводится на различных уровнях как с действиями, так и с данными. Оценка неоднозначности и полноты спецификации возможна между иерархическими уровнями, а также между двумя функциональными единицами (модулями) на одном и том же уровне.

Подробное описание метода/средства приведено в [57—59].

Б.2.4.4 Сущностные модели

Цель: предоставление пользователю возможности создания правильной спецификации, фокусируя внимание на использовании сущностей внутри системы и взаимоотношений между ними.

Описание: заданная система описывается в виде совокупности объектов и взаимоотношений между ними. Применение сущностных моделей позволяет определять, какие взаимоотношения могут интерпретироваться системой. В общем случае эти взаимоотношения позволяют описывать иерархическую структуру объектов, поток данных, взаимоотношения между данными и определять, какие данные являются предметом определенных технологических процессов. Классическая процедура расширяется применением управления процессами. Возможности обследования спецификации и поддержка пользователя зависят от разнообразия проиллюстрированных взаимоотношений. С другой стороны, множество представленных возможностей делает применение этого метода сложным.

Подробное описание метода/средства приведено в [66—68].

Б.2.4.5 Стимул и отклик

Цель: предоставление пользователю возможности создания правильной спецификации путем идентификации взаимоотношений «стимул — отклик».

Описание: взаимоотношения между объектами системы определены в нотациях «стимулы» и «отклики». Используется простой и легко расширяемый язык, который содержит элементы языка, представляющие объекты, взаимоотношения, характеристики и структуры.

Подробное описание данного метода/средства приведено в [69, 70].

Б.2.5 Таблица контрольных проверок

Примечание — На этот метод/средство дана ссылка в ГОСТ Р 53195.3 (таблицы Б.1, Б.2 и Б.6) и в ГОСТ Р 53195.4 (таблицы А.10 и Б.8).

Цель: сосредоточение внимания пользователя на всех важных аспектах системы на конкретной стадии жизненного цикла системы и управление критическими оценками, обеспечивая исчерпывающий охват без предъявления точных требований к системе.

Описание: метод/средство включает в свой состав набор вопросов, на которые должно дать ответ лицо, заполняющее таблицу контрольных проверок. Многие вопросы носят общий характер, и эксперт должен интерпретировать их как наиболее подходящие к конкретной оцениваемой системе. Таблицу контрольных проверок можно использовать на всех этапах всего жизненного цикла СБЗС-системы, жизненных циклов аппаратных средств и программного обеспечения. Метод/средство полезно в качестве инструмента, способствующего оценке функциональной безопасности.

Для сокращения широкого разнообразия проходящих подтверждение соответствия систем большинство таблиц контрольных проверок содержат вопросы, которые применимы ко многим типам систем. В результате в используемой таблице контрольных проверок может оказаться множество вопросов, которые не уместны в используемой системе и которые должны игнорироваться. В то же время для конкретной системы может возникнуть необходимость дополнения стандартной таблицы контрольных проверок вопросами, специально ориентированными на используемую систему.

Использование таблицы контрольных проверок в значительной степени зависит от экспертной оценки и суждения инженера, который выбирает и применяет таблицу контрольных проверок. В результате решения, принятые инженером относительно выбранных(ой) таблиц(ы) контрольных проверок, и любые дополнительные вопросы должны быть полностью задокументированы и обоснованы. Цель состоит в гарантировании возможности контроля применения таблиц контрольных проверок и того, что при использовании одних и тех же критериев будут получены одни и те же результаты.

Объект в заполненной таблице контрольных проверок должен быть как можно более сжатым. При необходимости исчерпывающего обоснования оно должно быть дано в виде ссылок на дополнительные документы. Для документирования результатов каждого вопроса должен использоваться ответ: «успешно», «безуспешно» или «не завершено» — либо некоторый аналогичный ограниченный набор ответов. Эта лаконичность существенно упрощает процедуру достижения общего заключения в виде результатов оценки таблицы контрольных проверок.

Б.2.6 Экспертиза спецификации

Примечание — На этот метод/средство дана ссылка в ГОСТ Р 53195.3 (таблицы Б.1 и Б.6).

Цель: исключение некомплектности и противоречивости спецификации.

Описание: общий метод, с помощью которого оценивается спецификация с различных сторон независимой группой экспертов. Группа экспертов задает вопросы разработчику спецификации, который должен дать ей удовлетворительные ответы. Анализ должен (по возможности) выполняться группой, которая не принимала участия в создании спецификации. Требуемая степень независимости определяется уровнями полноты безопасности, задаваемыми для системы. Независимые эксперты должны быть способны реконструировать эксплуатационную функцию системы бесспорным способом без ссылок на любые последующие спецификации. Они должны также убедиться в том, что охвачены все уместные аспекты безопасности и технические аспекты в эксплуатационных и организационных средствах. Эта процедура доказала свою высокую эффективность на практике.

Более подробное описание метода/средства приведено в ГОСТ Р МЭК 61160.

Б.3 Проектирование и разработка СБЗС-систем

Глобальная цель: создание проекта СБЗС-системы в соответствии со спецификацией.

Б.3.1 Соблюдение стандартов и руководств

Примечание — На этот метод/средство дана ссылка в ГОСТ Р 53195.3 (таблица Б.2).

Цель: рассмотрение стандартов сектора применения (не устанавливаемых в настоящем стандарте) на предмет правильного их использования при разработке и проектировании СБЗС-системы.

Описание: при проектировании СБЗС-системы должны составляться руководства. Эти руководящие материалы должны, во-первых, приводить к созданию СБЗС-систем, которые практически свободны от ошибок, и, во-вторых, упрощать последующее подтверждение соответствия требованиям безопасности. Они могут быть универсальными, установленными только для данного проекта или установленными только для отдельного его этапа.

Б.3.2 Структурное проектирование

Примечание — На этот метод/средство дана ссылка в ГОСТ Р 53195.3 (таблицы Б.2 и Б.6).

Цель: снижение сложности спецификации требований путем создания иерархической структуры частичных требований; исключение ошибок взаимосвязей между требованиями; упрощение верификации.

Описание: при проектировании аппаратных средств должны использоваться конкретные критерии или методы. Например, может потребоваться следующее:

- проектирование иерархически структурированных схем;
- использование изготовленных и прошедших тестирование частей схем.

Аналогично при проектировании программных средств использование структурных схем позволяет создать однозначную структуру программных модулей. Эта структура показывает взаимосвязь модулей друг с другом, конкретные данные, которые передаются между модулями, и конкретное управление, существующее между модулями.

Структурное проектирование более подробно описано в [70, 71].

Б.3.3 Использование надежно испытанных компонентов

Примечание — На этот метод/средство дана ссылка в ГОСТ Р 53195.3 (таблицы Б.2 и Б.6).

Цель: снижение риска появления ряда необнаруживаемых отказов путем использования компонентов с конкретными характеристиками.

Описание: выбор надежно испытанных компонентов для целей безопасности выполняется производителем в соответствии с надежностью компонентов (например, использование эксплуатационно-тестируемых физических модулей для удовлетворения высоких требований безопасности или хранение относящихся к безопасности программ только в безопасной памяти). Обеспечение безопасности памяти может касаться устранения несанкционированного доступа, влияний несанкционированной среды (электромагнитные воздействия, радиация и т. д.), а также отклика компонентов в случае обнаружения отказов.

Более подробное описание этого метода/средства приведено в [72].

Б.3.4 Модульное проектирование

Примечание — На этот метод/средство дана ссылка в ГОСТ Р 53195.3 (таблицы Б.2 и Б.6).

Цель: снижение сложности проектирования и исключение ошибок, связанных с интерфейсами между подсистемами.

Описание: каждая подсистема на всех уровнях проектирования четко определяется и ограничивается по размеру (только небольшим набором функций). Интерфейсы между подсистемами поддерживаются как можно более простыми, и пересечения (то есть разделяемые данные, обмен информацией) минимизируются. Сложность отдельных подсистем также ограничивается.

Более подробное описание метода/средства приведено в [73, 74].

Б.3.5 Средства автоматизированного проектирования

Примечание — На эти средства дана ссылка в ГОСТ Р 53195.3 (таблицы Б.2 и Б.6) и в ГОСТ Р 53195.4 (таблица А.4).

Цель: обеспечение наиболее систематического выполнения процедур проектирования. Включение в проект подходящих автоматически спроектированных элементов, которые уже созданы и проверены.

Описание: инструменты автоматизированного проектирования (САПР) должны использоваться в процессе проектирования как АС, так и ПО во всех случаях, когда они доступны и обоснованы сложностью системы. Правильность выбора таких инструментов должна быть продемонстрирована конкретным тестированием, обширной предысторией удовлетворительного использования либо независимой верификацией результата их применения для конкретной проектируемой СБЗС-системы.

Более подробно методы/средства автоматизированного проектирования описаны в [75, 76].

Б.3.6 Моделирование

Примечание — На этот метод/средство дана ссылка в ГОСТ Р 53195.3 (таблицы Б.2, Б.5 и Б.6).

Цель: выполнение систематической и полной проверки электрических/электронных схем, их функционирования, а также корректное задание размеров их компонентов.

Описание: функция схемы, реализующая СБЗС-систему, имитируется на компьютере с помощью запрограммированной модели ее поведения. Поведение каждого отдельного компонента схемы моделируется отдельно, и отклик схемы, в которую он входит, анализируется при задании предельных данных для каждого компонента.

Этот метод/средство более подробно описан в [75, 76].

Б.3.7 Проверка (обзор и анализ)

Примечание — На этот метод/средство дана ссылка в ГОСТ Р 53195.3 (таблицы Б.2 и Б.6).

Цель: выявление рассогласований между спецификацией и реализацией.

Описание: заданные функции СБЗС-системы проверяют и оценивают для гарантирования того, что СБЗС-система соответствует требованиям, приведенным в спецификации. Все сомнительные вопросы относительно реализации и использования системы документируются с целью их последующего разрешения. В отличие от сквозного анализа во время процедуры проверки автор пассивен, а эксперт активен.

Б.3.8 Сквозной анализ

Примечание — На этот метод/средство дана ссылка в ГОСТ Р 53195.3 (таблица Б.6).

Цель: выявление рассогласования между спецификацией и реализацией.

Описание: заданные функции СБЗС-системы оцениваются для гарантиирования того, что СБЗС-система соответствует требованиям, приведенным в спецификации. Все сомнительные вопросы относительно реализации и использования системы документируются с целью их последующего разрешения. В отличие от процедуры проверки во время сквозного анализа автор проекта активен, а эксперт пассивен.

Б.4 Процедуры эксплуатации и обслуживания СБЗС-систем

Глобальная цель: разработка процедур, исключающих ошибки во время эксплуатации и обслуживания СБЗС-систем.

Б.4.1 Инструкции по эксплуатации и обслуживанию

Примечание — На этот метод/средство дана ссылка в ГОСТ Р 53195.3 (таблица Б.4).

Цель: минимизация ошибок во время эксплуатации и обслуживания СБЗС-систем.

Описание: инструкции пользователя содержат важную информацию о способах использования и обслуживания систем. В особых случаях эти инструкции могут содержать также примеры общих способов установки СБЗС-систем. Все инструкции должны быть легко воспринимаемыми. Для описания сложных процедур и зависимостей должны использоваться рисунки и схемы.

Б.4.2 Удобство общения пользователя с системой

Примечание — На этот метод/средство дана ссылка в ГОСТ Р 53195.3 (таблица Б.4).

Цель: снижение сложности действий, выполняемых оператором во время эксплуатации СБЗС-систем.

Описание: правильность эксплуатации СБЗС-систем может зависеть в некоторой степени от оператора. Рассматривая соответствующий проект системы и проект рабочего места, разработчик СБЗС-систем должен предусмотреть меры, чтобы в период эксплуатации систем обеспечивалось следующее:

- необходимость вмешательства оператора в действия системы была ограничена абсолютным минимумом;
- необходимое вмешательство оператора было как можно более простым;
- возможность ущерба от ошибок оператора была минимизирована;
- средства вмешательства и средства индикации были спроектированы в соответствии с эргономическими требованиями;
- средства оператора были простыми, четко обозначенными и естественными для использования;
- оператор не был перенапряжен даже в экстремальной ситуации;
- обучение процедурам и средствам процесса вмешательства было адаптировано к уровню знаний и мотивации обучаемого оператора (пользователя).

Б.4.3 Удобство общения обслуживающего персонала с системой

Примечание — На этот метод/средство дана ссылка в ГОСТ Р 53195.3 (таблица Б.4).

Цель: упрощение процедуры обслуживания СБЗС-системы и проектирование необходимых средств для эффективной диагностики и ремонта.

Описание: профилактическое техническое обслуживание и ремонт СБЗС-системы часто производится в жестких временных рамках. Поэтому разработчик систем должен предусмотреть меры, чтобы в период эксплуатации обеспечивалось следующее:

- средства, относящиеся к техническому обслуживанию СБЗС-системы, требовались как можно реже или в идеале вообще не требовались;
- использовались достаточно чувствительные и легко управляемые диагностирующие инструменты для неизбежных ремонтов; эти инструменты должны включать в себя все необходимые интерфейсы;
- если отдельные инструменты диагностики должны быть разработаны или приобретены, то для этого должно быть достаточно времени.

Б.4.4 Сокращение работ на стадии эксплуатации

Примечание — На этот метод/средство дана ссылка в ГОСТ Р 53195.3 (таблицы Б.4 и Б.6).

Цель: ограничение эксплуатационных возможностей для обычного пользователя.

Описание: эксплуатационные возможности ограничивают путем:

- ограничения числа требуемых операций в рабочих режимах, например благодаря применению пультов управления;
- ограничения числа используемых в работе элементов контроля и управления;
- ограничения числа возможных в обычных условиях эксплуатации рабочих режимов.

Б.4.5 Эксплуатация только квалифицированным оператором

Примечание — На этот метод/средство дана ссылка в ГОСТ Р 53195.3 (таблицы Б.4 и Б.6).

Цель: исключение отказов, обусловленных ошибками оператора.

Описание: оператор СБЗС-системы должен быть обучен до такой степени, которая соответствует сложности и уровню безопасности СБЗС-системы. В обучение входит изучение основ технологического процесса эксплуатации СБЗС-системы и взаимосвязей между СБЗС-системой и УО.

Б.4.6 Защита от ошибок оператора

Примечание — На этот метод/средство дана ссылка в ГОСТ Р 53195.3 (таблица Б.6).

Цель: защищенность системы от всех прогнозируемых видов ошибок оператора.

Описание: ложные входные сообщения (о значениях параметров, времени и т. д.) обнаруживаются проверками правильности работы или мониторингом УО. Для объединения этих возможностей в проекте на самом раннем этапе проектирования необходимо установить, какие из входных сообщений возможны и какие допустимы.

Б.4.7 Защита от модификаций

Примечание — На этот метод/средство дана ссылка в ГОСТ Р 53195.3 (таблица А.18).

Цель: защита СБЗС-системы от модификаций аппаратных средств техническими средствами.

Описание: модификации или манипуляции должны обнаруживаться автоматически, например проверками правильности сигналов от датчиков, техническим процессом и автоматическим запуском тестирования. При обнаружении модификации должен выдаваться сигнал аварии.

Б.4.8 Подтверждение ввода

Примечание — На этот метод/средство дана ссылка в ГОСТ Р 53195.3 (таблицы А.18 и А.19).

Цель: обеспечение обнаружения ошибки во время работы самим оператором до активизации УО.

Описание: ввод данных управления в УО через СБЗС-систему отображается оператору до передачи их в УО, с тем чтобы оператор имел возможность обнаружить и исправить ошибки. Проект системы должен как реагировать на неправильные, неспровоцированные действия оператора, так и учитывать нижние/верхние пределы скорости и направление реакции человека. Этим можно исключить, например, более быстрое, чем предполагается, нажатие клавиш оператором и настроить систему на восприятие двойного нажатия клавиши как одинарное или как повторное из-за того, что система (например, изображение на экране) слишком медленно реагирует на первое нажатие клавиши. Последовательное нажатие одной и той же клавиши не должно действовать более одного раза при вводе критических данных; нажатие клавиши «ввод» (*enter*) или «да» (*yes*) неограниченное количество раз не должно приводить к нарушению безопасности системы.

Должны быть предусмотрены процедуры временных пауз с возможностью выбора разных ответов («да»/«нет» и т. п.), чтобы обеспечить возможность для размышления оператору, а системе — режим ожидания.

Недостаточная способность к перезагрузке СБЗС-системы делает ее уязвимой, если только программные/аппаратные средства не спроектированы с учетом этой ситуации.

Этот метод/средство более подробно описан в [77].

Б.5 Интеграция E/E/PE СБЗС-систем

Общая цель: исключение отказов на стадии интеграции и обнаружение любых отказов во время этой и предыдущих стадий.

Б.5.1 Функциональное тестирование

Примечание — На этот метод/средство дана ссылка в ГОСТ Р 53195.3 (таблица Б.3) и в ГОСТ Р 53195.4 (таблицы А.5 — А.7).

Цель: обнаружение отказов на стадиях создания спецификации и проектирования. Исключение отказов во время реализации и интеграция программных и аппаратных средств.

Описание: в процессе функционального тестирования определяется, достигнуты ли заданные характеристики системы. В систему поступают входные данные, которые адекватно характеризуют обычное выполнение операций. Наблюдаемые выходные результаты сравниваются с результатами, заданными в спецификации. Отклонения от спецификации и указания на неполноту спецификации документируются.

Функциональное тестирование электронных компонентов, предназначенных для многоканальной структуры, обычно включает в себя тестирование и покупных промышленных компонентов, по каждому из которых производитель (поставщик) уже провел тестирование и предварительно подтвердил соответствие. Помимо этого, рекомендуется, чтобы покупные промышленные компоненты были протестированы в сочетании с другими сопрягаемыми компонентами той же партии для выявления неисправностей группового типа, которые в противном случае могли бы остаться необнаруженными.

В целом для того, чтобы рабочие возможности системы были достаточными, следует выполнять рекомендации, приведенные в В.5.20 настоящего стандарта.

Более подробно этот метод/средство описан в [78—81].

Б.5.2 Тестирование методом «черного ящика»

Примечание — На этот метод/средство дана ссылка в ГОСТ Р 53195.3 (таблицы Б.3, Б.5 и Б.6) и в ГОСТ Р 53195.4 (таблицы А.5 — А.7).

Цель: проверка динамического поведения системы в реальных условиях функционирования; выявление несоответствия функциональной спецификации и оценка полезности и устойчивости.

Описание: функции системы или программы выполняются в заданном окружении с заданными показателями тестирования, которые систематически формируются из спецификации в соответствии с установленными критериями. Этим выявляется поведение системы и допускается возможность ее сравнения со спецификацией. При проведении тестирования никакие сведения о внутренней структуре системы не используются. Проверкой определяется правильность выполнения функциональным модулем всех функций, предусмотренных спецификацией. Метод формирования эквивалентных классов служит примером критерия тестирования данных методом «черного ящика». Массив входных данных подразделяется на конкретные диапазоны входных значений (эквивалентные классы) на основе спецификации. После этого формируются тестовые примеры с использованием:

- данных из допустимых диапазонов;
- данных из недопустимых диапазонов;
- данных предельных значений диапазонов;
- экстремальных значений;
- комбинаций перечисленных выше классов.

Эффективными могут оказаться и другие критерии для выбора тестовых примеров в различных режимах тестирования (тестирование модуля, тестирование интеграции и тестирование системы). Например, критерий «экстремальные эксплуатационные условия» используется при тестировании системы в процессе подтверждения соответствия.

Более подробное описание этого метода/средства приведено в [62—64].

Б.5.3 Статистическое тестирование

Примечание — На этот метод/средство дана ссылка в ГОСТ Р 53195.3 (таблицы Б.3, Б.5 и Б.6).

Цель: проверка динамического поведения СБЗС-системы и оценка ее полезности и устойчивости.

Описание: статистическое тестирование выполняется с входными данными, выбранными в соответствии с операционным (эксплуатационным) профилем, который отражает частоту выполнения пользователем различных операционных сценариев работы. Именно операционный профиль «руководит» выбором тестов, результаты которых образуют статистическую выборку для выполнения контроля. Это означает, что первыми будут обнаруживаться отказы наиболее часто используемых компонентов системы и устраняться дефекты в наиболее часто выполняемых фрагментах кода.

Более подробное описание этого метода/средства приведено в [84—87].

Б.5.4 Натурные испытания

Примечания

1 На этот метод/средство дана ссылка в ГОСТ Р 53195.3 (таблицы Б.3 и Б.5).

2 В контексте ПО см. также аналогичные средства в В.2.10 настоящего стандарта, а в приложении Г — статистический подход.

Цель: использование натурных (полевых) испытаний в качестве одного из средств исключения неисправностей во время интеграции СБЗС-систем и/или в процессе подтверждения соответствия требованиям безопасности СБЗС-систем.

Описание: применение компонентов или подсистем, которые при их использовании показали путем испытаний полное отсутствие ошибок или наличие только несущественных ошибок и несущественные их изменения в течение достаточно длительного периода времени эксплуатации во многих различных применениях. В частности, для сложных компонентов с множеством функций (например, операционные системы, интегральные схемы) разработчик должен обратить внимание на те функции, которые были фактически протестированы методом натурных испытаний. Например, рассмотреть подпрограммы самотестирования для обнаружения неисправностей при отсутствии в период эксплуатации неисправностей аппаратных средств. О подпрограммах нельзя сказать, что они протестированы, поскольку они никогда не выполняли функций обнаружения своих неисправностей.

При использовании натурных испытаний должны быть соблюдены следующие требования:

- неизменность спецификации;
- 10 систем в различных применениях;
- 10^5 часов работы и по меньшей мере один год сервисной поддержки.

Натурные испытания документируются производителем (поставщиком) и/или эксплуатирующей компанией. Эта документация должна содержать, по меньшей мере, следующие данные:

- точное обозначение (идентификацию) системы и ее компонентов, включая компоненты управления версией АС;
- сведения о пользователях и времени применения;
- время наработки в часах;

- процедуры выбора системы и прикладные программы, использованные при испытаниях;
- процедуры обнаружения, регистрации и устранения неисправностей, а также процедуры устранения последствий и причин их возникновения.

Данный метод/средство применим в большей степени к отдельным составляющим (подсистемам) СБЗС-систем. Его более подробное описание приведено в [88, 89].

Б.6 Оценка соответствия СБЗС-системы требованиям безопасности

Глобальная цель: оценка соответствия СБЗС-системы спецификации требований безопасности.

Б.6.1 Функциональное испытание в условиях окружающей среды

Примечание — На этот метод/средство дана ссылка в ГОСТ Р 53195.3 (таблица Б.5).

Цель: оценка, защищена ли СБЗС-система от типичных воздействий окружающей среды.

Описание: система подвергается воздействию окружающей среды при различных условиях.

Стандартизованные требования к ряду воздействий подробно описаны в [58—60, 62, 90].

Б.6.2 Испытание на устойчивость к пиковым выбросам внешних воздействий

Примечание — На этот метод/средство дана ссылка в ГОСТ Р 53195.3 (таблицы Б.5 и Б.6).

Цель: проверка способности СБЗС-систем в условиях пиковых выбросов внешних электромагнитных воздействий.

Описание: система загружается типичной прикладной программой, и все периферийные линии (все цифровые, аналоговые и последовательные интерфейсы, шины, источники питания и т. д.) подвергаются воздействию стандартных электромагнитных помех. Для получения количественной оценки целесообразно очень внимательно подходить к предельным значениям выбросов внешних воздействий. Выбранный класс помех окажется неподходящим, если функция не выполняется.

Б.6.3 Статический анализ

Примечание — На этот метод/средство дана ссылка в ГОСТ Р 53195.3 (таблицы Б.5 и Б.6) и в ГОСТ Р 53195.4 (таблица А.9).

Цель: исключение систематических неисправностей, которые могут приводить к отказам в тестируемой системе вначале или после многих лет эксплуатации.

Описание: этот систематический и, по возможности, автоматизированный подход предусматривают исследование конкретных статических характеристик опытных образцов составляющих системы для обеспечения полноты, согласованности, отсутствия неоднозначностей в сформулированных требованиях (например, в руководящих материалах по конструкции, системных спецификациях и в перечне данных о применении). Статический анализ воспроизводим. Он применим к опытному образцу, который воспроизводим и четко определяет завершающий этап. Примерами статического анализа АС и ПО являются:

- анализ согласованности потока данных;
- анализ управления потоком (определение маршрутов, определение кода недоступности);
- анализ интерфейсов (исследование передачи переменных между различными программными модулями);
- анализ потока данных для обнаружения вызывающих сомнения последовательностей по созданию, ссылкам и удалению переменных;
- применение тестирования к конкретным руководящим материалам (например, по вопросам: длина пути утечки тока и зазоры, расстояние между совокупностями модулей, расположение физических модулей, механически чувствительные физические модули, индивидуальное использование физических модулей при их внедрении).

Б.6.4 Динамический анализ

Примечание — На этот метод/средство дана ссылка в ГОСТ Р 53195.3 (таблицы Б.5 и Б.6) и в ГОСТ Р 53195.4 (таблицы А.5 и А.9).

Цель: обнаружение ошибок в спецификации путем исследования динамического поведения опытных образцов, составляющих системы.

Описание: динамический анализ СБЗС-систем выполняется, если подавать на вход близкой к эксплуатационному образцу элемента (модуля) системы, связанной с безопасностью, входные данные, типичные для заданного эксплуатационного окружения. Анализ признается удовлетворительным, если наблюдаемое поведение СБЗС-системы соответствует требуемому поведению. Любой отказ СБЗС-системы должен быть устранен, после чего следует проанализировать новые варианты эксплуатации.

Б.6.5 Анализ отказов

Примечание — На этот метод/средство дана ссылка в ГОСТ Р 53195.3 (таблицы Б.5 и Б.6).

Б.6.5.1 Виды отказов и анализ их последствий

Цель: проведение анализа проекта системы с исследованием всех возможных источников отказов компонентов системы и определением влияния этих отказов на поведение и безопасность системы.

Описание: анализ обычно производится совещанием инженеров. Каждый компонент системы анализируется по очереди для выявления набора режимов отказов, их причин и результатов, определения процедуры обнаружения и выдачи рекомендаций. При выдаче рекомендаций они документируются в виде корректирующих действий.

Виды отказов и анализ их последствий подробно описаны в [89—91].

Б.6.5.2 Диаграммы последовательностей событий

Примечание — На этот метод/средство дана ссылка в ГОСТ Р 53195.4 (таблицы А.10, Б.3 и Б.4).

Цель: моделирование СБЗС-системы с помощью диаграмм последовательностей событий для представления проекта системы в виде последовательности комбинаций базовых событий.

Описание: это средство может рассматриваться как комбинация анализа на основе дерева неисправностей и анализа на основе дерева событий. Начиная с критических событий, граф последовательностей причин проходит в прямом и обратном направлениях. Прохождение в обратном направлении эквивалентно дереву неисправностей, где критическое событие представлено в виде представленного верхнего события. Прохождение в прямом направлении позволяет определять возможные последствия, возникающие из события. В узле графа могут быть символы, которые описывают условия распространения причин по различным ветвям от этого узла. Временны́е задержки также могут учитываться. Эти условия также могут быть описаны с помощью деревьев неисправностей. Чтобы диаграмма выглядела более компактной, пути распространения могут быть объединены с логическими символами. Для использования в диаграммах последовательностей причин используются стандартные символы. Такие диаграммы могут быть применены для вычисления вероятности появления определенных критических последовательностей.

Диаграммы последовательности событий более подробно описаны в [92, 93].

Б.6.5.3 Анализ дерева событий

Примечание — На этот метод/средство дана ссылка в ГОСТ Р 53195.4 (таблица Б.4).

Цель: моделирование с помощью диаграмм последовательности событий, которая может возникать в системе после инициализирующего события, для установления серьезности возможных последовательностей.

Описание: в верхней части диаграммы записывают последовательность условий, которая соответствует последовательности событий, происходящих после инициализирующего события. Начиная с инициализирующего события, которое является целью анализа, проводят линию к первому условию последовательности. Наличие у диаграммы ветвей «да» и «нет» указывает, каким образом будущие события зависят от условий. Каждая из этих ветвей продолжается к следующему условию. Однако не все условия пригодны для всех ветвей. Какая-то из них продолжится до окончания последовательности условий, но каждая ветвь дерева, сконструированная таким способом, представляет возможную последовательность. Дерево событий может быть использовано для вычисления вероятности различных последовательностей с учетом значений вероятности и числа условий в последовательности.

Более подробное описание этого метода/средства приведено в ГОСТ 27.310 и [94].

Б.6.5.4 Виды неисправностей, анализ влияний и анализ критичности

Примечание — На этот метод/средство дана ссылка в ГОСТ Р 53195.4 (таблицы А.10 и Б.4).

Цель: ранжирование критичности компонентов, могущих вызвать нарушения, повреждения или ухудшение работы СБЗС-системы при одиночных ошибках, для определения, каким компонентам может потребоваться особое внимание и какие средства управления необходимы для обеспечения процессов проектирования или эксплуатации.

Описание: критичность компонентов может быть ранжирована многими способами. В процедуре ранжирования критичности компонентов величина критичности для любого компонента зависит от числа определенного вида отказов, предполагаемых в процессе выполнения каждого миллиона операций, реализуемых в критическом режиме. Величина критичности является функцией девяти параметров, большинство из которых должны быть измерены. Простой метод определения критичности состоит в умножении вероятности отказа компонента на ущерб, который может быть причинен этим отказом. Этот метод аналогичен простой оценке степени риска.

Более подробное описание данного метода/средства приведено в [95].

Б.6.5.5 Анализ дерева неисправностей

Примечание — На этот метод/средство дана ссылка в ГОСТ Р 53195.4 (таблица Б.4).

Цель: упрощение анализа событий или комбинаций событий, вызывающих опасности или серьезные последствия.

Описание: начиная с события, которое может непосредственно вызвать опасность или серьезные последствия («вершины дерева событий»), анализ выполняют по ветвям дерева. Комбинации причины описываются логическими операторами (И, ИЛИ, НЕ и т. п.). Затем анализируют промежуточные причины тем же способом и т. д., возвращаясь к базовым событиям, по достижении которых анализ прекращают.

Этот метод является графическим, и для изображения дерева неисправностей используют набор стандартизованных символов. Метод предназначен в основном для анализа АС, но следует, по возможности, применять его к анализу отказов ПО.

Более подробное описание данного метода/средства приведено в [96].

Б.6.6 Анализ наихудшего случая

Примечание — На этот метод/средство дана ссылка в ГОСТ Р 53195.3 (таблицы Б.5 и Б.6).

Цель: исключение систематических ошибок, возникающих в результате неблагоприятных сочетаний условий окружающей среды и допусков на компоненты.

Описание: эксплуатационные возможности системы и параметры компонентов исследуются или вычисляются теоретически. При этом для условий окружающей среды задаются их допустимые предельные значения. Анализируются и сопоставляются со спецификацией наиболее существенные характеристики системы.

Более подробное описание данного метода/средства приведено в [97].

Б.6.7 Расширенные функциональные испытания

Примечание — На этот метод/средство дана ссылка в ГОСТ Р 53195.3 (таблицы Б.5 и Б.6).

Цель: обнаружение неисправностей на стадиях составления спецификации, проектирования и разработки системы; проверка поведения СБЗС-системы в случаях ввода редко встречающихся или неспецифицированных видов данных.

Описание: расширенное функциональное тестирование обеспечивает проверку функционального поведения СБЗС-системы как реакцию на входные условия, которые ожидаются только в редких случаях (например, глобальный отказ) или которые не охватываются спецификацией СБЗС-системы (например, некорректные операции). Для редко встречающихся условий наблюдаемое поведение СБЗС-системы сравнивается со спецификацией. В тех случаях, когда реакция СБЗС-системы не специфицирована, следует убедиться в том, что заданная безопасность сохранена в наблюдаемой реакции системы.

Более подробное описание данного метода/средства приведено в [98].

Б.6.8 Испытание в наихудших случаях

Примечание — На этот метод/средство дана ссылка в ГОСТ Р 53195.3 (таблицы Б.5 и Б.6).

Цель: тестирование ситуаций, специфицированных во время анализа наихудших случаев.

Описание: эксплуатационные возможности СБЗС-системы и параметры компонентов тестируются в условиях наихудших случаев. При этом для условий окружающей среды задаются их допустимые предельные значения. Анализируются и сопоставляются со спецификацией наиболее существенные характеристики системы.

Б.6.9 Испытание с введением неисправностей

Примечание — На этот метод/средство дана ссылка в ГОСТ Р 53195.3 (таблицы Б.5 и Б.6).

Цель: внесение отказов в АС системы или их имитация и документирование реакции системы.

Описание: в настоящем стандарте представлен качественный метод оценки зависимости поведения системы от внесенных или имитированных неисправностей. Для описания местоположения и типа неисправностей, а также способа их внесения обычно используют детализированные функциональные блоки, принципиальные и структурные схемы. Например, электропитание может не поступать в различные модули; линии электропитания, линии общей шины или адресные линии могут быть разомкнуты или короткозамкнуты; компоненты или их порты могут быть разомкнуты или закорочены; реле могут быть замкнуты или разомкнуты, либо их действия могут выполняться в неправильные моменты времени и т. д. Возникающие в результате отказы системы классифицируют, например, как в МЭК 60812 (таблицы I и II), см. [99]. Обычно вводят одиночные неисправности в устойчивом состоянии системы. Однако в случае, когда тестом встроенной диагностики неисправность не обнаруживается или оказывается неочевидной, она может сохраниться в системе и вызвать следующую неисправность. При этом число неисправностей может быстро возрасти до сотен.

Эта работа проводится многопрофильным коллективом специалистов и в присутствии представителя поставщика системы, который должен давать необходимые консультации. Для тех отказов, которые приводят к серьезным последствиям, должно вычисляться и оцениваться среднее время наработки на отказ. Если это время мало́, необходима модификация системы.

Более подробное описание данного метода/средства приведено в [90, 95].

Приложение В
(справочное)

Методы/средства для достижения полноты безопасности программного обеспечения (см. ГОСТ Р 53195.4)

В.1 Общие положения

Методы/средства, содержащиеся в этом приложении, не следует рассматривать как полные или исчерпывающие. СБЗС-системы, средства программирования непрерывно развиваются, и методы/средства, предназначенные для достижения полноты безопасности программного обеспечения, непрерывно совершенствуются. В первую очередь следует ориентироваться на стандартизованные методы/средства. В случае применения новых методов/средств следует сформулировать и сохранить все доказательственные материалы, демонстрирующие преимущество новых методов, средств перед методами/средствами, описанными в настоящем приложении.

Более подробное описание некоторых методов/средств приведено в [99—103].

В.2 Требования и детальное проектирование

Примечание — Соответствующие методы/средства приведены в Б.2 настоящего стандарта.

В.2.1 Структурные методы

Примечание — На этот метод/средство дана ссылка в ГОСТ Р 53195.4 (таблицы А.2 и А.4).

В.2.1.1 Общие положения

Цель: обеспечение необходимого качества разработки ПО. Основное внимание уделено ранним стадиям жизненного цикла создаваемой системы. В структурных методах используются как точные, так и интуитивные процедуры и нотации (поддерживаемые компьютерами) для задания и документирования требований и обеспечения реализации программы в логической последовательности структурированным способом.

Описание: существует ряд структурных методов. Некоторые из них спроектированы для выполнения традиционных функций обработки данных и групповых операций, другие (MASCOT, JSD, Yourdon в режиме реального времени) — в большей степени ориентированы на процессы управления и задачи реального времени (которые более критичны с точки зрения безопасности).

Структурные методы — это, в основном, «интеллектуальные инструменты», предназначенные для обобщенного восприятия и декомпозиции задачи или системы. К их основным свойствам относятся:

- логичность рассуждений и выводов, декомпозиция сложной задачи на управляемые стадии;
- анализ и документирование общей системы, включая окружающую среду, а также разрабатываемую систему;
- декомпозиция данных и функций в разрабатываемой системе;
- контрольные таблицы, то есть списки типов объектов, нуждающихся в анализе;
- малая интеллектуальная перегрузка — просто, интуитивно, практично.

Нотации, используемые для анализа и документирования задач и объектов системы (например, на основе процессов и потоков данных), ориентированы на точность, однако нотации для выражения функций обработки, выполняемых этими объектами, оказываются более неформальными. В то же время в некоторых методах частично используют (математически) формальные нотации (например, в JSD используют регулярные выражения, в Yourdon, SOM и SDL используют теорию конечных автоматов). Увеличение точности не только повышает уровень понимания, но и обеспечивает возможность автоматизированной обработки.

Другим преимуществом структурных нотаций является их наглядность, которая позволяет пользователю интуитивно проверять возможности спецификации или проекта при неполной информации.

Настоящий раздел содержит подробное описание пяти структурных методов: «Представление требований», «Разработка системы по Джексону», MASCOT, «Yourdon для систем реального времени» и «Методология структурного анализа и проектирования (SADT)».

Более подробное описание данного метода/средства приведено в [104—106].

В.2.1.2 CORE — контролируемое представление требований

Цель: обеспечение представления и формулирования всех требований.

Описание: этот подход используют для улучшения взаимопонимания между потребителем/конечным пользователем и аналитиком. Он не основан на математически строгой теории, а является средством коммуникации. Метод CORE создан для представления требований, а не спецификаций. Этот подход структурирован, все его представления проходят через различные уровни уточнений. Метод CORE используют для широкого круга задач. Он учитывает сведения об окружающей среде, в которой функционирует система, а также различные точки зрения разных категорий пользователей. Метод CORE содержит руководящие материалы и тактические методы для ухода от «грандиозного проекта». Аргументы такого ухода могут быть скорректированы либо явным образом идентифицированы и задокументированы. Таким образом, спецификации могут быть неполными, однако выявленные нерешенные задачи и области высокого риска должны быть рассмотрены при последующем проектировании.

Подробное описание метода/средства приведено в [105—110].

В.2.1.3 Разработка системы по Джексону — JSD

Цель: разработка программной системы специально для реального времени, охватывающая стадии от разработки требований до кодирования.

Описание: метод JSD (Jackson Structured Development), разработанный Майклом Джексоном в середине 80-х годов, предлагает стиль разработки программных систем, отличный от стиля, принятого в методах SA/SD или OMT. В методе JSD не делается различий между этапом анализа требований к системе и этапом ее разработки. Оба этапа объединяются в один общий этап разработки спецификаций проектируемой системы. При этом этапе решается вопрос «Что должно быть сделано?». Вопрос «Как это должно быть сделано?» решается на следующем этапе — этапе реализации системы. Метод JSD часто применяют для проектирования систем реального времени. В нем использована система графических обозначений, хотя сам метод менее ориентирован на графику, чем методы SA/SD и OMT.

Разработка модели JSD начинается с изучения объектов реального мира. Целью системы является обеспечение требуемой функциональности, но сначала следует убедиться, что эта функциональность согласуется с реальным миром. Модель JSD описывает реальный мир в терминах сущностей (объектов), действий и порядка выполнения действий. Разработка системы по методу JSD включает в себя следующие шесть этапов:

- разработку действий и объектов;
- разработку структуры объектов;
- разработку исходной модели;
- разработку функций;
- разработку временны́х ограничений;
- реализацию системы.

На этапе разработки действий и объектов разработчик, руководствуясь внешними требованиями к проектируемой системе, составляет перечень сущностей (объектов) и действий реального мира, связанных с этой системой. Так, например, при проектировании системы управления двумя лифтами в шестиэтажном доме можно выделить два объекта: «лифт» и «кнопка» — и три действия: «нажатие кнопки», «лифт приходит на этаж *n*» и «лифт покидает этаж *n*». И объекты, и действия соответствуют реальной ситуации. Все действия являются атомарными (неразложимыми на поддействия) и происходят в фиксированные моменты времени.

На этапе разработки структуры объектов действия каждого объекта частично упорядочиваются во времени. Так, в рассматриваемом примере действия «лифт приходит на этаж *n*» и «лифт покидает этаж *n*» должны чередоваться: два действия «лифт приходит на этаж *n*» не могут идти одно за другим.

Этап разработки исходной модели связывает реальный мир с абстрактной моделью, устанавливая соответствие между вектором состояния и потоком данных. Вектор состояния обеспечивает «развязку» по управлению; так, в примере с лифтами первая же нажатая кнопка вверх установит значение переключателя (флажка), «вверх», после чего лифт не будет реагировать на дальнейшие нажатия кнопок вверх, так что нажатие кнопки вверх один или пять раз приведет к одинаковому результату. Аналогично поток данных позволяет обеспечить «развязку» по данным. Примером может служить буфер файла.

На этапе разработки функций с помощью специального псевдокода устанавливаются выходные данные каждого действия. Для системы управления лифтами примером функции является переключение лампочек на панели лифта при прибытии лифта на очередной этаж.

На этапе разработки временны́х ограничений решается вопрос о допустимых временны́х отклонениях системы от реального мира. В результате получается множество временны́х ограничений. В примере с лифтами одним из временны́х ограничений будет решение вопроса о том, как долго нужно нажимать на кнопку лифта, чтобы добиться его реакции.

Наконец, на этапе реализации системы решаются проблемы управления процессами и распределения процессов по процессорам.

Метод JSD может быть лишь условно назван объектно-ориентированным: в нем почти не рассматривается структура объектов, недостаточно внимания уделено их атрибутам. Некоторые действия метода JSD являются, по существу, зависимостями между объектами по методологии OMT.

Тем не менее метод JSD может успешно применяться для проектирования и реализации следующих типов прикладных программных систем:

- параллельные асинхронные программные системы, в которых процессы могут взаимно синхронизировать друг друга;
- программные системы реального времени; метод JSD ориентирован именно на такие системы;
- программные системы для параллельных компьютеров; парадигма, принятая в методе JSD, может оказаться полезной для этого случая.

Метод JSD плохо приспособлен для решения следующих задач:

- высокоуровневый анализ, так как метод JSD не обеспечивает широкого понимания задачи; он не эффективен для абстракции и упрощения задач;
- разработка баз данных, так как это слишком сложная задача для метода JSD.

Подробное описание метода/средства приведено в [111—113].

В.2.1.4 Модульный метод построения, эксплуатации и тестирования программных средств MASCOT

Цель: обеспечение проектирования и реализации систем реального времени.

Описание: MASCOT представляет собой программно поддерживаемый метод проектирования. Он позволяет описывать структуры систем реального времени способом, не зависящим от типа аппаратных средств или языка реализации. При его применении высокоорганизованно реализуется проектирование, приводящее к стро-

го модульной структуре, и обеспечивается близкое соответствие между функциональными элементами проекта и элементами АС, появляющимися при интеграции системы. Система представляется в виде сети конкурирующих процессов, которые взаимодействуют через каналы. Каналами могут быть совокупности файлов или очереди (конвейеры данных). Управление доступом к каналу описывается независимо от процессов в терминах механизмов доступа, которые активизируют также правила планирования процессов. Последняя версия MASCOT была спроектирована с учетом реализации ADA.

MASCOT обеспечивает стратегию приемлемости, основанную на тестировании и верификации как отдельных программных модулей, так и более крупных совокупностей функционально взаимосвязанных программных модулей. Реализация MASCOT ориентирована на ядро MASCOT — набор примитивов планирования, которые лежат в основе реализации и обеспечивают механизмы доступа.

Более подробное описание данного метода/средства приведено в [114].

В.2.1.5 Метод Йордона (Yourdon) для систем реального времени

Цель: обеспечение разработки спецификации и проектирования систем реального времени.

Описание: схема разработки системы, лежащая в основе этого метода, включает в себя три стадии. На первой стадии происходит создание «сущностной модели», которая описывает поведение системы. На второй стадии строится модель реализации, описывающая структуру и механизмы, которые, будучи реализованными, отражают требуемое поведение. На третьей стадии происходит фактическое построение АС и ПО системы. Три стадии строго соответствуют трем традиционным стадиям — спецификации, проектированию и разработке, но основное внимание уделено тому, чтобы разработчик на каждой стадии активно занимался моделированием.

Сущностная модель состоит из двух частей:

- модель окружающей среды, содержащая описание границ между системой и ее окружением, а также описание внешних событий, на которые должна реагировать система;

- модель поведения, которая содержит схемы, описывающие преобразования, выполняемые системой в ответ на события, и описание данных, которые система должна содержать для выдачи ответов.

Модель реализации также подразделяется на две подмодели, описывающие распределение отдельных процессов в процессорах и декомпозицию процессов на программные модули.

Для создания этих моделей рассматриваемый метод сочетает в себе множество хорошо известных методов и средств, в том числе построение диаграмм потоков данных, преобразование графов, структурированный английский язык, диаграммы переходов состояний и сети Петри. Кроме того, этот метод предоставляет средства моделирования предложенного проекта системы для описания на бумаге или автоматического построения из составленных моделей.

Более подробное описание данного метода/средства приведено в [106].

В.2.1.6 Методология структурного анализа и проектирования — SADT

Цель: моделирование и анализ процессов принятия решений и задачи управления в сложных системах на уровне информационных потоков, представленных в виде диаграмм (схем).

Описание: методология SADT представляет собой совокупность методов, правил и процедур для построения функциональной модели объекта какой-либо предметной области. Функциональная модель SADT отображает функциональную структуру объекта, т. е. производимые им действия и связи между этими действиями.

Основные элементы этой методологии основываются на следующих концепциях:

- графическое представление блочного моделирования (графика блоков и линий со стрелками SADT-схемы отображает функцию в виде блока действия, а интерфейсы входа/выхода представляются линиями, соответственно входящими в блок и выходящими из него). Взаимодействие блоков действия друг с другом описывается посредством интерфейсных линий, выражающих «ограничения», которые, в свою очередь, определяют, когда и каким образом функции выполняются и управляются;

- строгость и точность (выполнение правил SADT требует достаточной строгости и точности, не накладывая в то же время чрезмерных ограничений на действия разработчика).

Правила SADT включают в себя:

- ограничение количества блоков на каждом уровне декомпозиции (как правило, 3—6 блоков действия);
- связность диаграмм (номеров блоков);
- уникальность меток и наименований (отсутствие повторяющихся имен);
- синтаксические правила для графики (блоков действия и линий);
- разделение входов и управлений (правило определения роли данных);
- отделение организации от функции (т. е. исключение влияния организационной структуры на функциональную модель).

Методология SADT может быть использована для моделирования широкого круга систем и определения требований и функций, а также для разработки системы, которая удовлетворяет этим требованиям и реализует эти функции. Для уже существующих программных систем методология SADT может быть использована для анализа функций, выполняемых системой, а также для указания механизмов, посредством которых они осуществляются.

Результатом применения методологии SADT является модель, которая состоит из диаграмм, фрагментов текстов и глоссария, имеющих ссылки друг на друга. Диаграммы — главные компоненты модели, все функции ИС и интерфейсы на них представлены как блоки действия и линии (рисунок В.1). Место соединения линии с блоком действия определяет тип интерфейса. Линии обозначают следующее:

- «Вход»: указывается линией со стрелкой, входящей в блок действия слева. Входы могут быть представле-

ны материальными или нематериальными объектами, и они предназначены для обработки одним или несколькими блоками действий;

- «Управление»: обычно это инструкция, процедура, критерий выбора и т. п. Управление реализует выполнение действий и изображается линиями со стрелками сверху блока действия;
- «Механизм»: ресурс (в виде персонала, организационного подразделения, компьютера, автоматизированной системы или описания), необходимый действию для выполнения своих задач;
- «Выход»: все, что вырабатывается в результате действия, изображается линией со стрелкой с правой стороны блока действия.

Одной из наиболее важных особенностей методологии SADT является постепенное введение все бóльших уровней детализации по мере создания диаграмм, отображающих модель.

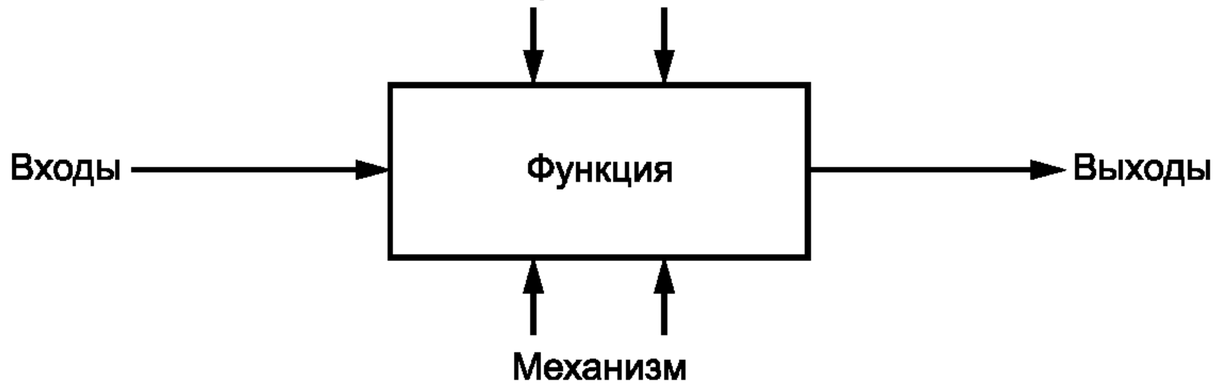


Рисунок В.1 — Функциональный блок действия и интерфейсные линии

На рисунке В.2, где приведены четыре диаграммы и их взаимосвязи, показана структура SADT-модели. Каждый компонент модели может быть декомпозирован на другой диаграмме. Каждая диаграмма иллюстрирует «внутреннее строение» блока действия на «родительской» диаграмме.

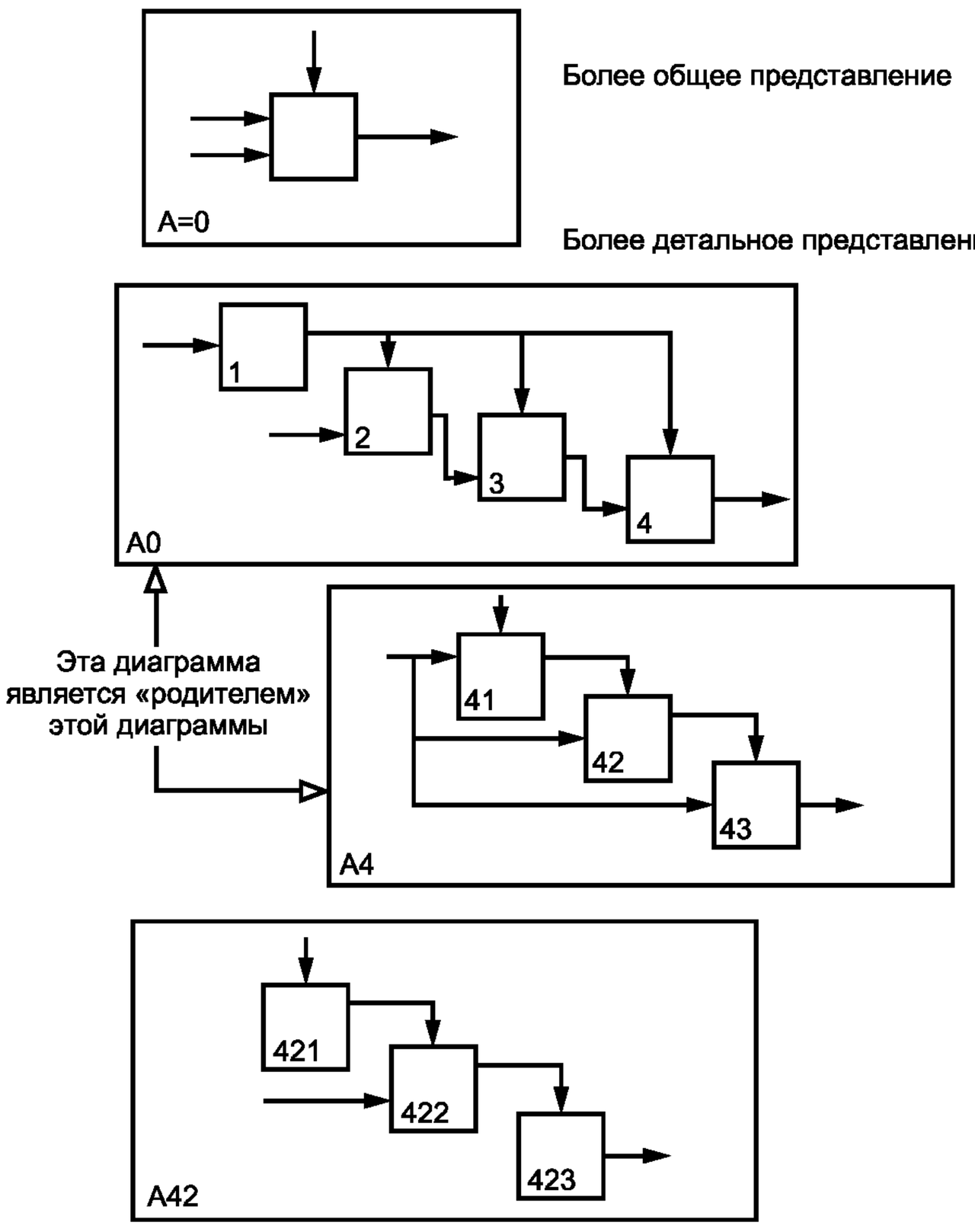


Рисунок В.2 — Структура SADT-модели: декомпозиция диаграмм

На рисунках В.3 — В.5 представлены различные варианты выполнения функций и соединения линий с блоками действия.

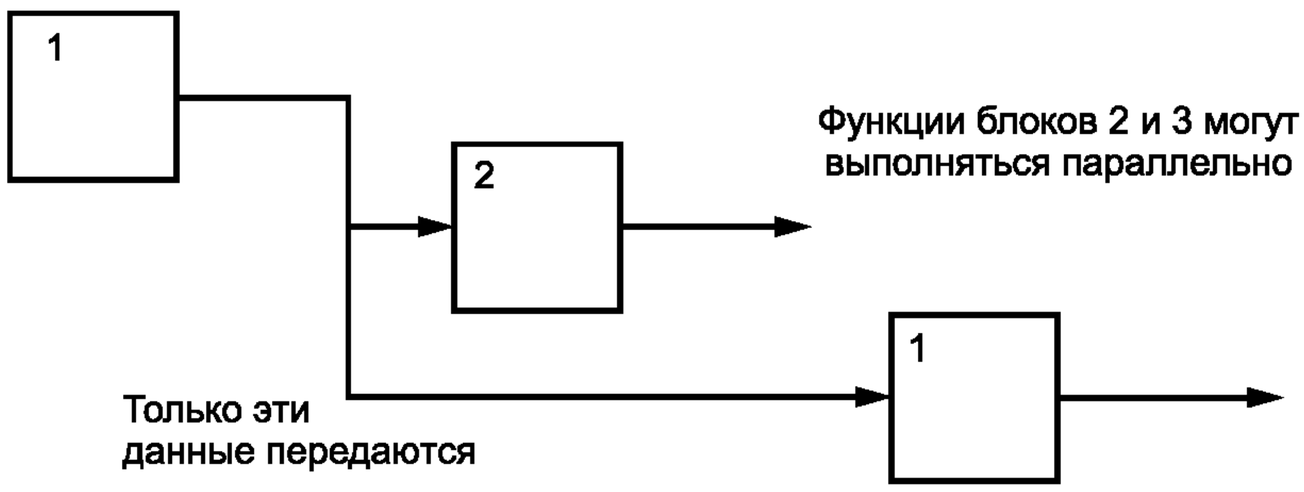


Рисунок В.3 — Одновременное выполнение

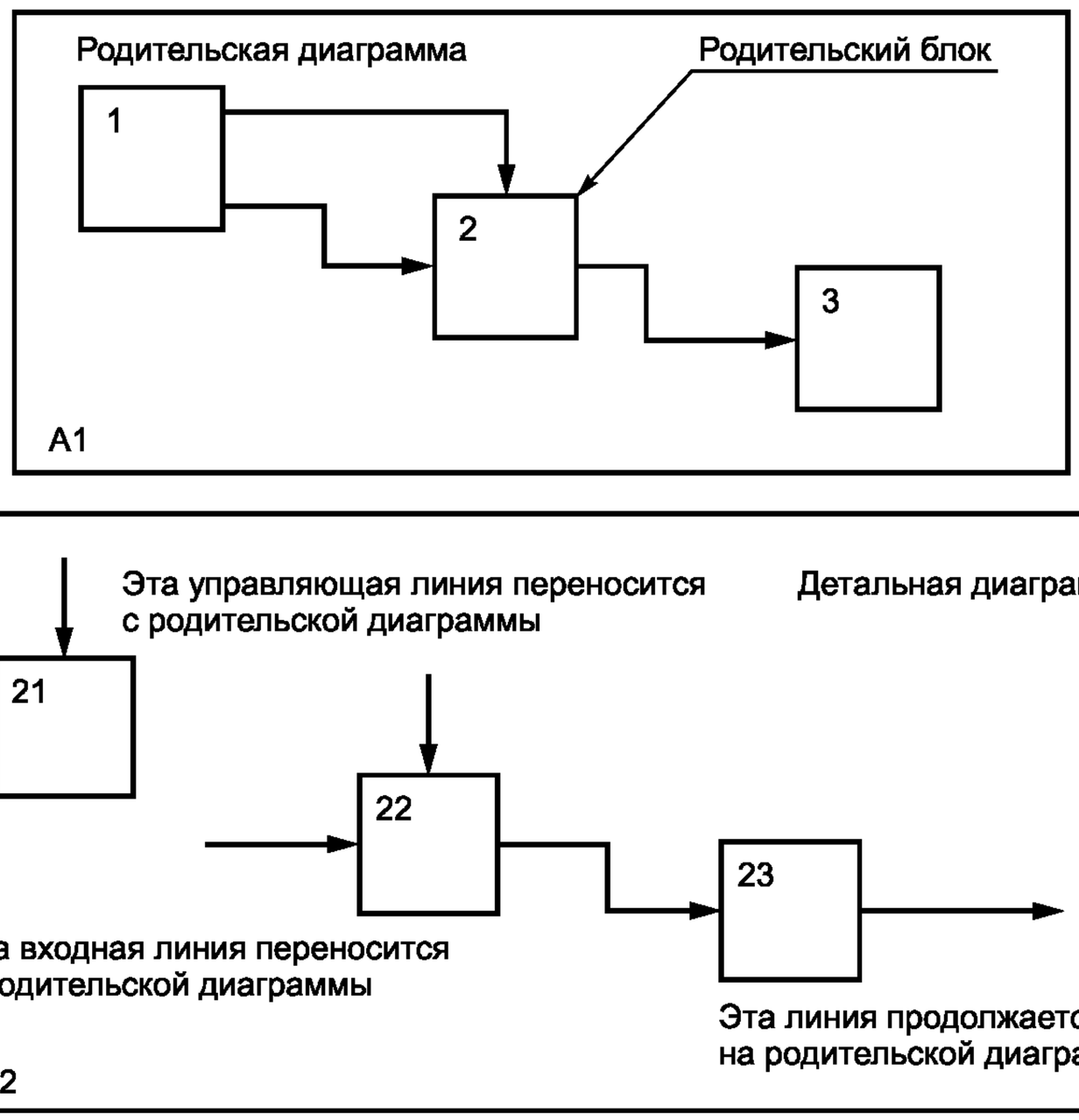


Рисунок В.4 — Взаимосвязь диаграммы А12 с родительской диаграммой А1

Некоторые линии присоединены к блокам действия диаграммы обоими концами, у других же один конец остается неприсоединенным. Неприсоединенные линии соответствуют входам, управлениям и выходам родительского блока. Источник или получатель этих пограничных линий может быть обнаружен только на родительской диаграмме. Неприсоединенные концы должны соответствовать линиям на исходной диаграмме. Все пограничные линии должны продолжаться на родительской диаграмме, чтобы она была полной и непротиворечивой.

На SADT-диаграммах явно не указаны ни последовательность, ни время. Обратные связи, итерации, продолжающиеся процессы и перекрывающиеся (по времени) функции могут быть изображены с помощью линий. Обратные связи могут выступать в качестве комментариев, замечаний, исправлений и т. д. (рисунок В.5).

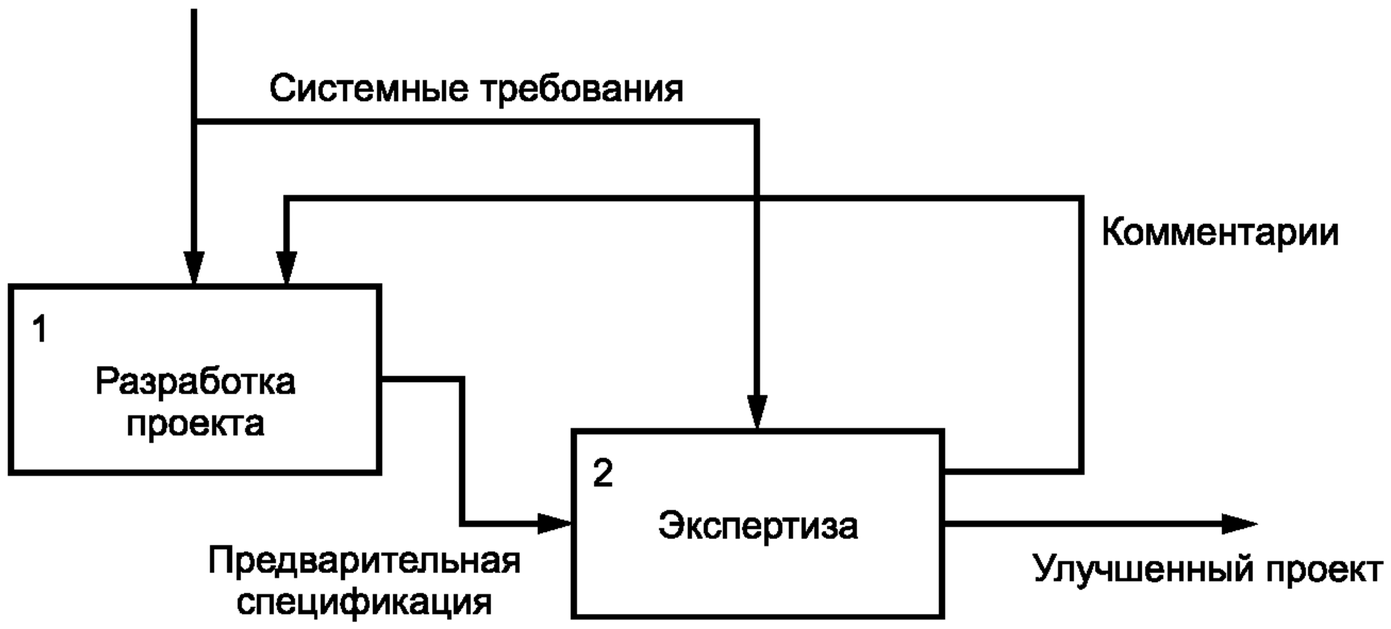


Рисунок В.5 — Пример обратной связи

Механизмы (линии с нижней стороны) показывают средства, с помощью которых осуществляется выполнение функций. Механизмом может быть человек, подразделение, компьютер или любое другое устройство, которое помогает выполнять данную функцию (рисунок В.6).

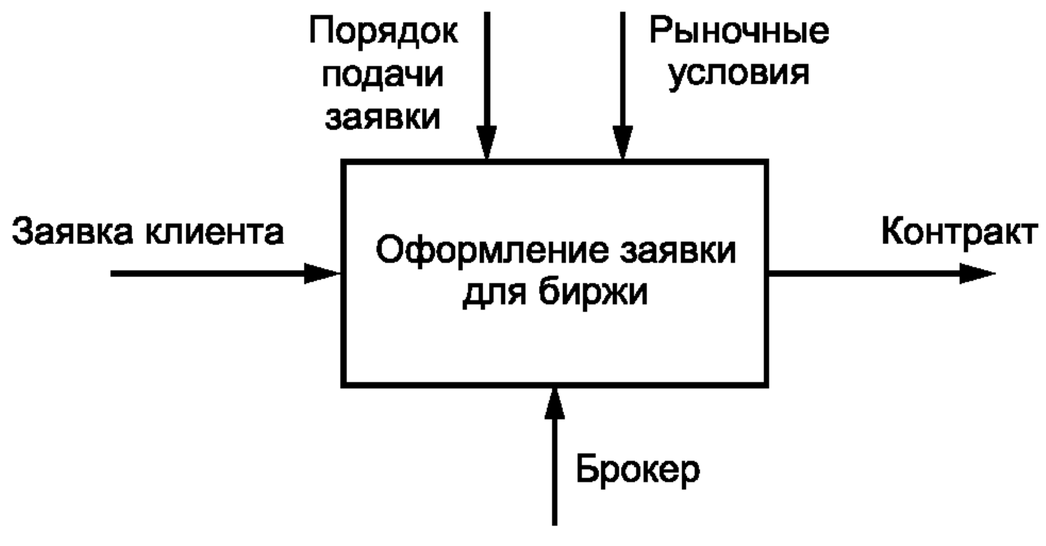


Рисунок В.6 — Пример диаграммы (механизмом является брокер)

Каждый блок действия на диаграмме имеет свой номер. Блок действия любой диаграммы может быть далее описан диаграммой нижнего уровня, которая, в свою очередь, может быть далее детализирована с помощью необходимого числа диаграмм. Таким образом, формируется иерархия диаграмм.

Для того чтобы указать положение любой диаграммы или блока действия в иерархии, используются номера диаграмм. Например, А21 является диаграммой, которая детализирует блок действия 1 на диаграмме А2. Аналогично А2 детализирует блок действия 2 на диаграмме А0, которая является самой верхней диаграммой модели. На рисунке В.7 показано типичное дерево диаграмм.

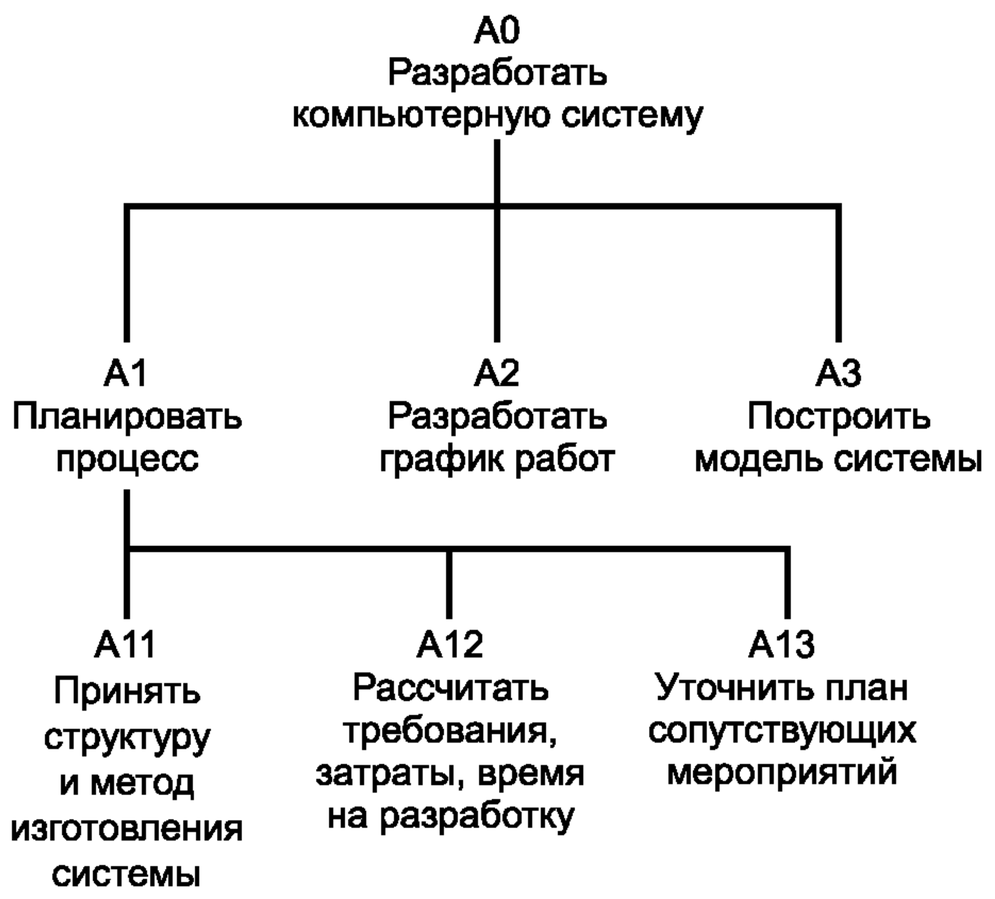


Рисунок В.7 — Иерархия диаграмм

Одним из важных моментов при проектировании ИС с помощью методологии SADT является точная согласованность типов связанности между функциями. Различают по крайней мере семь типов связанности, имеющих различные значимости (таблица В.1).

Т а б л и ц а В.1 — Типы связанности

Тип связанности	Уровень значимости
Случайная	0
Логическая	1
Временна́я	2
Процедурная	3
Коммуникационная	4
Последовательная	5
Функциональная	6

Каждый тип связанности кратко определен и проиллюстрирован ниже с помощью типичного примера из SADT.

Случайная связанность (тип 0) — наименее желательная.

Случайная связанность возникает, когда конкретная связь между функциями мала или полностью отсутствует. Это относится к ситуации, при которой имена данных на SADT-линиях в одной диаграмме имеют малую связанность друг с другом. Крайний вариант этого случая показан на рисунке В.8.

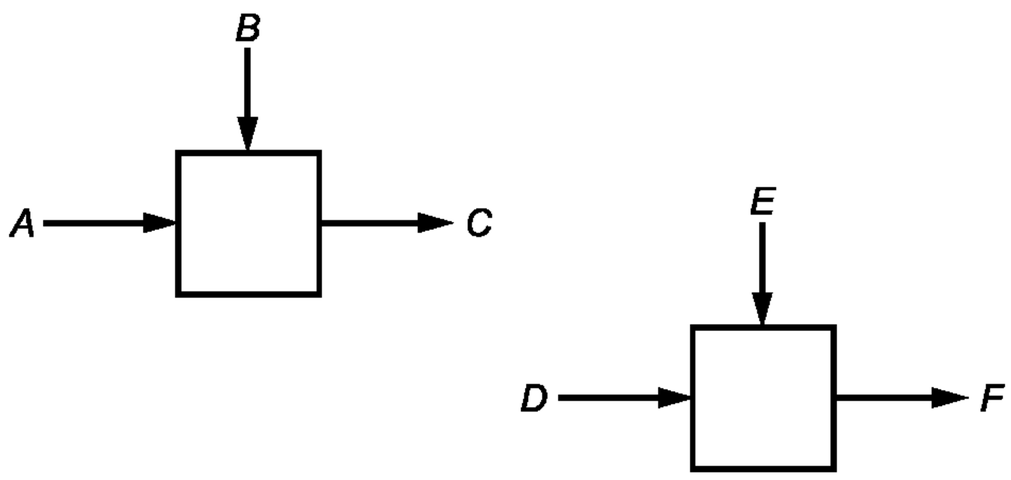


Рисунок В.8 — Случайная связанность

Логическая связанность (тип 1) образуется тогда, когда данные и функции собираются вместе из-за того, что они попадают в общий класс или набор элементов, но необходимые функциональные отношения между ними не обнаруживаются.

Временна́я связанность (тип 2) возникает вследствие того, что связанные по времени элементы представляют функции, связанные во времени, когда данные используются одновременно или функции включаются параллельно, а не последовательно.

Процедурная связанность (тип 3) — процедурно-связанные элементы появляются сгруппированными вместе вследствие того, что они выполняются в течение одной и той же части цикла или процесса. Пример процедурно-связанной диаграммы приведен на рисунке В.9.

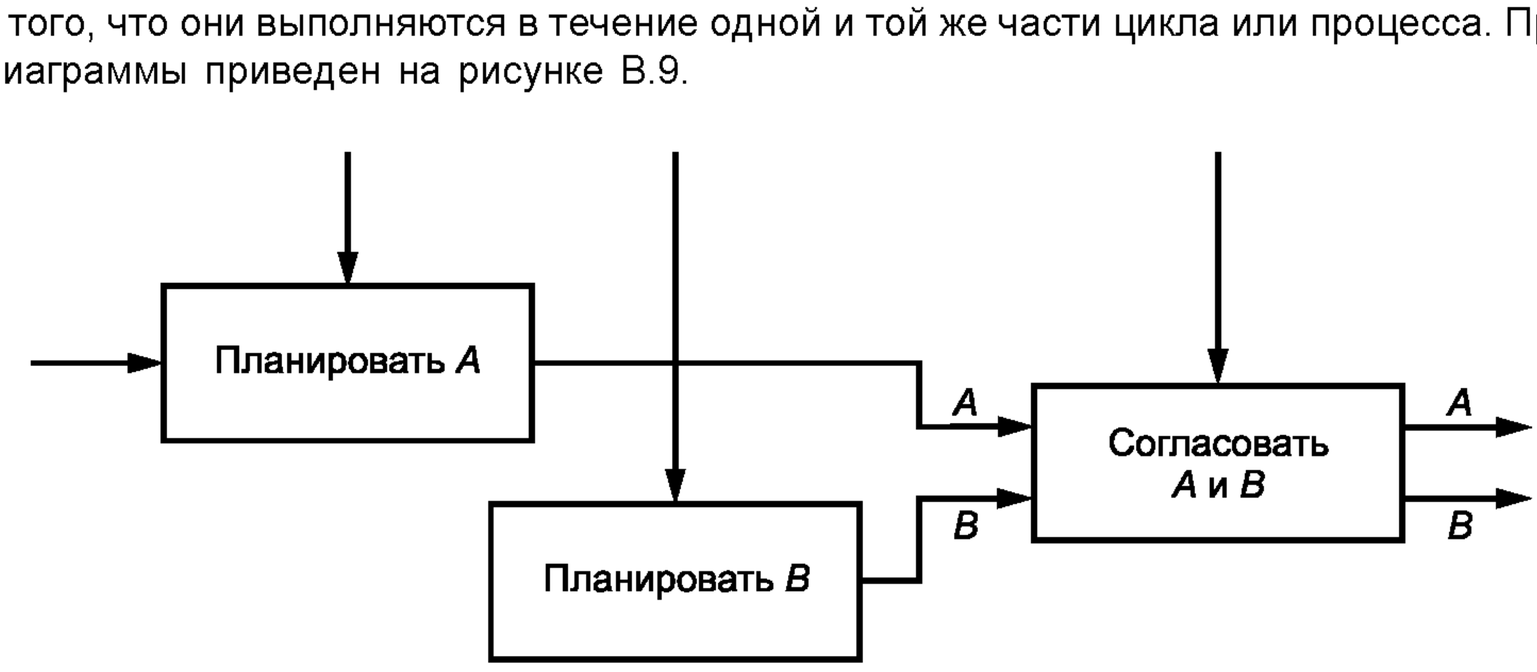


Рисунок В.9 — Процедурная связанность

Коммуникационная связанность (тип 4) — блоки группируются вследствие того, что они используют одни и те же входные данные и/или производят одни и те же выходные данные (рисунок В.10).

Последовательная связанность (тип 5) — выход одной функции служит входными данными для следующей функции. Связанность между элементами на диаграмме является более тесной, чем для рассмотренных выше типовых связанностей, поскольку моделируются причинно-следственные зависимости (рисунок В.11).

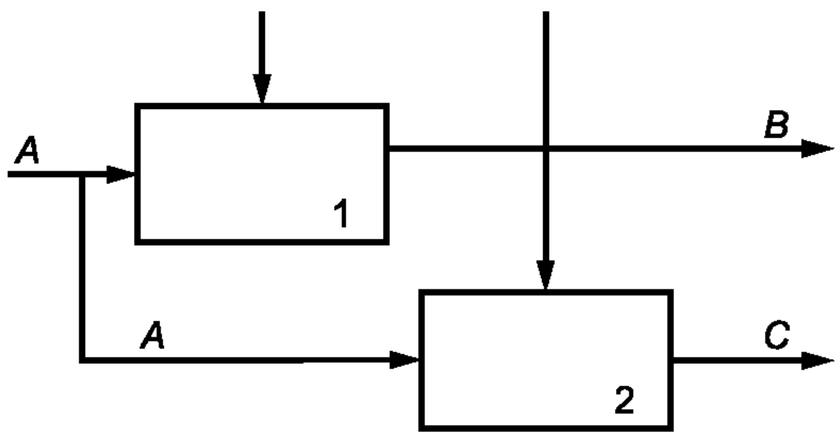


Рисунок В.10 — Коммуникационная связанность

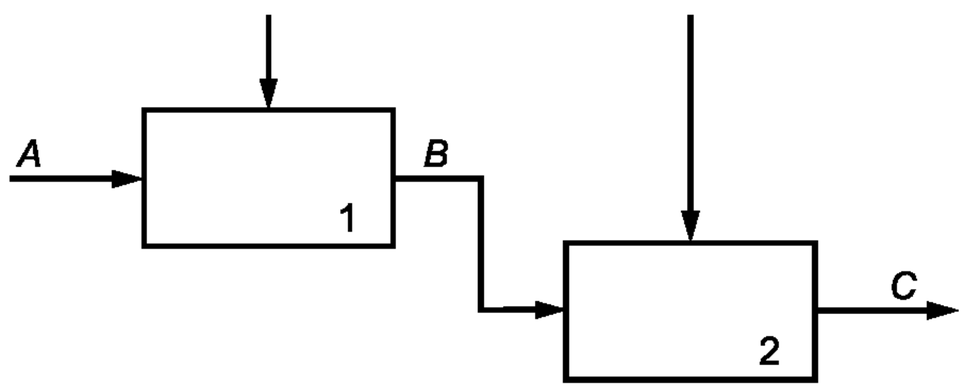


Рисунок В.11 — Последовательная связанность

Функциональная связанность (тип 6) — при наличии полной зависимости одной функции от другой. Диаграмма, которая является чисто функциональной, не содержит чужеродных элементов, относящихся к последовательному или более слабому типу связанности. Одним из способов определения функционально-связанных диаграмм является рассмотрение двух блоков, связанных через управляющие линии, как показано на рисунке В.12.

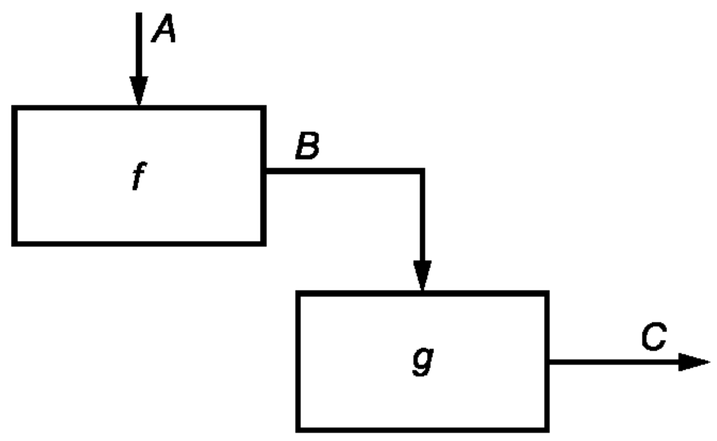


Рисунок В.12 — Функциональная связанность

В математических терминах необходимое условие для простейшего типа функциональной связанности, показанной на рисунке В.12, имеет следующий вид:

$$C = g(B) = g(f(A)),$$

где g — функция, реализуемая блоком действия A;

f — функция, реализуемая блоком действия B.

В таблице В.2 представлены все типы связанностей, рассмотренные выше. Важно отметить, что уровни 4—6 устанавливают типы связанностей, которые разработчики считают наиболее важными для получения диаграмм хорошего качества.

Таблица В.2 — Характеристики связанностей

Уровень значимости	Тип связанности	Для функций	Для данных
0	Случайная	Случайная	Случайная
1	Логическая	Функции одного и того же множества или типа (например, «редактировать все входы»)	Данные одного и того же множества или типа
2	Временна́я	Функции одного и того же периода времени (например, «операции инициализации»)	Данные, используемые в каком-либо временно́м интервале

Окончание таблицы В.2

Уровень значимости	Тип связанности	Для функций	Для данных
3	Процедурная	Функции, работающие в одной и той же фазе или итерации (например, «первый проход компилятора»)	Данные, используемые во время одной и той же фазы или итерации
3	Процедурная	Функции, использующие одни и те же данные	Данные, на которые воздействует одна и та же деятельность
4	Коммуникационная	Функции, выполняющие последовательные преобразования одних и тех же данных	Данные, преобразуемые последовательными функциями
5	Последовательная	Функции, объединяемые для выполнения одной функции	Данные, связанные с одной функцией
6	Функциональная	Внутренняя функция является аргументом внешней функции	Данные для внешней функции связаны с внутренней функцией

Когда действия сильно связаны между собой многими отношениями, то целесообразно объединить эти действия в единую группу, поместить в один блок действия, не детализируя в дальнейшем его содержание. Основополагающий принцип группирования действий в блоки действия состоит в том, что образуемые в результате блоки действия должны соединяться между собой только небольшим числом отношений.

Декомпозиция моделей диаграмм реализуется до тех пор, пока не потеряет смысл дальнейшая детализация блоков действия. Этот процесс завершается, когда действия внутри блоков действия становятся неразделимыми или когда последующая детализация действий внутри блоков действия выходит за область анализа системы.

Более подробное описание данного метода/средства приведено в [115, 116].

В.2.2 Диаграммы потоков данных

Примечание — На этот метод/средство дана ссылка в ГОСТ Р 53195.4 (таблицы Б.5 и Б.7).

Цель: программная поддержка описания потока данных в форме диаграмм.

Описание: диаграммы потоков данных описывают преобразование входных данных в выходные для каждого компонента диаграммы, представляющего различные преобразования.

Диаграммы потоков данных состоят из трех компонентов:

- именованные стрелки — представляют поток данных, входящих и исходящих из блоков преобразования, с кратким описанием этих данных;
- именованные кружки (эллипсы) — представляют блоки преобразования с кратким описанием преобразований;
- операторы («and», «xor») — используются для связи именованных стрелок.

Каждый кружок на диаграмме потока данных может рассматриваться как самостоятельный блок, который при появлении на его входах данных преобразует их в выходные. Одним из основных преимуществ диаграмм является то, что они показывают преобразования, не устанавливая, как они реализуются. Чистая диаграмма потоков данных не включает в себя управляющую информацию или информацию о последовательности процесса, ибо это реализуется в расширениях для реального времени, как в методе Yourdon для систем реального времени (см. В.2.1.5).

Создание диаграмм потока данных является наилучшим подходом при анализе систем от входов к выходам. Каждый кружок на диаграмме должен представлять разное преобразование — его выходы должны отличаться от его входов. Не существует правил определения общей структуры диаграммы, и создание диаграммы потока данных является одним из творческих аспектов создания проекта системы. Подобно всем проектам, это итеративная процедура, уточняющая начальную диаграмму для создания конечной диаграммы.

Более подробное описание данного метода/средства приведено в [117, 118].

В.2.3 Структурные диаграммы

Примечание — На этот метод/средство дана ссылка в ГОСТ Р 53195.4 (таблица Б.5).

Цель: представление структуры программы в виде диаграммы.

Описание: структурные диаграммы дополняют диаграммы потоков данных. Они описывают программируемую систему и иерархию ее компонентов, а также отображают их графически в виде дерева. Они описывают способ реализации элементов диаграммы потоков данных в виде иерархии программных модулей.

Структурная схема показывает взаимоотношения между программными модулями, не указывая при этом порядок активизации этих модулей. Структурные диаграммы изображаются с использованием следующих четырех символов:

- прямоугольник с именем модуля;
- линия, соединяющая эти прямоугольники, формирующие структуру;
- стрелка, отмеченная кругом (без штриховки), с именем данных, передаваемых в направлении элементов структурной схемы и обратно (обычно такая стрелка изображается параллельно с линиями, соединяющими прямоугольники схемы);
- стрелка, отмеченная кругом (заштрихованным), с именем сигнала управления, проходящего в структурной схеме от одного модуля к другому, и эта стрелка также изображается параллельно линии, соединяющей два модуля.

Из любой нетривиальной диаграммы потока данных можно создать множество различных структурных схем.

Диаграммы потоков данных отображают взаимоотношение между информацией и функциями системы. Структурные схемы отображают способ реализации элементов системы. Оба метода представляют две действующие, хотя и различные, точки зрения на систему.

Более подробное описание данного метода/средства приведено в [116].

В.2.4 Формальные методы

Примечание — На этот метод/средство дана ссылка в ГОСТ Р 53195.4 (таблицы А.1, А.2, А.4 и Б.5).

В.2.4.1 Общие положения

Цель: разработка программных средств, основанных на математических принципах. К ним относятся методы формального проектирования и формального кодирования.

Описание: на основе формальных методов разработаны средства описания системы для решения отдельных задач на стадиях спецификации, проектирования или реализации. Создаваемое в результате описание представляет собой строгую нотацию, которая математически анализируется для обнаружения различных видов несогласованностей или некорректностей. Более того, такое описание может быть в некоторых случаях проанализировано автоматически по аналогии с проверкой компилятором синтаксиса исходной программы или использована анимация для представления различных аспектов поведения описываемой системы. Анимация способствует повышению уверенности в том, что система соответствует реальным и формально специфицированным требованиям, поскольку она улучшает восприятие человеком специфицированного поведения системы.

Формальный метод, в основном, предлагает нотацию (в общем случае используется некоторый метод дискретной математики), метод вывода описания в этой нотации и различные виды анализа описания для проверки корректности различных типов.

Примечание — Приведенное выше описание содержится также в Б.2.2.

Ряд формальных методов (CCS, CSP, HOL, LOTOS, OBJ, временна́я логика, VDM и Z) описан в следующих подразделах данного раздела. Другие методы, например метод конечных автоматов (см. Б.2.3.2) и сети Петри (см. Б.2.3.3), могут также рассматриваться как формальные методы в зависимости от корректности использования методов соответствующего строгого математического аппарата.

Более подробное описание данного метода/средства приведено в [119].

В.2.4.2 CCS — средства расчета взаимодействующих систем

Цель: обеспечение описания и анализа поведения систем, реализующих параллельные коммуникационные процессы.

Описание: CCS — это математический аппарат, описывающий поведение систем. Проект системы моделируется в виде сети независимых процессов, реализующихся последовательно или параллельно. Процессы могут взаимодействовать через порты (аналогичные каналам CSP), и коммуникация осуществляется, только когда оба процесса готовы к этому. Отсутствие детерминизма может быть смоделировано. Начиная с описания всей системы на высоком уровне абстрагирования (известного как трассирование), можно выполнять пошаговое уточнение системы в рамках композиции коммуникационных процессов, общее поведение которых формирует и поведение всей системы. В равной степени можно действовать и снизу вверх, комбинируя процессы и получая в результате необходимые свойства формируемой системы, используя правила вывода композиционного типа.

Более подробное описание данного метода/средства приведено в [120].

В.2.4.3 CSP — коммуникационные последовательные процессы

Цель: предоставление способа спецификации конкурирующих программных систем, то есть систем, процессы которых реализуются одновременно.

Описание: CSP предоставляет язык для системных спецификаций процессов и для подтверждения того, что реализация процессов соответствует их спецификациям (описывается как трасса — допустимая последовательность событий).

Система моделируется в виде сети независимых процессов, составленных последовательно или параллельно. Каждый процесс описывается в терминах всех его возможных поведений. Процессы могут взаимодей-

ствовать (действуя синхронно или обмениваясь данными) через каналы, и взаимодействие происходит только при готовности обоих процессов. Может быть промоделирована относительная синхронизация событий.

Теоретические положения метода/средства CSP были непосредственно включены в архитектуру транспьютера INMOS, а язык OCCAM позволил непосредственно реализовывать на сетях транспьютеров системы, специфицированные в CSP.

Более подробное описание данного метода/средства приведено в [121, 122].

B.2.4.4 HOL — логика высокого порядка

Цель: предоставление формального языка, применяемого в качестве основы для спецификации и верификации аппаратных средств.

Описание: HOL представляет собой конкретную логическую нотацию и систему, которая ее автоматически поддерживает. Языки HOL были разработаны в компьютерной лаборатории Кембриджского университета. Логическая нотация взята в основном из простой теории типов Черча, а машинная реализация основана на теории LCF (логике вычислимых функциий).

Более подробное описание данного метода/средства приведено в [123].

B.2.4.5 LOTOS

Цель: обеспечение описания и анализа поведения систем, реализующих параллельные коммуникационные процессы.

Описание: LOTOS (язык для спецификации процессов, упорядоченных во времени) основан на CCS с дополнительными возможностями из близких алгебраических теорий CSP и CIRCAL (теория цепей). Он, преодолевая недостатки языка CCS в управлении структурами данных и в представлении значений выражений, объединяет его со вторым компонентом, основанным на языке абстрактных типов данных ACT ONE. Процесс описания компонентов в LOTOS может быть, однако, использован для других формальных методов при описании абстрактных типов данных.

Более подробное описание данного метода/средства приведено в [124].

B.2.4.6 Язык OBJ

Цель: обеспечение точной спецификации системы при обратной связи с пользователем и подтверждение соответствия системы до ее реализации.

Описание: OBJ представляет собой алгебраический язык спецификаций. Пользователи определяют требования в терминах алгебраических выражений. Системные аспекты — поведение или конструктивы — специфицированы в терминах операций, действующих над абстрактными типами данных (ADT). Язык ADT подобен языку ADA, в котором поведение оператора наблюдаемо, однако подробности реализации скрыты.

Спецификация OBJ и последующая пошаговая реализация подвергаются тем же формальным методам проверки, что и в других формальных методах. Более того, поскольку конструктивные аспекты спецификации OBJ автоматически исполнимы, существует непосредственная возможность подтверждения соответствия системы на основе самой спецификации. Исполнение это, по существу, оценка функций путем подстановки (перезаписи) выражений, которая продолжается до тех пор, пока не будут получены конкретные выходные значения. Это исполнение позволяет конечным пользователям рассматриваемой системы получать «облик» планируемой системы на этапе ее спецификации без необходимости знакомства с методами, лежащими в основе формальных спецификаций.

Как и все другие средства ADT, язык OBJ применим только к последовательным системам или к последовательным аспектам параллельных систем. OBJ используется для спецификации как малых, так и крупных промышленных применений.

Более подробное описание данного метода/средства приведено в [125, 126].

B.2.4.7 Временна́я логика

Цель: непосредственное выражение требований к безопасности и эксплуатации, а также формальное представление сохранения этих качеств на последующих этапах разработки.

Описание: стандартная предикатная логика первого порядка не содержит концепций времени. Временна́я логика расширяет логику первого порядка добавлением модальных операторов (например, «с этого момента» и «случайно»). Эти операторы могут использоваться для уточнения суждений о системе. Например, свойства безопасности могут потребовать использовать оператор «с этого момента», тогда как может потребоваться, чтобы другие необходимые состояния системы были достигнуты «случайно» из некоторого другого начального состояния. Временные формулы интерпретируются последовательностями состояний (поведениями). Что такое «состояние», зависит от выбранного уровня описания. Оно может относиться ко всей системе, системным компонентам или компьютерной программе.

Задаваемые количественно (квонтифицированные) временные интервалы и ограничения во временной логике не обрабатываются явно. Абсолютное время обрабатываются путем образования дополнительных временны́х состояний как части описания состояния.

Более подробное описание данного метода/средства приведено в [127—129].

B.2.4.8 VDM, VDM++ — метод разработки Vienna

Цель: систематическая спецификация и реализация последовательных (VDM) и параллельных (VDM++) программ реального времени.

Описание: VDM — это математический метод спецификации и математический метод уточнения реализаций, который позволяет доказать их корректность относительно спецификации.

В этом основанном на модели методе спецификации состояние системы моделируется в терминах теоретико-множественных структур, в которых описаны инварианты (предикаты), а операции над этим состоянием моделируются путем определения их пред- и постусловий в терминах системных состояний. Операции могут проверяться на сохранение системных инвариантов.

Выполнение спецификаций осуществляется путем реализации состояния системы в терминах структур данных в заданном языке и путем уточнения операций в терминах программы на заданном языке. Этапы реализации и уточнения позволяют логически вывести свойства, которые устанавливают корректность этих этапов. Выполняются или не выполняются эти свойства, определяется разработчиком.

VDM в принципе используется на этапе создания спецификации, но может также использоваться на этапах проектирования и реализации исходного кода. Он может быть также применен к последовательно структурированным программам или к последовательным процессам в параллельных системах.

Объектно-ориентированное и параллельное для реального времени расширения, VDM, VDM++ представляют собой язык формализованных спецификаций, основанный на языке VDM-SL, созданном в ISO, и на объектно-ориентированном языке Smalltalk.

VDM++ имеет широкий диапазон конструкций, с помощью которых пользователь может формально специфицировать параллельные системы реального времени в объектно-ориентированной среде. В VDM++ полная формальная спецификация содержит совокупность спецификаций классов и отдельных характеристик рабочего пространства.

К средствам описания реального времени в языке VDM++ относятся:

- временны́е выражения, предусмотренные для представления как текущего момента, так и момента вызова метода внутри тела метода;

- выражение, описывающее синхронизирующий сигнал, которое может быть добавлено к методу для спецификации верхних (или нижних) пределов времени исполнения для корректности реализаций;

- переменные непрерывного времени, которые должны быть введены. С условными операторами и операторами действия можно специфицировать отношения (например, дифференциальные уравнения) между этими временны́ми функциями. Это свойство оказывается очень полезным при спецификации требований к системам, которые действуют в среде с непрерывным временем. Уточняющие шаги приводят к дискретным программным решениям для систем подобного вида.

Более подробное описание данного метода/средства приведено в [130].

B.2.4.9 Z-метод

Цель: предоставление нотации языка спецификаций для последовательных систем и метода проектирования, применяемого разработчиком на стадиях от составления спецификации на языке Z до разработки исполнительных алгоритмов, позволяющей при этом доказать их корректность по отношению к спецификации. Больше всего он подходит для разработки последовательных систем, ориентированных на данные.

Описание: как и в методе VDM, в этом основанном на модели Z-методе спецификации состояний системы моделируется в терминах теоретико-множественных структур, в которых описаны инварианты (предикаты), а операции над этим состоянием моделируются путем определения их пред- и постусловий в терминах системных состояний. Операции могут проверяться на сохранение системных инвариантов, демонстрируя тем самым их согласованность. Формальная часть спецификации подразделяется на схемы, которые обеспечивают возможность структурирования спецификаций путем их усовершенствования.

Обычно спецификация Z представляет собой сочетание формального Z-текста и неформального пояснительного текста на естественном языке. (Формальный текст сам по себе может оказаться слишком сжатым для простого восприятия, и часто его смысл необходимо пояснять, тогда как неформальный естественный язык может оказаться неоднозначным и неточным.)

В отличие от VDM язык Z представляет собой скорее нотацию, чем завершенный метод. Однако был разработан близкий метод (названный B-методом), который может быть использован в сочетании с языком Z. Метод B основан на принципе пошагового уточнения.

Более подробное описание данного метода/средства приведено в [131].

B.2.5 Программирование с защитой

П р и м е ч а н и е — На этот метод/средство дана ссылка в ГОСТ Р 53195.4 (таблица А.4).

Цель: создание программ, которые во время их исполнения выявляют аномальные потоки сигналов управления, потоки данных или значения данных и реагируют на них заранее определенным и подходящим способом.

Описание: в процессе разработки программ можно использовать много методов для проверки аномалий в сигналах управления или в данных. Эти методы/средства могут применяться систематически в процессе программирования системы с целью уменьшения вероятности ошибочной обработки данных.

Имеются два пересекающихся множества методов защиты. Внутренние методы/средства защиты от ошибок проектируются в программных средствах для преодоления недостатков их проектирования. Эти недостатки могут быть обусловлены ошибками при проектировании или кодировании либо ошибочными требованиями. Ниже перечислены некоторые из методов защиты:

- проверка диапазона значений переменных;

- по возможности проверка значений переменных на их достоверность;
- на входе процедур проверка типа, размерности и диапазона значений параметров процедур.

Эти три метода помогают гарантировать, что значения, которые обрабатываются в программах, допустимы с точки зрения как терминов программных функций, так и физических значений переменных.

Параметры только для считывания и параметры для считывания-записи должны быть разделены, и доступ к ним должен проверяться. Функции должны рассматривать все параметры как параметры только для считывания. Буквенные константы не должны быть доступны для записи. Это помогает обнаруживать случайные перезаписи или ошибочное использование переменных.

Устойчивые к ошибкам программные средства проектируются в «предположении», что ошибки существуют в собственной среде либо при использовании выходящих за номиналы или предполагаемых условий, и при этом ведут себя заранее определенным образом. В таком случае используются следующие методы:

- проверка на достоверность физических значений входных и промежуточных переменных;
- проверка влияния выходных переменных, предпочтительно путем прямого наблюдения соответствующих изменений состояния системы;
- проверка самими программными средствами своей конфигурации, включая наличие и доступность предполагаемых АС, а также завершенность самих программ — это особенно важно для поддержки полноты в процессе их эксплуатации.

Некоторые из методов защиты программ, такие как проверки последовательности потока управления, также справляются с внешними ошибками.

Более подробное описание данного метода/средства приведено в [132—136].

В.2.6 Стандарты по проектированию и кодированию

Примечание — На эти методы/средства дана ссылка в ГОСТ Р 53195.4 (таблица А.4).

В.2.6.1 Общие положения

Цель: упрощение верификации для поддержания группового объективного подхода и установления стандартного метода проектирования.

Описание: в самом начале разработки между участниками создания системы должны быть согласованы необходимые правила. Они охватывают рассмотренные ниже методы проектирования и разработки (например, JSP, MASCOT, сети Петри и т. д.), а также соответствующие стандарты кодирования (см. В.2.6.2 настоящего приложения).

Эти правила создаются для облегчения разработки, верификации, оценки соответствия и эксплуатации. При этом должны учитываться доступные инструментальные средства, в частности для аналитиков, и развитие средств проектирования.

Более подробное описание данного метода/средства приведено в [137].

В.2.6.2 Стандарты кодирования

Примечание — На эти методы/средства дана ссылка в ГОСТ Р 53195.4 (таблица Б.1).

Цель: упрощение верификации разработанного кода.

Описание: до выполнения кодирования должны быть полностью согласованы подробные правила, которых следует придерживаться. К этим правилам обычно относят:

- наличие подробных сведений о модульности, например о виде интерфейса, о размерах программных модулей;
- использование инкапсуляции, наследования (ограниченного по глубине) и полиморфизма в случае объектно-ориентированных языков;
- исключение или ограниченное использование некоторых языковых конструкций, например, «goto», «equivalence», динамических объектов, динамических данных, структур динамических данных, рекурсии, указателей и т. п.;
- ограничение прерываний, допустимых при выполнении критичного для безопасности кода;
- распечатывание программного кода (формирование листинга);
- исключение безусловных переходов (например, «goto») в программах на языках высокого уровня.

Эти правила созданы для облегчения процессов тестирования программных модулей, верификации, оценки соответствия и обслуживания. При этом должны учитываться доступные инструментальные средства, в частности для аналитиков.

Примечание — Более подробная информация по этим вопросам приведена в В.2.6.3 — В.2.6.7.

Более подробное описание данного метода/средства приведено в [102, 137—142].

В.2.6.3 Отказ от динамических переменных или динамических объектов

Примечание — На этот метод/средство дана ссылка в ГОСТ Р 53195.4 (таблица Б.1).

Цель: исключение динамических и переменных объектов во избежание:
- нежелательных или необнаруживаемых наложений в памяти;
- узких мест ресурсов в процессе (связанном с безопасностью) выполнения программы.

Описание: в случае применения этого подхода динамические переменные и динамические объекты оказываются переменными и объектами, которые имеют свои определенные и абсолютные адреса в памяти, устанавливаемые во время выполнения программы. Объем распределяемой памяти и ее адреса зависят от состояния системы в момент распределения памяти, а это означает, что они не могут быть проверены компилятором или любым другим автономным инструментом.

Так как число динамических переменных и объектов и существующее свободное пространство памяти для размещения новых динамических переменных или объектов зависят от состояния системы в момент размещения, то возможны сбои при размещении или при использовании переменных или объектов. Например, если объем свободной памяти для распределяемой переменной системы не достаточен, то содержимое другой переменной в памяти может быть нечаянно стерто. Если динамические переменные или объекты не используются, то появление этих сбоев исключено.

В.2.6.4 Проверка создания динамических переменных или динамических объектов при выполнении программы

Примечание — На этот метод/средство дана ссылка в ГОСТ Р 53195.4 (таблица Б.1).

Цель: убедиться в том, что память, в которой должны быть размещены динамические переменные и объекты, свободна до ее загрузки, гарантируя при этом, что размещение в ней динамических переменных и объектов во время выполнения программы не повлияет на уже существующие в ней переменные, данные или коды.

Описание: в случае применения этих методов/средств к динамическим переменным относят переменные, имеющие свои определенные и абсолютные адреса в памяти, устанавливаемые во время выполнения программы (в этом смысле переменные являются также атрибутами экземпляров объектов).

Аппаратными либо программными средствами память проверяется на то, что она свободна до размещения в ней динамических переменных или объектов (например, для того, чтобы исключить переполнение стека). Если размещение не разрешается (например, если памяти по определенному адресу недостаточно), должны быть предприняты соответствующие действия. После использования динамических переменных или объектов (например, после выхода из подпрограммы) вся используемая ими память должна быть освобождена.

Примечание — Альтернативой служит статическая демонстрация того, что память будет адекватной во всех случаях.

В.2.6.5 Ограниченное использование прерываний

Примечание — На этот метод/средство дана ссылка в ГОСТ Р 53195.4 (таблица Б.1).

Цель: сохранение верифицируемости и тестируемости ПО.

Описание: использование прерываний должно быть ограничено. Прерывания могут использоваться, если они упрощают систему. Использование программных средств для обработки прерываний должно быть запрещено в критических ситуациях для выполняемых функций (например, при критичности по времени, критичности изменения данных). Если прерывания все же используются, то непрерываемые фрагменты должны иметь определенное максимальное время вычисления, на основании которого определяется максимальное время, в течение которого прерывание запрещено. Использование прерываний и их маскирование должно четко документироваться.

В.2.6.6 Ограниченное использование указателей

Примечание — На этот метод/средство дана ссылка в ГОСТ Р 53195.4 (таблица Б.1).

Цель: исключение проблем, связанных с доступом к данным без предварительной проверки типа и диапазона указателя; обеспечение возможности модульного тестирования и верификации программных средств; снижение тяжести последствий отказов.

Описание: в прикладных программных средствах указатель арифметических действий может быть использован на уровне исходного кода только в том случае, когда тип и диапазон значений указателя данных будут проверены перед доступом (для гарантирования того, что ссылка указателя находится внутри корректного адресного пространства). Связи между задачами в прикладных программах не должны осуществляться с помощью непосредственных ссылок между задачами. Обмен данными должен осуществляться через операционную систему.

В.2.6.7 Ограниченное использование рекурсий

Примечание — На этот метод/средство дана ссылка в ГОСТ Р 53195.4 (таблица Б.1).

Цель: исключение использования вызовов неверифицируемых и нетестируемых подпрограмм.

Описание: при использовании рекурсии должен быть установлен четкий критерий, обеспечивающий предсказуемость глубины рекурсии.

В.2.7 Структурное программирование

Примечание — На этот метод/средство дана ссылка в ГОСТ Р 53195.4 (таблица А.4).

Цель: проектирование и реализация программы с использованием практического анализа программы без ее выполнения. Программа может содержать только абсолютный минимум статистически нетестируемого поведения.

Описание: для минимизации структурной сложности следует применять следующие принципы и действия:

- разделять программу на подходящие небольшие программные модули, гарантируя при этом, что они являются минимально связанными, насколько возможно, и что все взаимодействия явные;

- составлять поток управления программными модулями с использованием структурированных конструкций, таких как последовательности, итерации и выбор;

- обеспечивать небольшое количество возможных путей через программные модули и как можно более простые отношения между входными и выходными параметрами;

- исключать сложные ветвления и, в частности, исключать безусловные переходы («goto») при использовании языков высокого уровня;

- по возможности, связывать ограничения на цикл и ветвление с входными параметрами;

- исключать использование сложных вычислений в ветвлении и цикле.

Свойства языков программирования, которые способствуют указанному выше подходу, следует использовать, предпочитая их другим свойствам, которые более эффективны, но за исключением тех случаев, когда эффективность приобретает абсолютный приоритет (например, для некоторых критичных к безопасности систем).

Более подробное описание данного метода/средства приведено в [102, 137, 138, 141, 143–149].

В.2.8 Ограничение доступа / инкапсуляция информации

Примечание — На этот метод/средство дана ссылка в ГОСТ Р 53195.4 (таблица Б.9).

Цель: предотвращение непреднамеренного доступа к данным или процедурам, и тем самым обеспечение качественной структуры программных средств.

Описание: общедоступные для всех программных компонентов данные могут быть случайно или некорректно модифицированы любым из этих компонентов. Любые изменения этих структур данных могут потребовать подробной проверки программного кода и серьезных исправлений.

Ограничение доступа представляет собой общий подход к минимизации указанных проблем. Ключевые структуры данных «скрыты», и с ними можно работать, только применив определенный набор процедур доступа. Это позволяет модифицировать внутренние структуры или добавлять новые процедуры, не оказывая влияния при этом на функциональное поведение остальных программных средств. Например, имя директории может иметь процедуры доступа «вставить», «удалить» и «найти». Процедуры доступа и структуры внутренних данных могут быть изменены (например, при использовании различных методов просмотра или для сохранения имен на жестком диске), не оказывая влияния на логическое поведение остальных программных средств, использующих эти процедуры.

В таком случае следует использовать концепцию абстрактных типов данных. Если непосредственная проверка не предусмотрена, может оказаться необходимой проверка того, что абстрагирование не было непреднамеренно разрушено.

Более подробное описание данного метода/средства приведено в [150, 151].

В.2.9 Модульный подход

Примечание — На этот метод/средство дана ссылка в ГОСТ Р 53195.4 (таблицы А.4 и Б.9).

Цель: декомпозиция программной системы на небольшие ясные для понимания модули для упрощения системы.

Описание: модульный подход, или модуляризация, включает в себя несколько различных правил для стадий проектирования, кодирования и эксплуатации разработанных программных средств. Эти правила меняются в соответствии с реализуемым методом проектирования. Большинство же методов содержат следующие правила:

- программный модуль должен выполнять одну четко сформулированную задачу или функцию;

- связи между программными модулями должны быть ограничены и строго определены, уровень связанности каждого программного модуля должен быть высоким;

- совокупности подпрограмм должны строиться так, чтобы обеспечивать несколько уровней программных модулей;

- размеры подпрограмм следует ограничить некоторыми конкретными значениями, обычно от двух до четырех размеров экрана.

- подпрограммы должны иметь только один вход и один выход;

- программные модули должны взаимодействовать с другими программными модулями через свои интерфейсы, где используются глобальные или общие переменные, которые должны быть хорошо структурированы, доступ к ним должен быть контролируемым и их использование в каждом конкретном случае должно быть обосновано;

- все интерфейсы программных модулей должны быть полностью документированы;

- все интерфейсы программных модулей должны содержать только те параметры, которые необходимы для их функционирования.

Более подробное описание данного метода/средства приведено в [70, 138, 152].

В.2.10 Использование заслуживающих доверия/проверенных программных модулей и их компонентов

Примечания

1 На этот метод/средство дана ссылка в ГОСТ Р 53195.4 (таблица А.4).

2 Некоторые математические методы, обеспечивающие последующие численные оценки, приведены в приложении Г. Аналогичные средства и статистический подход изложены также в Б.5.4.

Цель: исключение вариантов проектирования компонентов программных модулей и АС, предусматривающих необходимость их интенсивных повторных проверок или перепроектирования для каждого нового применения; использование преимуществ проектов, которые не были формально или строго проверены, но для которых имеются продолжительные эксплуатационные предыстории.

Описание: применение таких модулей и компонентов гарантирует, что программные модули и компоненты достаточно свободны от систематических ошибок проектирования и/или эксплуатационных отказов. Использование заслуживающих доверия программных модулей и компонентов (то есть проверенных в эксплуатации) может быть достаточным в качестве единственной меры, гарантирующей достижение необходимого уровня полноты безопасности, лишь в редких случаях. Для сложных компонентов со многими возможными функциями (например, операционной системы) важно установить, какая из функций реально достаточно проверена при ее использовании. Например, в тех случаях, когда используется процедура самотестирования для обнаружения сбоев АС и когда в период эксплуатации никаких отказов АС не случилось, процедуру самотестирования на обнаружение сбоев нельзя рассматривать как проверенную на практике.

Компонент или программный модуль может быть в достаточной мере заслуживающим доверия, если он уже проверен для требуемого уровня полноты безопасности или если он соответствует следующим критериям:

- спецификация не изменялась;
- системы использовались в различных применениях;
- имеется по меньшей мере один год предыстории работы;
- время эксплуатации соответствует уровню полноты безопасности или соответствующему числу запросов; демонстрируются не связанные с безопасностью частоты отказов, меньшие чем:
- 10^{-2} отказов на запрос (в год) с уверенностью 95 % при требуемом числе эксплуатационных прогонов (в год) 30;
- 10^{-5} отказов на запрос (в год) с уверенностью 99,9 % при требуемом числе эксплуатационных прогонов (в год) 690 000;
- весь опыт эксплуатации должен быть соотнесен с известным профилем запросов функций программного модуля для гарантирования того, что увеличивающийся опыт эксплуатации действительно приводит к увеличению сведений о поведении программного модуля, связанного с соответствующим профилем запроса;
- отказы, не связанные с безопасностью.

Примечание — Отказ, который может быть некритичным для безопасности в одном контексте, может оказаться критичным для безопасности в другом контексте, и наоборот.

Чтобы обеспечить достоверность того, что компонент или программный модуль соответствует критерию, должно быть задокументировано следующее:

- точная идентификация каждой системы и ее компонентов, включая номера версий (как для ПО, так и для АС);
- идентификация пользователей и время применения;
- время эксплуатации;
- процедура выбора систем, применяемых пользователями, и случаев применения;
- процедуры обнаружения и регистрации отказов и устранения сбоев.

Более подробное описание данного метода/средства приведено в [148, 150, 153].

В.3 Структурное проектирование

В.3.1 Обнаружение и диагностика сбоев

Примечание — На этот метод/средство дана ссылка в ГОСТ Р 53195.4 (таблица А.2).

Цель: обнаружение сбоев в системе, которые могут привести к отказам, и тем самым обеспечение основы для мер по минимизации последовательностей ошибок.

Описание: обнаружение сбоев представляет собой действие по проверке системы на наличие ошибочных состояний (обусловленных сбоями в проверяемой системе или подсистеме), предпринимаемое для предотвращения появления неверных результатов. Система, действующая в сочетании с параллельными компонентами, останавливающая управление при обнаружении некорректности ее собственных результатов, называется самопроверяемой.

Обнаружение сбоев основывается на принципах избыточности (в основном при обнаружении сбоев АС — см. ГОСТ Р 53195.3, приложение А) и разнообразия (программные сбои). Для определения корректности результатов требуется некоторый вид голосования. Могут быть применены определенные специальные методы, к которым относятся: программирование утверждений, метод программирования N-версий и так называемая

«подушка безопасности» (см. В.3.4 настоящего приложения); а для АС — применение дополнительных сенсоров, контуров регулирования, кодов, обнаруживающих ошибки, и др.

Обнаружение сбоев может обеспечиваться проверками в области значений или во временно́й области на различных уровнях, особенно на физическом уровне (температура, напряжение и т. п.), на логическом (коды, обнаруживающие ошибки), на функциональном (утверждения) или на внешнем (проверки достоверности) уровне. Результаты этих проверок могут быть сохранены и увязаны с влияющими данными для обеспечения возможности отслеживания отказов.

Сложные системы состоят из подсистем. Эффективность обнаружения сбоев, диагностики и компенсации сбоев зависит от сложности взаимодействия между подсистемами, которые влияют на распространение сбоев.

Диагностику сбоев следует применять на уровне самых малых подсистем, поскольку подсистемы меньших размеров допускают более детальную диагностику сбоев (обнаружение ошибочных состояний).

Интегрированные информационные системы (например, уровня предприятия) могут обычным способом передавать состояния безопасности системы, включая информацию диагностического тестирования, другим управляющим системам. При обнаружении некорректного поведения оно может быть выделено и использовано для запуска корректирующих действий до возникновения опасной ситуации. В конце концов, при появлении инцидента документирование таких отклонений может способствовать последующему анализу.

Более подробное описание данного метода/средства приведено в [153].

В.3.2 Обнаружение и исправление ошибок

Примечание — На этот метод/средство дана ссылка в ГОСТ Р 53195.4 (таблица А.2).

Цель: обнаружение и исправление ошибок в чувствительной к ним информации.

Описание: для информации, состоящей из *n* битов, генерируется закодированный блок из *k* битов, который позволяет обнаруживать и исправлять *r* ошибок. Примерами служат код Хэмминга и полиномиальные коды.

Следует заметить, что в системах, связанных с безопасностью, лучше уничтожить ошибочные данные, чем пытаться исправлять их, поскольку лишь заранее определенная часть ошибок может быть правильно исправлена.

Более подробное описание данного метода/средства приведено в [154].

В.3.3 Программирование с проверкой ошибок

Примечание — На этот метод/средство дана ссылка в ГОСТ Р 53195.3 (таблица А.18) и в ГОСТ Р 53195.4 (таблица А.2).

Цель: обнаружение ошибок, оставшихся при проектировании ПО в процессе выполнения программ для предотвращения критичных для безопасности отказов систем и продолжения выполнения программы с высокой надежностью.

Описание: в методе программирования утверждений заложена идея проверки предусловий (до выполнения последовательности операторов начальные условия проверяются на соответствие) и постусловий (проверяются результаты после выполнения последовательности операторов). Если предусловия или постусловия не соблюдаются, то выдается сообщение об ошибке.

Пример

assert < pre-condition>;
action 1;
...
action x;
assert < post-condition>.

Более подробное описание данного метода/средства приведено в [155 — 157].

В.3.4 Методы «подушки безопасности»

Примечание — На этот метод/средство дана ссылка в ГОСТ Р 53195.4 (таблица А.2).

Цель: защита от необнаруженных на этапах спецификации и реализации ошибок в ПО, неблагоприятно влияющих на безопасность.

Описание: «подушка безопасности» представляет собой внешний монитор, реализованный на независимом компьютере в другой спецификации. Эта «подушка безопасности» касается исключительно гарантии того, чтобы главный компьютер выполнял безопасные, не обязательно корректирующие, действия. Она непрерывно контролирует главный компьютер. «Подушка безопасности» предотвращает вхождение системы в небезопасное состояние. Кроме того, если обнаружится, что главный компьютер вошел в потенциально опасное состояние, система должна возвратиться обратно в безопасное состояние с помощью либо «подушки безопасности», либо главного компьютера.

АС и ПО «подушки безопасности» следует классифицировать и квалифицировать в соответствии с подходящим уровнем полноты безопасности SIL.

Более подробное описание данного метода/средства приведено в [158—160].

В.3.5 Многовариантное программирование

Примечание — На этот метод/средство дана ссылка в ГОСТ Р 53195.4 (таблица А.2).

Цель: обнаружение и наложение маски при выполнении программ на не выявленные на этапах проектирования и реализации ошибки ПО для предотвращения критичных для безопасности отказов системы и для продолжения ее работы с высокой надежностью.

Описание: при многовариантном программировании заданная спецификация ПО проектируется и реализуется различными способами *N* раз. Одни и те же входные значения поступают в *N* версий, и результаты, выданные *N* версиями, сравниваются. Если определяется, что результат правильный, он поступает на выходы компьютера.

N версий могут выполняться параллельно на различных компьютерах, либо все версии могут выполняться на одном и том же компьютере, и результаты будут обработаны внутренним голосованием. Для этих *N* результатов в зависимости от применяемых требований могут быть использованы различные стратегии голосования следующим образом:

- если система находится в безопасном состоянии, можно потребовать полного согласия (все *N согласны*), в противном случае используется выходное значение, которое заставит систему перейти в безопасное состояние. Для простых пошаговых систем голосование может происходить в направлении безопасности. В этом случае безопасное действие может быть разбито по шагам, если какая-либо версия реализует пошаговые операции. Этот подход обычно используется только для двух версий ($N = 2$);

- для систем, находящихся в небезопасном состоянии, могут быть реализованы стратегии мажоритарного голосования. В тех случаях, когда отсутствует общее согласие, могут использоваться вероятностные подходы, с тем чтобы максимизировать вероятность выбора правильного значения, например принять среднее значение, временно зафиксировать выходы, пока не будет достигнуто согласие, и т. п.

Этот метод не устраняет ошибки, не выявленные при проектировании программ, а также не устраняет ошибки в интерпретации спецификации, однако он является средством для обнаружения и маскирования ошибок, прежде чем они смогут повлиять на безопасность.

Более подробное описание данного метода/средства приведено в [160—162].

В.3.6 Блоки восстановления

Примечание — На этот метод/средство дана ссылка в ГОСТ Р 53195.4 (таблица А.2).

Цель: повышение вероятности выполнения программой своих заданных функций.

Описание: некоторые различные разделы программы, часто разработанные независимо, предназначены для выполнения одной и той же требуемой функции. Окончательная программа конструируется из таких разделов. Первый раздел, называемый первичным, выполняется первым. Далее происходит тестирование его результатов. Если тест проходит, результат принимается и передается последующим разделам программы. Если тест не проходит, то все побочные эффекты первого раздела сбрасываются и выполняется второй раздел, называемый первой альтернативой. За ним также следует тест, который выполняется, как и в первом случае. При необходимости могут быть предусмотрены вторая, третья и так далее альтернативы.

В.3.7 Восстановление предыдущего состояния

Примечание — На этот метод/средство дана ссылка в ГОСТ Р 53195.4 (таблица А.2).

Цель: обеспечение исправления функциональных операций при наличии одного или нескольких сбоев.

Описание: при обнаружении сбоя система возвращается в первоначальное внутреннее состояние, согласованность которого была подтверждена ранее. Этот метод предполагает частое сохранение внутреннего состояния в так называемых четко определенных контрольных точках. Метод может быть применен глобально (для всей базы данных) или частично (для изменений только между контрольными точками). По завершении операции система должна устранить изменения, которые произошли за это время, путем занесения в журнал (аудиторское отслеживание действий), компенсацией (все результаты этих изменений аннулируются) или внешним (ручным) способом.

Более подробное описание данного метода/средства приведено в [163, 164].

В.3.8 Прямое восстановление функций

Примечание — На этот метод/средство дана ссылка в ГОСТ Р 53195.4 (таблица А.2).

Цель: обеспечение исправления функциональных операций при наличии одного или нескольких сбоев.

Описание: при обнаружении сбоя текущее состояние системы обрабатывается для достижения состояния, которое через некоторое время будет согласовано. Эта концепция особенно подходит для систем реального времени с небольшой базой данных и с высокой скоростью изменения внутреннего состояния. Предполагается, что по меньшей мере часть системного состояния может влиять на окружение, и только на часть системных состояний влияет окружение.

Более подробное описание данного метода/средства приведено в [165].

В.3.9 Методы повторных попыток восстановления неисправностей

Примечание — На эти методы/средства дана ссылка в ГОСТ Р 53185.4 (таблица А.2).

Цель: функциональное восстановление системы из состояния обнаруженного сбоя с помощью методов повторных попыток.

Описание: в случае обнаружения сбоя или ошибочного условия предпринимаются попытки восстановления ситуации путем повторного выполнения того же кода. Восстановление с помощью повторной попытки может быть полным в виде перезагрузки и повторного пуска процедуры, либо небольшим в виде перепланирования и повторного пуска задачи после выполнения блокировки по времени программы или управляющего действия задачи. Методы повторной попытки широко используются при коммуникационных сбоях или при восстановлении от ошибок, и условия повторной попытки могут быть отделены флажками от ошибки протокола связи (контрольная сумма и т. д.) или от подтверждающего ответа блокировки по времени коммуникации.

Более подробное описание данного метода/средства приведено в [165].

В.3.10 Сохранение достигнутых состояний

Примечание — На этот метод/средство дана ссылка в ГОСТ Р 53195.4 (таблица А.2).

Цель: заставить программу безопасно прекратить работу, если она попытается выполнить неразрешенное действие.

Описание: все соответствующие подробные сведения о каждом выполнении программы документируются. При нормальной работе каждое выполнение программы сравнивается с ранее задокументированными сведениями. При обнаружении различий выполняются действия по безопасности.

Документация о выполнении может содержать последовательность индивидуальных шагов «от решения к решению», или последовательность отдельных обращений к массивам, записям или томам, либо к тому и другому.

Возможны различные методы хранения сведений о последовательностях шагов выполнения программы. Могут быть использованы методы хэш-кодирования для отображения этих последовательностей в виде одного большого числа или последовательности чисел. При нормальной работе перед выполнением выходной операции значения чисел, отображающих последовательности шагов выполнения программы, должны быть сопоставлены со значениями, сохраненными в памяти.

Поскольку возможные комбинации таких последовательностей шагов при выполнении одной программы получаются достаточно большими, может оказаться невозможным рассматривать программы как единое целое. В этом случае метод может быть применен на уровне программных модулей.

В.3.11 Постепенное отключение функций

Примечание — На этот метод/средство дана ссылка в ГОСТ Р 53195.4 (таблица А.2).

Цель: обеспечение возможности выполнения наиболее критичных системных функций, несмотря на отказы, путем игнорирования наименее критичных функций.

Описание: этот метод предоставляет приоритеты различным функциям, выполняемым системой. Проект гарантирует, что в случае недостаточности ресурсов для выполнения всех системных функций функции высшего приоритета будут выполнены в предпочтение функциям более низкого приоритета. Например, функции регистрации ошибки и события могут оказаться более низкого приоритета, чем системные функции управления, и в этом случае управление системой будет продолжаться, даже если аппаратные средства из-за регистрации ошибки окажутся неработоспособными. Более того, если аппаратные средства управления системой окажутся неисправными, а аппаратные средства регистрации ошибок останутся работоспособными, то аппаратные средства регистрации ошибок возьмут на себя функцию управления.

Эти соображения относятся в основном к аппаратным средствам, но они применимы также и ко всей СБЗС-системе. Они должны учитываться начиная с самых ранних этапов проектирования.

Более подробное описание данного метода/средства приведено в [166—168].

В.3.12 Исправление ошибок методами искусственного интеллекта

Примечание — На эти методы/средства дана ссылка в ГОСТ Р 53195.4 (таблица А.2).

Цель: обеспечение способности системы гибко реагировать на возможные опасности с использованием сочетания методов данных, модели процессов и анализа надежности СБЗС-системы.

Описание: прогнозирование ошибок (вычисление трендов), исправление ошибок, техническое обслуживание и контролирующие действия могут быть с большой эффективностью поддержаны системами, основанными на искусственном интеллекте, в различных каналах СБЗС-системы, поскольку правила ее поведения могут быть получены непосредственно из спецификации и проверены на соответствие им.

Для различных каналов связи системы прогнозирование ошибок (вычисление тенденций), исправление ошибок, обслуживание и контролирующие действия могут достаточно эффективным способом поддерживаться системами, основанными на методах искусственного интеллекта (AI). Это связано с тем, что правила для таких систем могут быть образованы непосредственно из спецификаций и проверены на соответствие. На основе этого подхода могут быть эффективно исключены некоторые общие ошибки, уже внесенные в спецификацию, путем косвенного изучения некоего уже имеющегося проекта и получения представления о возможных правилах поведения системы, особенно в случае применения комбинации моделей и методов в функциональной и описательной формах.

Методы выбираются таким образом, что ошибки могли быть устранены и влияние отказов могло быть минимизировано для получения требуемой полноты безопасности.

Примечание — Должны быть учтены предупреждения об исправлении ошибочных данных, приведенные в В.3.2, и об отрицательных рекомендациях применения этого метода, приведенные в ГОСТ Р 53195.4 (таблица А.2, пункт 5).

Более подробное описание данного метода/средства приведено в [169].

В.3.13 Динамическая реконфигурация

Примечание — На этот метод/средство дана ссылка в ГОСТ Р 53195.4 (таблица А.2).

Цель: обеспечение функционирования системы, несмотря на внутренний сбой.

Описание: логическая архитектура системы должна быть такой, чтобы ее можно было отобразить в подмножестве доступных ресурсов системы. Архитектура должна быть способна к обнаружению отказа на физическом уровне и далее к повторному преобразованию логической архитектуры в ограниченные функционирующие ресурсы. Несмотря на то что эта концепция, в основном, традиционно ограничена только восстановлением неисправных модулей АС, она применима также к сбоям в ПО при наличии достаточной «избыточности времени прогона» для повторного выполнения программы или наличии достаточных избыточных данных, которые обеспечивают незначительное влияние отдельного и изолированного отказа.

Этот метод должен рассматриваться на первом этапе проектирования системы.

Более подробное описание данного метода/средства приведено в [169].

В.4 Инструменты разработки и языки программирования

В.4.1 Строго типизированные языки программирования

Примечание — Ссылка на данные методы/средства приведена в ГОСТ Р 53195.4 (таблица А.3).

Цель: снижение вероятности ошибок путем использования языка, который обеспечивает высокий уровень проверки компилятором.

Описание: если скомпилирован строго типизированный язык программирования, то проводится много проверок по использованию типов переменных, например в вызовах процедур и доступе к внешним данным. Компиляция может оказаться безуспешной, и будет выдано сообщение об ошибке при любом использовании типа переменных, которое не соответствует заранее установленным правилам.

Подобные языки обычно позволяют определять установленные пользователем типы данных на основе типов данных базового языка (например, целое число, реальное число). Затем эти типы могут быть использованы так же, как и базовый тип. Вводятся строгие проверки для гарантирования использования правильного типа. Эти проверки проводятся для всей программы, даже если она построена из отдельных скомпилированных модулей. Данные проверки гарантируют также, что число и тип аргументов конкретной процедуры соответствуют числу и типу аргументов при ее вызове, даже если к ней обращаются из отдельно скомпилированных программных модулей.

Строго типизированные языки обычно обеспечивают другие аспекты проверенной на практике техники проектирования ПО, например легко анализируемые структуры управления («if», «then», «else», «do», «while» и т. п.), которые приводят к четко структурированным программам.

Типичными примерами строго типизированных языков являются *C ++*, *Delphi, Java, ML, Pascal, ADA, Modula 2.*

Более подробное описание данного метода/средства приведено в [170—173].

В.4.2 Подмножество языка

Примечание — Ссылка на данный метод/средство приведена в ГОСТ Р 53195.4 (таблица А.3).

Цель: снижение вероятности внесения программных ошибок и повышение вероятности обнаружения оставшихся ошибок.

Описание: проводится исследование языка для определения программных конструкций, подверженных ошибкам либо сложных для анализа, например при использовании методов статического анализа. После этого определяется языковое подмножество, которое исключает такие конструкции.

Более подробное описание данного метода/средства приведено в [173].

В.4.3 Сертифицированные средства

Примечание — Ссылка на данный метод/средство приведена в ГОСТ Р 53195.4 (таблица А.3).

Цель: предоставление разработчику на различных этапах разработки ПО необходимых сертифицированных инструментальных средств для обеспечения конкретной степени уверенности в корректности результатов.

Описание: сертификацию инструментальных средств в общем случае допускается проводить независимо, как правило, в национальных органах по сертификации по независимому набору критериев, установленных обычно в национальных или международных стандартах. В идеальном случае инструментальные средства, применяемые на всех стадиях разработки (спецификация, проектирование, кодирование, тестирование и оценка соответствия), а также используемые в управлении конфигурацией, должны быть сертифицированы.

В настоящее время регулярным процедурам сертификации подвергаются только компиляторы (трансляторы); сертификация проводится национальными органами по сертификации. Она заключается в проверке компиляторов (трансляторов) на соответствие национальным (международным) стандартам, например, для языков *ADA* или *Pascal* и в подтверждении соответствия.

Важно отметить, что сертифицированные инструментальные средства и сертифицированные трансляторы обычно сертифицируются только на соответствие стандартам на определенный язык или процесс. Обычно они никак не сертифицируются на соответствие стандартам по безопасности.

Более подробное описание данного метода/средства приведено в [174, 175].

В.4.4 Инструментальные средства, заслуживающие доверия на основании опыта использования

Примечание — Ссылка на данные методы/средства приведена в ГОСТ Р 53195.4 (таблица А.3).

Цель: исключение проблем, обусловленных ошибками транслятора, которые могут появиться во время разработки, верификации и эксплуатации ПО.

Описание: транслятор используется в тех случаях, когда неправильное исполнение многих предыдущих проектов неочевидно. Если отсутствует опыт эксплуатации трансляторов или в них обнаружены любые известные серьезные ошибки, то от таких трансляторов следует отказаться при отсутствии других гарантий корректной работы транслятора (см. В.4.4.1).

Если в трансляторе выявлены небольшие недостатки, то соответствующие языковые конструкции фиксируются и в проектах СБЗС-систем не применяются.

Другим вариантом исключения проблем, обусловленных ошибками транслятора, является ограничение языка до конструкций, признанных общепринятыми.

Доказано, что недоработанные трансляторы служат серьезным препятствием в любой разработке ПО. Такие трансляторы в общем случае делают невозможной разработку ПО СБЗС-систем.

В настоящее время не существует методов подтверждения корректности всего транслятора или отдельных его частей.

В.4.5 Сравнение исходных программ и исполнимых кодов

Цель: удостовериться в том, что инструменты, используемые для создания образа PROM, не вносят в него никаких ошибок.

Описание: образ PROM преобразуется обратно в совокупность «объектных» модулей, а эти «объектные» модули преобразуются обратно в скомпонованные файлы языка, которые затем с помощью подходящих методов сравниваются с фактическими исходными файлами, первоначально использованными для разработки PROM.

Основное преимущество данного метода состоит в том, что инструменты (компиляторы, редакторы связей (компоновщики) и т. п.), используемые для разработки образа PROM, не требуют подтверждения соответствия. Этим методом проверяют правильность преобразования исходного файла, используемого для конкретной СБЗС-системы.

Более подробное описание данного метода/средства приведено в [176—178].

В.4.6 Библиотека проверенных/верифицированных модулей и компонентов

Примечание — Ссылка на данный метод/средство приведена в ГОСТ Р 53195.4 (таблица А.3).

Цель: исключение необходимости многократных повторных проверок или перепроектирования компонентов ПО и АС при каждом новом применении; содействие созданию проектов, которые не были формально или строго проверены, но относительно которых имеется значительная предыстория эксплуатации.

Описание: хорошо спроектированные и структурированные СБЗС-системы строятся из множества компонентов и модулей АС и ПО, которые четко различаются и которые взаимодействуют друг с другом строго специфицированным способом.

Различные СБЗС-системы, созданные для различных применений, могут содержать большое число одинаковых или очень схожих между собой программных модулей или компонентов. Создание библиотеки таких общеприменимых программных модулей позволяет использовать бóльшую часть ресурсов, необходимых для подтверждения соответствия проекта, одновременно для нескольких применений.

Кроме того, использование подобных программных модулей для многих применений дает практическое подтверждение их успешной эксплуатации. Это практическое подтверждение увеличивает доверие пользователей к программным модулям.

Один из подходов, в соответствии с которым программному модулю можно доверять при его практическом использовании, описан в В.2.10.

Более подробное описание данного метода/средства приведено в [179, 180].

В.4.7 Выбор соответствующего языка программирования

Примечание — Ссылка на данный метод/средство приведена в ГОСТ Р 53195.4 (таблица А.3).

Цель: обеспечение в максимальной степени требований настоящего стандарта для специального защищающего программирования, строгой типизации, структурного программирования и, возможно, суждений. Выбранный язык программирования должен обеспечить легко верифицируемый код и простые процедуры разработки, верификации и эксплуатации ПО.

Описание: язык программирования должен быть полностью и однозначно определен. Язык должен быть ориентирован на пользователя или задачу, а не на процессор или платформу. Широко используемые языки программирования или их подмножества должны быть предпочтительнее языков специального применения.

Языки программирования также должны обеспечивать:

- блоковую структуру организации программ;
- проверку времени трансляции;
- проверку типа и границы массива во время выполнения программы.

Язык программирования должен обеспечивать:

- использование небольших и управляемых программных модулей;
- ограничение доступа к данным в конкретных программных модулях;
- определение поддиапазонов переменных;
- любые другие виды конструкций, ограничивающих ошибки.

Если действия системы по обеспечению безопасности зависят от ограничений реального времени, то язык программирования должен обеспечивать также обработку исключений и/или прерываний.

Желательно, чтобы язык программирования обеспечивался соответствующим транслятором, подходящими библиотеками с заранее созданными программными модулями, отладчиком и инструментами для управления и разработки.

В настоящее время еще не ясно, будут ли объектно-ориентированные языки программирования предпочтительнее других общепринятых языков.

К свойствам, которые усложняют верификацию и поэтому должны быть исключены, относятся:

- безусловные переходы (за исключением вызовов подпрограмм);
- рекурсии;
- указатели, динамически распределяемые области памяти или любые типы динамических переменных или объектов;
- обработка прерываний на уровне исходного кода;
- множественность входов или выходов в циклах, блоках или подпрограммах;
- инициализация или декларация неявных переменных;
- вариантные записи и эквивалентность;
- процедурные параметры.

Языки программирования низкого уровня, в частности ассемблеры, обладают недостатками, связанными с их жесткой ориентацией на процессор машины или на определенную платформу.

Желательным свойством языка программирования является его пригодность к созданию программ, выполнение которых предсказуемо. Если используется подходящий конкретный язык программирования, то в нем должно существовать подмножество, которое гарантирует, что выполнение программы предсказуемо. Это подмножество не может быть (в общем случае) статически определено, несмотря на то что многие статические ограничения помогают гарантировать предсказуемое выполнение. Обычно это может потребовать демонстрацию того, что индексы массива находятся в установленных пределах и что числовое переполнение не может возникнуть, и т. п.

Рекомендации по применению некоторых языков программирования приведены в таблице В.3. Обозначения рангов применимости языков программирования следующие:

КР (HR) — крайне рекомендуемый для данного уровня полноты безопасности. Если его не используют, то на этапе планирования должно быть дано подробное обоснование отказа от его применения, согласованное с экспертом;

Р (R) — рекомендуемый для данного уровня полноты безопасности. Степень обязательности его применения ниже, чем в случае рекомендации КР (HR);

-- — отсутствие рекомендаций по применению или неприменению;

НР (NR) — нерекомендуемый к применению для данного уровня полноты безопасности. Если его применяют, то на стадии планирования должно быть приведено подробное обоснование его применения, согласованное с экспертом.

Т а б л и ц а В.3 — Рекомендации по применению языков программирования

Наименование, обозначение языка программирования	Ранг применимости языка для			
	SIL1	SIL2	SIL3	SIL4
1 ADA	КР (HR)	КР (HR)	Р (R)	Р (R)
2 ADA с подмножеством	КР (HR)	КР (HR)	КР (HR)	КР (HR)
3 MODULA-2	КР (HR)	КР (HR)	Р (R)	Р (R)
4 MODULA с подмножеством	КР (HR)	КР (HR)	КР (HR)	КР (HR)

Окончание таблицы В.3

Наименование, обозначение языка программирования	Ранг применимости языка для			
	SIL1	SIL2	SIL3	SIL4
5 PASCAL	КР (HR)	КР (HR)	Р (R)	Р (R)
6 PASCAL с подмножеством	КР (HR)	КР (HR)	КР (HR)	КР (HR)
7 FORTRAN 77	Р (R)	Р (R)	Р (R)	Р (R)
8 FORTRAN 77 с подмножеством	КР (HR)	КР (HR)	КР (HR)	КР (HR)
9 С	Р (R)	--	НР (NR)	НР (NR)
10 Язык С с подмножеством и стандартом кодирования, а также использование инструментов статического анализа	КР (HR)	КР (HR)	КР (HR)	КР (HR)
11 PL/M	Р (R)	--	НР (NR)	НР (NR)
12 PL/M с подмножеством и стандартом кодирования	КЗ (HR)	Р (R)	Р (R)	Р (R)
13 Ассемблер	Р (R)	Р (R)	--	--
14 Ассемблер с подмножеством и стандартом кодирования	Р (R)	Р (R)	Р (R)	Р (R)
15 Многоступенчатые диаграммы	Р (R)	Р (R)	Р (R)	Р (R)
16 Многоступенчатая диаграмма с определенным подмножеством языка	КР (HR)	КР (HR)	КР (HR)	КР (HR)
17 Диаграмма функциональных блоков	Р (R)	Р (R)	Р (R)	Р (R)
18 Диаграмма функциональных блоков с определенным подмножеством языка	КР (HR)	КР (HR)	КР (HR)	КР (HR)
19 Структурированный текст	Р (R)	Р (R)	Р (R)	Р (R)
20 Структурированный текст с определенным подмножеством языка	КР (HR)	КР (HR)	КР (HR)	КР (HR)
21 Последовательная функциональная диаграмма	Р (R)	Р (R)	Р (R)	Р (R)
22 Последовательная функциональная диаграмма с определенным подмножеством языка	КР (HR)	КР (HR)	КР (HR)	КР (HR)
23 Список команд	Р (R)	--	НР (NR)	НР (NR)
24 Список команд с определенным подмножеством языка	КР (HR)	Р (R)	Р (R)	Р (R)

Примечания

1 Системное программное обеспечение включает в себя операционную систему, драйверы, встроенные функции и программные модули, являющиеся частью системы. ПО обычно обеспечивается поставщиком СБЗС-систем (подсистем). Подмножество языка следует выбирать очень внимательно, с тем чтобы исключить сложные структуры, которые могут образоваться в результате ошибок реализации. Следует проводить проверки, чтобы убедиться в правильном использовании подмножества языка программирования.

2 Прикладная программа представляет собой программу, разработанную для конкретного применения СБЗС-системы. Во многих случаях такая программа разрабатывается конечным пользователем либо подрядчиком, ориентированным на разработку прикладных программ. В тех случаях, когда ряд языков программирования поддерживают одни и те же рекомендации, разработчику следует выбрать тот язык, который повсеместно используется персоналом в конкретной промышленности или отрасли. Подмножество языка программирования следует выбирать с особым вниманием, чтобы исключить сложные структуры, которые могут привести к ошибкам реализации.

3 Если конкретный язык программирования не представлен в настоящей таблице, то это не означает, что он исключен. Этот конкретный язык программирования должен соответствовать требованиям настоящего стандарта.

4 О пунктах 15—24 см. ГОСТ Р 53195.4.

Более подробное описание данного метода/средства приведено в [181].

В.5 Верификация и модификация

В.5.1 Вероятностное тестирование

Примечание — Ссылка на данный метод/средство приведена в ГОСТ Р 53195.4 (таблицы А.5, А.7 и А.9).

Цель: получение количественных показателей надежности исследуемой программы.

Описание: количественные показатели могут быть получены с учетом относительных уровней доверия и значимости. В их состав входят:

- вероятность ошибки при запросе;
- вероятность ошибки в течение определенного периода времени;
- вероятность последствий ошибки.

Из этих показателей могут быть получены другие показатели, например:

- вероятность безошибочной работы;
- вероятность живучести;
- доступность;
- среднее время наработки на отказ (MTBF) или частота отказов;
- вероятность безопасного исполнения.

Вероятностные показатели основываются либо на статистических испытаниях, либо на опыте эксплуатации. Как правило, число тестовых примеров или наблюдаемых практических примеров очень велико. Обычно тестирование в режиме запросов занимает значительно меньше времени, чем в непрерывном режиме работы.

Для формирования входных данных тестирования и управления выходными данными тестирования обычно используются инструменты автоматического тестирования. Крупные тесты прогоняются на больших центральных компьютерах с имитацией соответствующей периферии. Тестируемые данные выбираются с учетом как систематических, так и случайных ошибок АС. Например, общее управление тестированием гарантирует профиль тестируемых данных, тогда как случайный выбор тестируемых данных может управлять отдельными тестовыми примерами более детально.

Индивидуальные средства для тестирования, выполнение тестирования и управление тестированием определяются детализированными целями тестирования. Другие важные условия задаются математическими предпосылками, которые должны быть соблюдены, если оценка тестирования удовлетворяет заданным целям тестирования.

Из опыта эксплуатации также могут быть получены вероятностные представления поведения любого тестируемого объекта. Если соблюдаются одинаковые условия, то к оценкам результатов тестирования может быть применен одинаковый математический аппарат.

При использовании этих методов достаточно сложно продемонстрировать на практике сверхвысокие уровни надежности.

Более подробное описание данного метода/средства приведено в [182, 183].

В.5.2 Регистрация и анализ данных

Примечание — Ссылка на данный метод/средство приведена в ГОСТ Р 53195.4 (таблицы А.5 и А.8).

Цель: документирование всех данных, решений и разумного обоснования программных проектов для обеспечения верификации, оценки, подтверждения соответствия и эксплуатации.

Описание: в процессе всего проектирования разрабатывается подробная документация, в которую входят:

- тестирование, выполняемое на каждом программном модуле;
- решения и их обоснования;
- проблемы и их решения.

В процессе проектирования и по завершении проекта эта документация может быть проанализирована на наличие широкого набора информации. В частности, такая информация, использовавшаяся в качестве обоснования при принятии конкретных решений в процессе разработки проекта и очень важная для обслуживания вычислительных систем, не всегда известна инженерам по эксплуатации.

Более подробное описание данного метода/средства приведено в [184].

В.5.3 Тестирование интерфейса

Примечание — Ссылка на данный метод/средство приведена в ГОСТ Р 53195.4 (таблица А.5).

Цель: обнаружение ошибок в интерфейсах подпрограмм.

Описание: возможно применение нескольких уровней детализации или полноты тестирования. К наиболее важным уровням относится тестирование:

- всех интерфейсных переменных с их предельными значениями;
- всех отдельных интерфейсных переменных с их предельными значениями с другими интерфейсными переменными с их нормальными значениями;
- всех значений предметной области каждой интерфейсной переменной с другими интерфейсными переменными с их нормальными значениями;
- всех значений всех переменных в разных комбинациях (возможно только для небольших интерфейсов);
- каждого вызова каждой подпрограммы, уместного при специфицированных условиях тестирования.

Эти тестирования особенно важны, если интерфейсы не обладают способностью обнаруживать неправильные значения параметров. Такие тестирования также важны при генерации новых конфигураций ранее существовавших подпрограмм.

Более подробное описание данного метода/средства приведено в [185].

В.5.4 Анализ граничных значений

Примечание — Ссылка на данный метод/средство приведена в ГОСТ Р 53195.4 (таблицы Б.2, Б.3 и Б.8).

Цель: обнаружение программных ошибок при предельных и граничных значениях параметров.

Описание: предметная входная область программы разделяется на множество входных классов в соответствии с отношениями эквивалентности (см. В.5.7). Тестирование должно охватывать границы и экстремальные значения классов. Данное тестирование проверяет совпадение границы предметной входной области в спецификации с границами, установленными программой. Использование нулевого значения в непосредственных и в косвенных преобразованиях часто приводит к ошибкам. Особого внимания требуют:

- нулевой делитель;
- знаки пробела ASCII;
- пустой стек или элемент списка;
- заполненная матрица;
- ввод нулевой таблицы.

Обычно границы входных значений напрямую соотносятся с границами выходных значений. Для установления выходных параметров в их предельные значения необходимо записывать специальные тестовые примеры. Следует также по возможности рассмотреть спецификацию такого тестового примера, который побуждает выходное значение превысить установленные спецификацией граничные значения.

Если выходные значения являются последовательностью данных, например таблица, то особое внимание следует уделить первому и последнему элементам, а также спискам, содержащим либо ни одного, либо один, либо два элемента.

Более подробное описание данного метода/средства приведено в [186].

В.5.5 Предположение ошибок

Примечание — Ссылка на данный метод/средство приведена в ГОСТ Р 53195.4 (таблицы Б.2 и Б.8).

Цель: исключение ошибки программирования.

Описание: опыт тестирования и интуиция в сочетании со сведениями и заинтересованностью относительно тестируемой системы могут добавить некоторые неклассифицированные тестовые примеры к набору заданных тестовых примеров.

Специальные значения или комбинации значений могут быть подвержены ошибкам. Некоторые вызывающие интерес тестовые примеры могут быть получены из анализа контрольных списков. Следует также рассмотреть, является ли система достаточно устойчивой. Например: следует ли нажимать клавиши на передней панели слишком быстро или слишком часто; что произойдет, если две клавиши нажать одновременно.

Более подробное описание данного метода/средства приведено в [187].

В.5.6 Введение ошибок

Примечание — Ссылка на данный метод/средство приведена в ГОСТ Р 53195.4 (таблица Б.2).

Цель: подтверждение адекватности набора тестовых примеров.

Описание: некоторые известные типы ошибок вводятся (подмешиваются) в программу, и программа выполняется с тестовыми примерами в режиме тестирования. При обнаружении только некоторых подмешанных ошибок тестовый пример становится неадекватным. Отношение числа обнаруженных подмешанных ошибок к общему числу подмешанных ошибок оценивается как отношение числа обнаруженных реальных ошибок к общему числу реальных ошибок. Это дает возможность оценить число остаточных ошибок и, тем самым, остальную работу по тестированию.

$$\frac{\textit{Обнаруженные подмешанные ошибки}}{\textit{Общее число подмешанных ошибок}} = \frac{\textit{Обнаруженные реальные ошибки}}{\textit{Общее число реальных ошибок}}$$

Обнаружение всех подмешанных ошибок может указывать либо на адекватность тестового примера, либо на то, что подмешанные ошибки было слишком легко найти. Ограничениями данного метода являются: порядок получения любых полезных результатов, типы ошибок. Также необходимо, чтобы позиции подмешивания ошибок отражали статистическое распределение реальных ошибок.

Более подробное описание данного метода/средства приведено в [186 — 189].

В.5.7 Классы эквивалентности и разделенное тестирование входов

Примечание — Ссылка на данный метод/средство приведена в ГОСТ Р 53195.4 (таблицы Б.2 и Б.3).

Цель: адекватное тестирование программных средств с использованием минимума тестируемых данных. Тестируемые данные образуются путем выбора частей входных данных предметной области, требующихся для анализа программных средств.

Описание: применяемая стратегия испытаний базируется на отношении эквивалентности входов, которое определяет разделение входной области.

Тестовые примеры выбираются с учетом охвата всех предварительно специфицированных разбиений. Из каждого класса эквивалентности выбирается по меньшей мере один тестовый пример.

Существуют следующие основные возможности разбиения входных данных:

- классы эквивалентности, образованные из спецификации (интерпретация спецификации может быть ориентирована либо на вход, например, когда выбранные значения считаются одинаковыми, либо на выход, например, когда набор значений приводит к одному и тому же функциональному результату);

- классы эквивалентности, образованные в соответствии с внутренней структурой программы (результаты класса эквивалентности определяются из статического анализа программ, например, набор значений обрабатывается одним и тем же способом).

Более подробное описание данного метода/средства приведено в [190—194].

В.5.8 Структурное тестирование

Примечание — Ссылка на данный метод/средство приведена в ГОСТ Р 53195.4 (таблица Б.2).

Цель: применение тестов, анализирующих определенные подмножества структуры программы.

Описание: на основе анализа программы определяется набор входных данных так, чтобы мог быть проанализирован достаточно большой (часто с заранее заданным назначением) процент программных кодов. Средства охвата программы, в зависимости от степени требуемой строгости могут быть различными:

- утверждение — это наименее строгий тест, поскольку можно выполнить все закодированные утверждения без анализа обеих ветвей условного утверждения;

- ветвление — следует проверять обе стороны каждой ветви (это может оказаться непрактичным для некоторых типов кодов защиты);

- составные условия — анализируется каждое условие в составной ветви (связанное оператором И/ИЛИ) (см., например, охват решения модифицированными условиями MCDC, который означает, что каждая точка входа и выхода в программе была задействована по меньшей мере один раз, что любое решение в программе получило все возможные результаты по крайней мере один раз и что для каждого условия в решении был показан независимый результат, влияющий на результирующее решение). Для каждого набора переменных (внутри логического выражения), как истинных, так и ложных, должны быть разработаны Булевы таблицы истинности;

- LCSAJ — последовательность линейного кода и переходов представляет собой любую линейную последовательность закодированных утверждений, включая условные утверждения, заканчивающиеся переходом. Многие потенциальные подпоследовательности могут оказаться невыполнимыми из-за ограничений, которые налагаются на входные данные в результате выполнения предыдущего кода;

- поток данных — выполняющиеся последовательности выбираются на основе используемых данных; например, последовательность, где одна и та же переменная и записывается, и считывается;

- граф вызовов — программа, состоящая из подпрограмм, которые могут быть вызваны из других подпрограмм. Этот граф вызовов представляет собой дерево вызовов подпрограмм в программе. Тесты должны охватывать все вызовы в дереве;

- базовая последовательность — одна из минимального набора конечных последовательностей от начала до конца, когда охвачены все дуги (перекрывающиеся комбинации последовательностей в этом базовом наборе могут сформировать любую последовательность через эту часть программы). Тесты всех базовых последовательностей показали свою эффективность при обнаружении ошибок.

Более подробное описание данного метода/средства приведено в [195—200].

В.5.9 Анализ потоков управления

Примечание — Ссылка на данный метод/средство приведена в ГОСТ Р 53195.4 (таблица Б.8).

Цель: обнаружение низкокачественных и потенциально некорректных структур программ.

Описание: анализ потока управления представляет собой метод статического тестирования для нахождения подозреваемых областей программы, которые не соответствуют оправдавшей себя практике программирования. Программа анализируется, формируя направленный граф, который может быть проанализирован на наличие:

- недоступных фрагментов программы, например безусловных переходов, которые делают фрагменты программы недостижимыми;

- запутанных кодов. Хорошо структурированный код имеет управляющий граф, допускающий сокращение путем последовательного сокращения графа до одного узла. В отличие от этого плохо структурированный код может быть сокращен только до группы, состоящей из нескольких узлов.

Более подробное описание данного метода/средства приведено в [201, 202].

В.5.10 Анализ потоков данных

Примечание — Ссылка на данный метод/средство приведена в ГОСТ Р 53195.4 (таблица Б.8).

Цель: обнаружение низкокачественных и потенциально некорректных структур программ.

Описание: анализ потока данных представляет собой метод статического тестирования, объединяющий информацию, полученную из анализа потока управления, с информацией о том, какие переменные считываются или записываются в различных частях кода. Данный метод может проверять:

- переменные, которые могут быть считаны до присвоения им значений. Такую ситуацию можно исключить, если всегда присваивать значение при объявлении новой переменной;

- переменные, записанные несколько раз, но не считанные. Такая ситуация может указывать на пропущенный код;

- переменные, которые записаны, но никогда не считываются. Такая ситуация может указывать на избыточный код.

Аномальный поток данных не всегда непосредственно соответствует программным ошибкам, но если аномалии исключены, то маловероятно, что код будет содержать ошибки.

Более подробное описание данного метода/средства приведено в [201—203].

В.5.11 Выявление скрытых схем исполнения

Примечание — Ссылка на данный метод/средство приведена в ГОСТ Р 53195.4 (таблица Б.8).

Цель: обнаружение неожидаемых путей или логических потоков в системе, в конкретных условиях инициирующих нежелательные функции или запрещающих выполнение необходимых функций.

Описание: путь паразитной схемы может содержать аппаратные, программные средства, операторы действий или комбинации этих элементов. Паразитные схемы не являются результатом неисправностей аппаратных средств, а представляют собой скрытые условия невнимательного проектирования системы или кодирования прикладных программ, что при определенных условиях может привести к неправильному функционированию системы.

Паразитные схемы разделяют на следующие категории:

- паразитные пути, вызывающие потоки тока, энергии или логических последовательностей по неожидаемому пути или в незаданном направлении;

- паразитная синхронизация, при которой события происходят в неожидаемой или противоречивой последовательности;

- паразитная индикация, вызывающая неоднозначные или ложные изображения условий эксплуатации системы, что может привести к нежелательным действиям оператора;

- паразитные метки, некорректно или неточно размечающие системные функции, например системные входы, коды управления, изображения, шины и т. д., что может ввести в заблуждение оператора, который может выполнить в системе некорректные действия.

Анализ паразитных схем основывается на распознавании базовых топологических комбинаций в аппаратной или программной структуре. Анализ осуществляется с помощью контрольного списка вопросов об использовании базовых топологических компонентов и отношениях между ними.

Более подробное описание данного метода/средства приведено в [204, 205].

В.5.12 Тестирование на символьном уровне

Примечание — Ссылка на данный метод/средство приведена в ГОСТ Р 53195.4 (таблица Б.8).

Цель: показать соответствие между исходным кодом и спецификацией.

Описание: переменные программы оцениваются после замены во всех операторах присваивания левой его части на правую. Условные ветви и циклы преобразуются в булевы выражения. Окончательный результат представляет собой символьное выражение для каждой переменной программы. Оно может быть проверено относительно предполагаемого символьного выражения.

Более подробное описание данного метода/средства приведено в [206, 207].

В.5.13 Формальное доказательство

Примечание — Ссылка на данный метод/средство приведена в ГОСТ Р 53195.4 (таблица А.9).

Цель: верификация (путем доказательства) корректности программ или спецификаций без их исполнения, используя теоретические и математические модели и правила.

Описание: ряд утверждений устанавливается в различных точках программы, и они используются в качестве предусловий и постусловий для различных путей программы. Доказательство демонстрирует, что программа преобразует предусловия в постусловия в соответствии с набором логических правил и завершается.

В настоящем стандарте описаны различные формальные методы, например CCS, CSP, HOL, LOTOS, OBJ, временная логика, VDM и Z (см. В.2.4).

Альтернативным методу формального доказательства является «строгий аргумент». Подготавливается процедура формального доказательства, в которой представлены основные этапы, но включены не все математические подробности. Метод «строгий аргумент» является более слабым методом верификации, устанавливающим, что доказательство было бы возможным, если бы к этому были предприняты попытки.

Более подробное описание данного метода/средства приведено в [207—210].

В.5.14 Метрики сложности программного обеспечения

Примечание — Ссылка на данный метод/средство приведена в ГОСТ Р 53195.4 (таблицы А.9 и А.10).

Цель: прогнозирование характеристик программ исходя из свойств самих программ или их разработки либо предыстории тестирования.

Описание: данные методы оценивают некоторые структурные свойства программных средств и их отноше-

ния к требуемым характеристикам, например надежность или сложность. Для оценки большинства средств требуются программные инструменты. Некоторые применяющиеся метрики перечислены ниже:

- теоретическая сложность графа. Эта метрика может быть применена на раннем этапе жизненного цикла для оценки компромиссных решений и основана на величине сложности графа управления программы, представленной ее цикломатическим числом;

- число способов активизации некоторых программных модулей (доступность) — чем больше программных модулей может быть доступно, тем должна быть бóльшая вероятность их отладки;

- теория метрик Холстеда. При помощи этих средств вычисляют длину программы путем подсчета числа операторов и операндов; данная метрика дает меру сложности и размеры, которые формируют основу для сравнений при оценке будущих разрабатываемых ресурсов;

- число входов и выходов на программный модуль. Сведéние к минимуму числа точек входов/выходов является ключевой особенностью методов структурного проектирования и программирования.

Более подробное описание данного метода/средства приведено в [211—213].

В.5.15 Инспекция программ

Примечание — Ссылка на данный метод/средство приведена в ГОСТ Р 53195.4 (таблица Б.8).

Цель: обнаружение ошибок на всех этапах разработки программ.

Описание: формальный аудит гарантирующих качество документов, направленный на отыскание ошибок. Процедура инспекции (проверки) состоит из пяти этапов: планирование, подготовка, исследование, анализ и учет. Каждый из этих этапов имеет свои конкретные цели. Должна быть проанализирована вся разработка системы (спецификация, проектирование, кодирование и тестирование).

Более подробное описание данного метода/средства приведено в [214, 215].

В.5.16 Сквозной контроль/анализ проекта

Примечание — Ссылка на данный метод/средство приведена в ГОСТ Р 53195.4 (таблица Б.8).

Цель: обнаружение ошибок в различных частях проекта с высокой оперативностью и экономичностью.

Описание: в МЭК опубликовано руководство по общему представлению формального анализа проектов, которое содержит общее описание представления формального анализа проектов, его цели, подробные сведения о различных типах анализа проекта, состав группы анализа проекта и относящиеся к ним обязанности и ответственности. Это руководство содержит также общие руководящие материалы по планированию и выполнению формального анализа проектов, а также конкретные подробные сведения, относящиеся к роли независимых специалистов в группе по анализу проекта. Например, помимо прочего, в функции специалистов входят надежность, поддержка обслуживания и доступность.

В упомянутом выше руководстве МЭК рекомендуется, чтобы формальный анализ проекта проводился для всех новых изделий/процессов, применений и при пересмотрах существующих изделий и производственных процессов, влияющих на функции, производительность, безопасность, надежность, способность анализировать обслуживание, доступность, способность к экономичности и другие характеристики, влияющие на конечные изделия/процессы, пользователей или наблюдателей.

Закодированный сквозной контроль состоит из группы сквозного контроля, выбирающей небольшой набор изложенных на бумаге тестовых примеров, представляющих наборы входных данных и соответствующие предполагаемые выходы для программы. После этого тестовые данные вручную трассируются через логику программы.

Более подробное описание данного метода/средства приведено в [200, 216, 217].

В.5.17 Макетирование/анимация

Примечание — Ссылка на данный метод/средство приведена в ГОСТ Р 53195.4 (таблицы Б.3 и Б.5).

Цель: проверка возможности реализации системы при наличии заданных ограничений. Увязка интерпретации разработчика спецификации системы с ее потребителем для исключения непонимания между ними.

Описание: выделяются подмножество системных функций, ограничения и требования к рабочим параметрам. С помощью инструментов высокого уровня строится макет. На данном этапе не требуется рассмотрение ограничений (например, используемый компьютер, язык реализации, объем программ, обслуживание, надежность и доступность). Макет оценивается по критериям потребителя, и системные требования могут быть модифицированы в свете этой оценки.

Более подробное описание данного метода/средства приведено в [218—220].

В.5.18 Моделирование процесса

Примечание — Ссылка на данный метод/средство приведена в ГОСТ Р 53195.4 (таблица Б.3).

Цель: тестирование функции программной системы вместе с ее интерфейсами во внешнем окружении, не допуская модификации реального окружения.

Описание: создание системы только для целей тестирования, имитирующей поведение управляемого оборудования (УО).

Имитация может осуществляться только программным обеспечением либо сочетанием ПО и АС. Она должна:

- обеспечить входы, эквивалентные входам, которые могут быть при фактической установке УО;

- реагировать на выходные результаты тестирования программных средств способом, точно отражающим объект управления;

- обладать средствами для входных данных оператора, обеспечивающими любые нарушения, с которыми должна справиться тестируемая система.

По завершении тестирования ПО созданная система может тестировать АС с их входами и выходами.

Более подробное описание данного метода/средства приведено в [221, 222].

В.5.19 Требования к реализации

Примечание — Ссылка на данный метод/средство приведена в ГОСТ Р 53195.4 (таблица Б.6).

Цель: установление демонстрируемых требований к рабочим характеристикам системы ПО.

Описание: выполняется анализ как системы, так и спецификаций требований к ПО для спецификации всех общих и конкретных, явных и неявных требований к функционированию.

Каждое требование к функционированию анализируется по очереди для определения:

- критериев успешности результата, который следует получить;
- возможности получения меры критерия успешности;
- потенциальной точности таких результатов измерения;
- этапов проектирования, на которых эти результаты измерения могут быть оценены;
- этапов проектирования, на которых могут быть получены эти результаты измерений.

Затем анализируется целесообразность каждого требования к функционированию для получения списка требований к рабочим характеристикам, критериев успешности результата и возможных результатов измерений. Основными целями являются:

- связь каждой рабочей характеристики по крайней мере с одной мерой;
- выбор (по возможности) точных и эффективных мер, которые могут быть использованы на самых ранних стадиях разработки;
- спецификация важных и факультативных рабочих характеристик и критериев успешности результата;
- использование (по возможности) преимуществ применения одной меры для нескольких рабочих характеристик.

Более подробное описание данного метода/средства приведено в [222—224].

В.5.20 Моделирование реализации

Примечание — Ссылка на данный метод/средство приведена в ГОСТ Р 53195.4 (таблицы А.5, Б.2 и Б.5).

Цель: достижение достаточной для удовлетворения специфицированных требований рабочей производительности системы.

Описание: спецификация требований включает в себя требования к пропускной способности и реакции конкретных функций, возможно, объединенных с ограничениями на использование общих системных ресурсов. Предложенный проект системы сравнивается с установленными требованиями следующим путем:

- создание модели процессов системы и их взаимодействий;
- определение ресурсов, используемых каждым процессом (время процессора, полоса пропускания канала связи, объем памяти и т. п.);
- определение распределения запросов, выдаваемых системе при средних и наихудших условиях;
- вычисление средних и наихудших случаев значений величин пропускной способности и времени отклика для конкретных функций системы.

Для простых систем может оказаться достаточным аналитическое решение, тогда как для более сложных систем более подходящей для получения точных результатов является создание модели системы.

Перед детальным моделированием может быть использована более простая проверка «бюджета ресурсов», которая суммирует требования к ресурсам всех процессов. Если сумма этих требований к системе превышает возможности спроектированной системы, проект считается нереализуемым. Даже в случае, если проект проходит эту простую проверку, моделирование выполнения может показать, что слишком большие задержки и времена откликов происходят из-за недостатка ресурсов. Для исключения такой ситуации инженеры часто проектируют системы, использующие только часть (например, 50 %) общих ресурсов для уменьшения вероятности нехватки ресурсов.

Более подробное описание данного метода/средства приведено в [222, 225, 226].

В.5.21 Проверка на критические и напряженные нагрузки

Примечание — Ссылка на данный метод/средство приведена в ГОСТ Р 53195.3 (таблица Б.6).

Цель: подвержение тестируемого объекта исключительно высокой нагрузке для демонстрации, что тестируемый объект будет легко выдерживать нормальную рабочую нагрузку.

Описание: существует множество тестов для проверки на критические и напряженные нагрузки, например:

- при работе объекта в режиме упорядоченного опроса он подвергается тестированию в единицу времени гораздо чаще, что приводит к бóльшим входным изменениям, чем при нормальных условиях;
- при работе объекта по запросам число запросов к тестируемому объекту увеличивают в единицу времени по сравнению с нормальными условиями;

- если объем базы данных играет важную роль, то этот объем увеличивают относительно объема при нормальных условиях;

- устройства, имеющие решающее влияние, настраивают на их максимальные или минимальные скорости соответственно;

- для экстремальных тестов все факторы, имеющие решающее влияние, по возможности вводят одновременно в граничные условия.

Для указанных выше тестов может быть оценено поведение во времени тестируемого объекта. Можно также исследовать изменения нагрузки и проверить размер внутренних буферов или динамических переменных, стеков и т. п.

Более подробное описание данного метода/средства приведено в [227, 228].

В.5.22 Ограничения на время реакции и объем памяти

Примечание — Ссылка на данный метод/средство приведена в ГОСТ Р 53195.4 (таблица Б.6).

Цель: обеспечение соответствия системы требованиям к параметрам времени и памяти.

Описание: спецификация требований к системе и программному обеспечению включает в себя требования к памяти и времени выполнения системой конкретных функций, возможно, объединенных с ограничениями на использование общих системных ресурсов.

Данный метод выполняется для установления распределения запросов при средних и наихудших условиях. Рассматриваемый метод требует оценки используемых ресурсов и затраченного времени каждой функцией системы. Такие оценки могут быть получены различными способами, например сравнением с существующей системой или макетированием и дальнейшим сравнением времени реакции с критическими системами.

Более подробное описание данного метода/средства приведено в [227, 229–232].

В.5.23 Анализ влияния

Примечание — Ссылка на данный метод/средство приведена в ГОСТ Р 53195.4 (таблица А.8).

Цель: определение влияния, изменяющего или расширяющего программную систему, которому могут подвергаться также и другие программные модули в данной программной системе, а также другие системы.

Описание: перед выполнением модификации или расширения программного обеспечения следует определить влияние модификаций или расширений на программное обеспечение, а также определить, на какие программные системы и программные модули это повлияет.

Далее принимается решение о повторной верификации программной системы. Это зависит от числа подвергнувшихся воздействию программных модулей, их критичности и характера изменений. Возможными решениями могут быть:

- повторная проверка только изменений программного модуля;

- повторная проверка всех подвергнувшихся воздействию программных модулей;

- повторная проверка всей системы.

Более подробное описание данного метода/средства приведено в [200, 233].

В.5.24 Управление конфигурацией программного обеспечения

Примечание — Ссылка на данный метод/средство приведена в ГОСТ Р 53195.4 (таблица А.8).

Цель: обеспечение согласованности результатов работы групп поставщиков составляющих проекта, а также изменений в этих поставках. В общем случае управление конфигурацией применимо к разработке как АС, так и ПО.

Описание: управление конфигурацией ПО представляет собой метод, используемый в течение всей разработки. В сущности, он требует документирования разработки каждой версии, каждой значимой ее поставки и каждой взаимосвязи между различными версиями разработки различных поставщиков. Полученная документация позволяет разработчику определять, как влияет на другие поставки изменение в первой поставке (особенно одного из его компонентов). В частности, системы или подсистемы могут надежно компоноваться (конфигурироваться) из согласованных наборов версий компонентов.

Более подробное описание данного метода/средства приведено в [234, 235].

В.6 Оценка функциональной безопасности

Примечание — Соответствующие методы и средства см. также в Б.6 настоящего стандарта.

В.6.1 Таблицы решений и таблицы истинности

Примечание — Ссылка на данный метод/средство приведена в ГОСТ Р 53195.4 (таблицы А.10 и Б.7).

Цель: обеспечение ясных и согласующихся спецификаций и анализа сложных логических комбинаций и их отношений.

Описание: в данном методе используют бинарные таблицы для точного описания логических отношений между булевыми переменными программы.

Использование таблиц и точность метода позволили применить его в качестве средства анализа сложных логических комбинаций, выраженных в бинарных кодах.

Рассматриваемый метод достаточно легко автоматизируется, поэтому его можно использовать в качестве средства спецификации систем.

В.6.2 Исследование опасности и работоспособности (HAZOP)

Цель: определение угроз безопасности в предлагаемой или существующей системе, их возможных причин и последствий, а также рекомендуемых действий по минимизации вероятности их появления.

Описание: группа специалистов в области создаваемой системы принимает участие в структурном анализе проекта системы путем ряда запланированных совещаний. Они рассматривают как реализацию функций проекта системы, так и способы работы системы на практике (включая действия персонала и процедуры эксплуатации системы). Руководитель группы специалистов инициирует ее участников создавать потенциальные опасности и управляет этой процедурой, описывая каждую часть системы в сочетании с отдельными ключевыми словами: «отсутствует», «более», «менее», «часть целого», «больше чем» (или «так же как и») и «иначе чем». Каждое применимое условие или режим отказа рассматривается с точки зрения реализуемости, причин возникновения, возможных последствий (появляется ли опасность), способа устранения и, в случае устранения, выбора наиболее целесообразного метода.

Затем часто возникает необходимость провести дальнейшее исследование опасностей (методом вероятностной или количественной оценки риска) с целью их более подробного рассмотрения.

Исследование опасностей может выполняться на разных стадиях разработки проекта, однако наиболее эффективным такое исследование может быть на начальных стадиях, с тем чтобы как можно раньше повлиять на основные решения по проектированию и работоспособности системы. Полезно в графике выполнения проекта определить фиксированное время для совещаний продолжительностью не менее половины дня и не более четырех раз в неделю с тем чтобы рассматривать весь поток сопроводительной документации. Сопроводительная документация, выработанная на совещаниях, должна составлять существенную часть досье об опасности/безопасности системы.

Метод HAZOP создавался для производственных процессов, и без модификации его сложно применить к программным элементам программируемых электронных систем (PE-систем — PES). Были разработаны различные производные методы PES HAZOP (или Computer HAZOPs — «CHAZOPs»), которые сопровождались новыми руководящими материалами и/или реализовывали способы систематического охвата системной и программной архитектур.

Более подробное описание данного метода/средства приведено в [236, 237].

В.6.3 Анализ отказов по общей причине

Примечание — Ссылка на данный метод/средство приведена в ГОСТ Р 53195.4 (таблица А.10).

Цель: определение возможных отказов в нескольких системах или нескольких подсистемах, которые могут свести к нулю преимущества избыточности из-за одновременного появления одних и тех же отказов во многих частях системы.

Описание: системы, ориентированные на безопасность объекта, часто используют избыточность аппаратных средств и мажоритарный принцип голосования. Этот подход исключает случайные отказы в компонентах или подсистемах аппаратных средств, которые могут помешать корректной обработке данных.

Однако некоторые отказы могут оказаться общими для нескольких компонентов или подсистем. Например, если система установлена в одном помещении, то недостатки вентиляции могут снизить преимущества избыточности. Это может оказаться верным и для других внешних влияний на систему (например, пожар, затопление, электромагнитные влияния, трещины в печатных платах и землетрясение). Система может быть также подвержена воздействиям, относящимся к ее функционированию и эксплуатации. Поэтому важно, чтобы в рабочих инструкциях были предусмотрены адекватные и хорошо задокументированные процедуры по функционированию и эксплуатации системы, а обслуживающий персонал был хорошо обучен.

Внутренние причины также вносят большой вклад в общее число отказов. Их основой могут являться ошибки проектирования общих или идентичных компонентов и их интерфейсов, в том числе и устаревших компонентов. Анализ отказов по общей причине должен отыскивать также общие дефекты в системе. К методам анализа отказов по общей причине относятся:

- общее управление качеством;
- анализ проектов;
- верификация и тестирование независимой группой;
- анализ реальных ситуаций, полученных из опыта работы аналогичных систем.

Однако область применения такого анализа выходит за рамки только АС. Даже если разные программы используются в разных каналах избыточных систем, возможна некоторая общность в программных подходах, которая может привести к росту отказов по общей причине (например, ошибки в общей спецификации).

Если отказы по общей причине не появляются точно в одно и то же время, то должны быть предприняты меры предосторожности путем сравнения методов, применяемых в различных каналах. При этом использование каждого метода должно приводить к обнаружению отказа до того, как он окажется общим для всех каналов. При анализе отказов по общей причине следует использовать этот подход.

Более подробное описание данного метода/средства приведено в [238, 239].

В.6.4 Модели Маркова

Цель: оценка надежности, безопасности и доступности систем.

Описание: строится граф системы, представляющий состояния системы, связанные с ее отказами (состояния отказов представляются узлами графов). Связи между узлами, представляющие собой события-отказы или события-восстановления, имеют весовые коэффициенты, соответствующие частотам отказов или частотам восстановлений. Предполагается, что переход из состояния *N* в последующее состояние *N* + 1 не зависит от предыдущего состояния *N* – 1. Следует заметить, что события, состояния и частоты отказов могут быть детализированы так, что может быть получено точное описание системы, например обнаруженные или необнаруженные отказы, обнаружение наибольшего отказа и т. п.

Метод Маркова подходит для моделирования многих систем, уровень избыточности которых изменяется со временем вследствие нахождения компонента в состоянии отказа или восстановления. Другие классические методы, например FMEA и FTA, не могут быть адаптированы к моделированию влияний отказов в течение жизненного цикла системы, поскольку не существует простой комбинаторной формулы для вычисления соответствующих вероятностей.

В простейших случаях такую формулу, описывающую вероятности системы, можно найти в литературе или вывести самостоятельно. В более сложных случаях существуют методы упрощения (то есть сокращение числа состояний). Для очень сложных случаев результаты могут быть вычислены с помощью моделирования графа на компьютере.

Более подробное описание данного метода/средства приведено в [240—244].

В.6.5 Структурные схемы надежности

Примечание — Ссылка на данный метод/средство приведена в ГОСТ Р 53195.4 (таблица А.10).

Цель: моделирование в форме диаграмм набора событий, которые должны происходить, и условий, которые должны быть удовлетворены для успешного выполнения операций системы или задач.

Описание: данный метод позволяет сформировать успешный маршрут, состоящий из блоков, линий и логических переходов. Такой успешный маршрут начинается от одной стороны диаграммы и проходит через блоки и логические переходы до другой стороны диаграммы. Блок представляет собой условие или событие, маршрут проходит через него, если условие истинно или событие произошло. Когда маршрут подходит к логическому переходу, то он продолжается, если критерий логического перехода выполняется. Если маршрут достигает какой-либо вершины, то он может продолжаться по всем исходящим из нее путям. Если существует по меньшей мере один успешный маршрут через всю диаграмму, то цель анализа считается достигнутой.

Более подробное описание данного метода/средства приведено в [245, 246].

В.6.6 Моделирование методом Монте-Карло

Примечание — Ссылка на данный метод/средство приведена в ГОСТ Р 53195.4 (таблица Б.4).

Цель: моделирование ситуаций реального мира с помощью программных средств методом генерации случайных чисел.

Описание: моделирование методом Монте-Карло используется для решения двух классов задач:

- вероятностных, в которых для генерации стохастических ситуаций используются случайные числа;
- детерминистических, которые математически преобразуются в эквивалентную вероятностную форму.

При методе Монте-Карло формируются потоки случайных чисел, с тем чтобы генерировать шум при анализе сигналов или добавлять их в случайные смещения или допуски.

Данный метод гарантированно обеспечивает нахождение смещений, допусков или шума в приемлемых диапазонах.

Общие принципы моделирования методом Монте-Карло заключаются в переформулировании задачи так, чтобы полученные результаты были как можно более точными, что позволяет отказаться от решения проблемы в ее исходной постановке.

Более подробное описание данного метода/средства приведено в [247, 248].

Приложение Г
(справочное)

Методы оценки. Вероятностный подход к определению полноты безопасности предварительно разработанных программных средств

Г.1 Общие положения

Настоящее приложение содержит исходные руководящие материалы по использованию вероятностного подхода к определению полноты безопасности ПО СБЗС-систем для предварительно разработанных программ на основе их опыта эксплуатации. Вероятностный подход является наиболее подходящим для оценки операционных систем, библиотечных компонентов, компиляторов и других программных систем. Настоящее приложение содержит также описание возможностей вероятностного подхода, однако его следует использовать только специалистам, компетентным в статистическом анализе.

Предложенные в настоящем приложении методы могут быть также использованы для демонстрации роста уровня полноты безопасности программных средств, которые некоторое время успешно эксплуатировались. Например, программные средства, созданные в соответствии с требованиями ГОСТ Р 53195.4 для SIL1, после соответствующего периода успешной работы в большом числе применений могут продемонстрировать соответствие уровню полноты безопасности SIL2.

Число запросов без отказов при испытании или число часов, необходимое для работы без отказов, для определения конкретного уровня полноты безопасности представлено в таблице Г.1. В таблице Г.1 также обобщены результаты, приведенные в Г.2.1 и Г.2.3 настоящего приложения.

Опыт эксплуатации может быть выражен математически, как показано в Г.2, для дополнения или замены статистического тестирования, а опыт эксплуатации, полученный из нескольких мест эксплуатации, может быть объединен путем добавления конкретного числа обработанных запросов или часов работы в течение эксплуатации, но только в случае, если:

- программная версия, подлежащая использованию в E/E/PE СБЗС-системе, будет идентична версии, для которой предъявлен результат опыта ее эксплуатации;
- их эксплуатационный профиль и входные условия схожи;
- существует эффективная система уведомлений и документирования отказов;
- справедливы принятые в Г.2 предположения.

Таблица Г.1 — Необходимая предыстория для определения уровня полноты безопасности

Уровень полноты безопасности	Значение вероятности отказа при выполнении планируемых функций по запросу (режим работы с низкой интенсивностью запросов)	Число реальных запросов		Значение вероятности опасного отказа в час (режим с высокой интенсивностью запросов или непрерывным запросом)	Общее число часов эксплуатации	
		$1 - \alpha = 0{,}99$	$1 - \alpha = 0{,}95$		$1 - \alpha = 0{,}99$	$1 - \alpha = 0{,}95$
SIL 4	От 10^{-5} включ. до 10^{-4}	$4{,}6 \times 10^5$	3×10^5	От 10^{-9} включ. до 10^{-8}	$4{,}6 \times 10^9$	3×10^9
SIL 3	От 10^{-4} включ. до 10^{-3}	$4{,}6 \times 10^4$	3×10^4	От 10^{-8} включ. до 10^{-7}	$4{,}6 \times 10^8$	3×10^8
SIL 2	От 10^{-3} включ. до 10^{-2}	$4{,}6 \times 10^3$	3×10^3	От 10^{-7} включ. до 10^{-6}	$4{,}6 \times 10^7$	3×10^7
SIL 1	От 10^{-2} включ. до 10^{-1}	$4{,}6 \times 10^2$	3×10^2	От 10^{-6} включ. до 10^{-5}	$4{,}6 \times 10^6$	3×10^6

Примечания

1 Величина $1 - \alpha$ представляет собой уровень доверия.

2 Предпосылки и описание процедур получения числовых значений в настоящей таблице см. в Г.2.1 и Г.2.3 настоящего приложения.

Г.2 Формулы статистического тестирования и примеры их использования

Г.2.1 Простой статистический тест для режима работы с низкой интенсивностью запросов

Г.2.1.1 Исходные предпосылки

Тест применим при следующих предпосылках:

- распределение тестовых данных равно распределению запросов при выполнении операций в режиме он-лайн;
- прохождения тестов статистически не зависят друг от друга в отношении причины отказа;

- для обнаружения любых отказов, которые могут появиться, существует адекватный механизм;
- число тестовых примеров $n > 100$;
- во время прогона n тестовых примеров отказы отсутствуют.

Г.2.1.2 Результаты

Вероятность отказа p (на один запрос) при уровне доверия $1 - \alpha$ определяется из выражения

$$p \le 1 - \sqrt[n]{\alpha} \text{ или } n \ge -\frac{\ln \alpha}{p}.$$

ПРИМЕР

Таблица Г.2 — Вероятности отказа при режиме работы с низкой интенсивностью запросов

Значение уровня доверия $1 - \alpha$	*Вероятность отказов p*
0,95	*3/n*
0,99	*4,6/n*

Для вероятности отказа при запросе для уровня полноты безопасности SIL3 при уровне доверия 95 % применение указанной формулы дает 30 000 тестовых примеров при выполнении условий принятых предпосылок. Результаты для каждого уровня полноты безопасности объединены в таблице Г.1.

Г.2.2 Тестирование входного массива (предметной области) для режима работы с низкой интенсивностью запросов

Г.2.2.1 Исходные предпосылки

Единственная исходная предпосылка состоит в том, что тестируемые данные выбираются так, чтобы обеспечить случайное унифицированное распределение по входному массиву (предметной области).

Г.2.2.2 Результаты

Необходимо определить число тестов n, которые требуются, исходя из порога точности δ, входов для тестируемой функции с низкой интенсивностью запросов (например, безопасное отключение).

Таблица Г.3 — Средние расстояния между двумя точками тестирования

Размер предметного пространства	Среднее расстояние между двумя точками тестирования в произвольном направлении
1	$\delta = 1/n$
2	$\delta = \sqrt[2]{1/n}$
3	$\delta = \sqrt[3]{1/n}$
k	$\delta = \sqrt[k]{1/n}$
Примечание — k может быть любым положительным целым числом. Значения 1, 2 и 3 приведены только в качестве примеров.	

ПРИМЕР

Рассмотрим безопасное отключение, которое зависит только от переменных A и B. Если проверкой было установлено, что пороговые значения, которые разделяют входную пару переменных A и B, определены с точностью до 1 % диапазона измерения A или B, то число равномерно распределенных тестовых примеров, требуемое в области A и B, будет равно

$$n = 1/\delta^2 = 10^4.$$

Г.2.3 Простой статистический тест для режима с высокой интенсивностью запросов или непрерывного режима работы

Г.2.3.1 Исходные предпосылки

Тест применим при следующих предпосылках:

- распределение тестовых данных такое же, как и распределение данных при выполнении операций в режиме с внешним управлением он-лайн;

- относительное уменьшение вероятности отсутствия отказа пропорционально продолжительности рассматриваемого интервала времени и постоянно в противном случае;
- для обнаружения любых отказов, которые могут появиться, существует адекватный механизм;
- тест выполняется в течение времени тестирования *t*;
- во время тестирования *t* никаких отказов не происходит.

Г.2.3.2 Результаты

Соотношение между интенсивностью отказов λ, уровнем доверия $1-\alpha$ и временем тестирования *t* имеет вид

$$\lambda = -\frac{\ln\alpha}{t}.$$

Интенсивность отказов обратно пропорциональна среднему времени наработки на отказ (*MTBF*):

$$\lambda = \frac{1}{MTBF}.$$

Примечание — В настоящем стандарте не делается различий между интенсивностью отказов в час и частотой отказов в час. Строго говоря, вероятность отказа *F* связана с частотой отказов *f* выражением $F = 1 - e^{-ft}$, однако область применения настоящего стандарта охватывает частоту отказов менее 10^{-5} 1/ч, а для небольших значений частоты справедливо $F \sim f \cdot t$.

ПРИМЕР

Таблица Г.4 — Вероятности отказа для режима с высокой интенсивностью запросов или непрерывным запросом

Значение уровня доверия $1-\alpha$	*Вероятность отказов в час* γ
0,95	*3 t*
0,99	*4,6/t*

Для подтверждения того, что среднее время наработки на отказ составляет по меньшей мере 10^8 час. с уровнем доверия 95 %, требуется время тестирования 3×10^8 ч и должны быть соблюдены исходные предпосылки. Число тестов, необходимое для каждого уровня полноты безопасности, — в соответствии с таблицей Г.1.

Г.2.4 Полное тестирование

Программу можно рассматривать как урну, содержащую *N* шаров. Каждый шар представляет собой конкретное свойство программы. Шары извлекаются случайно и заменяются после проверки. Полное тестирование достигается, если все шары извлечены.

Г.2.4.1 Исходные предпосылки

Тест применим при следующих предпосылках:

- распределение тестируемых данных таково, что каждое из *N* свойств программы тестируется с равной вероятностью;
- тесты проводятся независимо друг от друга;
- каждый появляющийся отказ обнаруживается;
- число случаев тестирования $n >> N$;
- во время *n* случаев тестирования отказы не появляются;
- каждый прогон теста контролирует одно свойство программы (свойство программы — это то, что может быть протестировано во время одного прогона теста).

Г.2.4.2 Результаты

Вероятность тестирования всех свойств программы *p* определяется выражением

$$p = \sum_{j=0}^{N-1}(-1)^j\left(\frac{N}{j}\right)\left(\frac{N-j}{N}\right)^n \quad \text{или} \quad p = 1 + \sum_{j=1}^{N}(-1)^j C_{jN}\left(\frac{N-j}{N}\right)^n,$$

где $C_{jN} = \dfrac{N(N-j)...(N-j+1)^n}{j!}$.

При оценке этого выражения обычно только первые его члены имеют значение, поскольку в реальных условиях выполняется соотношение $n >> N$, что делает все члены этого выражения при большом значении *j* несущественными. Это видно из таблицы Г.5.

ПРИМЕР

Рассмотрим программу, которая имела несколько инсталляций в течение нескольких лет. За это время она выполнялась по меньшей мере $7,5\times10^6$ раз. Предположим, что каждое 100-е выполнение программы соответствует перечисленным выше исходным предпосылкам (см. Г.2.4.1). Поэтому для статистической оценки могут быть приняты $7,5\times10^4$ выполнений программы. Если предположить, что 4000 тестовых прогонов программы могут выполнить исчерпывающее тестирование, считая такую оценку консервативной, то в соответствии с таблицей Г.5 вероятность того, что не все будет протестировано, составляет $2,87\times10^{-5}$.

При N = 4000 значения первых членов в зависимости от n представлены в таблице Г.5.

Таблица Г.5 — Вероятность тестирования всех свойств программы

Число случаев тестирования n	*Вероятность тестирования всех свойств программы p*
5×10^4	$1 - 1,9 \times 10^{-2} + 1,10\times10^{-4} - \ldots$
$7,5 \times 10^4$	$1 - 2,87 \times 10^{-5} + 4\times10^{-10} - \ldots$
1×10^5	$1 - 5,54 \times 10^{-8} + 1,52\times10^{-15} - \ldots$
2×10^5	$1 - 7,67 \times 10^{-19} + 2,9\times10^{-37} - \ldots$

На практике такие оценки должны быть консервативными.

Более подробные сведения по оценке полноты безопасности ПО систем приведены в [88, 148, 249, 250].

Библиография

[1] Ларионов А. М., Майоров С. А., Новиков Г. И. Вычислительные комплексы, системы и сети. Л.: Ленинградское отделение ЭНЕРГОАТОМИЗДАТ, 1987. http://sergey.weblab.ru/AVSiS/book/ Larionov-VKSiS.htm (дата обращения 30.06.2009)

[2] Денисенко В. В. Компьютерное управление технологическим процессом, экспериментом, оборудованием. М.: Горячая линия-Телеком, 2009, 608 с., ил.

[3] Сети хранения данных Fibre Channel. Аппаратные средства технологии Fibre Channel. http:// www.fibrechannel.ru/app.htm (дата обращения 29.06.2009)

[4] Компания Backhoff. Комплексная система противоаварийной защиты TwinSAFE // Автоматизация в промышленности. Июнь 2006, с. 31—34

[5] Компания Backhoff. «Желтые» модули противоаварийной защиты работают по промышленной шине//Автоматизация в промышленности. Январь 2005, с. 36—38

[6] Дудкин А. В. (Backhoff GmbH) ПО TwinCAT CNT решает сложные задачи движения по заданной траектории //Автоматизация в промышленности. Май 2004, с. 52—54

[7] Жуков В. В., Лабковский М. Д. Регулировка электромеханических и радиотехнических приборов и систем: Учеб. пособ. для сред., проф.-техн. училищ. М.: Высш. шк., 1984, 200 с., ил. (Профессионально-техническое образование)

[8] Платунов А., Постников Н., Чистяков А. Механизм граничного сканирования в неоднородных микропроцессорных системах. http://www.chipnews.ru/html.cgi/arhiv/00_10/stat_8.htm (дата обращения 27.03.2009)

[9] Грушвицкий Р., Ильин И., Михайлов М. Метод граничного сканирования для смешанных сигналов //Компоненты и технологии. № 8, 2006

[10] Грушвицкий Р., Ильин И., Михайлов М. Метод граничного сканирования для смешанных сигналов. http://www.kit-e.ru/articles/plis/2006_8_118.php (дата обращения 27.03.2009)

[11] Система безопасного отключения для MOVIDRIVE® MDX60B/61B — Условия применения. Издание «SEWEurudrive» — On-line, 03.2004. http://www.sew-eurodrive.ru/files/pdf/11255064.pdf (дата обращения 06.07.2009)

[12] Применение сертифицированных устройств безопасности производства немецкой компании WIELAND ELECTRIC GMBH на российских предприятиях. Информационно-консультативное издание «Технадзор». Май 2007, № 6

[13] Брагин Г. Безопасность и сертификация. http://www.safemar.ru/articles.php?id=10 (дата обращения 06.07.2009)

[14] Система безопасного отключения для MOVIDRIVE® MDX60B/61B — Условия применения. Издание SEWEurudrive, 03.2004. (дата обращения 06.07.2009). http://www.sew-eurodrive.ru/files/pdf/11255064.pdf (дата обращения 06.07.2009)

[15] Применение сертифицированных устройств безопасности производства немецкой компании WIELAND ELECTRIC GMBH на российских предприятиях. Информационно-консультативное издание «Технадзор». Май 2007, № 6

[16] Хотек М. Методы достижения высокой отказоустойчивости: Windows &. NET Magazine/RE: Открытые системы. http://www.rnivc.kis.ru/?id=420 (дата обращения 27.03.2009). Постоянный адрес статьи: http:// www.osp.ru/win2000/sql/312_4.htm

[17] Неплохов И. Мировые тенденции развития адресно-аналоговых систем пожарной сигнализации. http:// articles.security-bridge.com/articles/13/11792 (дата обращения 27.03.2009)

[18] Щербина В. И. Комплексные системы безопасности высотных и многофункциональных зданий и сооружений. Построение систем, технические средства, рекомендации по применению. М.: Изд-во УКСБиИО, 2006. 216 с., ил. (Учебно-методическое, справочное пособие)

[19] Харченко В., Юрченко Ю. IOTS-подход: анализ вариантов структур отказоустойчивых бортовых комплексов при использовании электронных компонентов Industry // Технология и конструирование в электронной аппаратуре, 2003, № 2

[20] Харченко В., Юрченко Ю. IOTS-подход: анализ вариантов структур отказоустойчивых бортовых комплексов при использовании электронных компонентов Industry. http://www.chipinfo.ru/literature /chipnews/200307/7.html (дата обращения 12.07.2009)

[21] Вернер М. Основы кодирования: Учебник для вузов. М.: Техносфера, 2004, 286 с

[22] Блох Э. Л., Зяблов В. В. Обобщенные каскады-коды: алгебраическая теория и сложность реализации. М.: Связь, 1976

[23] Блох Э. Л., Зяблов В. В. Линейные каскады-коды. М.: Наука, 1982

[24] Питерсон У., Уэлдон Э. Коды, исправляющие ошибки. Пер. с англ. Изд. 2-е, М.: Мир, 1976, 596 с.

[25] Конопелько В. К., Липницкий В. А. Теория норм синдромов и перестановочное декодирование помехоустойчивых кодов. М.: Эдиториал УРСС, 2004, 176 с.

[26] Жирнов М. Н. Анализ методов и синтез программно-технического комплекса для диагностирования дискретных систем на основе эталонных моделей. Вологодский государственный технический университет. http://nit.miem.edu.ru/2006/sb/section1/112.htm (дата обращения 28.03.2009)

[27] Многопроцессорные системы. Классификация систем параллельной обработки данных. http://www.lcard.ru/~nail/database/skbd/glava_10.htm (дата обращения 07.07.2009)
[28] Калядин А. Отладчики микроконтроллеров и их применение в разработке микроконтроллерных приложений. Мир компьютерной электроники / МКА — On-Line. http://www.mka.ru/?p=42051 (дата обращения 12.07.2009)
[29] Встраиваемый контроллер самотестирования памяти ARM11. Техническое руководство (ARM11 Memory Built-In Self Test Controller Technical Reference Manual) www.htmldatasheet.ru/pdf/ arm/arm11.pdf (дата обращения 28.03.2009)
[30] Интеллектуальные САПР. Таганрог, Известия ЮФУ. Технические науки — тематический сборник. Сентябрь 2008, № 9. http://yandex.ru/yandsearch?p=2&text=тестирование%20ОЗУ%20и%20ПЗУ,тест%20Abraham (дата обращения 12.07.2009)
[31] Методы и алгоритмы тестирования памяти ЭВМ с обнаружением кратных функциональных неисправностей: Автореф. диссертация канд. техн. наук (05.13.15 — вычислительные машины и системы) / Новиков А. С.; Науч. рук. Шаршунов С. Г. — Владивосток: ДВГТУ. [s. n.], 2002. 18 с.
[32] Харкевич А. А. Борьба с помехами. М.: Наука, гл. ред. физ.-мат. лит., 1965
[33] Хемминг Р. В. Теория кодирования и теория информации. М.: Мир, 1983
[34] Питерсон У., Уэлдон Э. Коды, исправляющие ошибки: Пер. с англ. / Под ред. Р. Л. Добрушина и С. И. Самойленко. М.: Мир, 1976, 594 с.
[35] Морелос-Сарагоса Р. Искусство помехоустойчивого кодирования. Методы, алгоритмы, применение. М.: Техносфера, 2005
[36] Хмельнов А. Е. Организация ввода/вывода. Страница Хмельнова Алексея Евгеньевича — On-Line. http://hmelnov.icc.ru/stud/lit/ Shnitman/143-2.html (дата обращения 12.07.2009)
[37] Хмельнов А. Е. Системы высокой готовности и отказоустойчивые системы. Страница Хмельнова Алексея Евгеньевича — On-Line. http://hmelnov.icc.ru/stud/lit/Shnitman/143-4.html (дата обращения 12.07.2009)
[38] RAID. Глоссарий промышленной компании «СПЛАЙН». http://www.spline.ru/information/reviews/interface/SCSI_glossary (дата обращения 12.07.2009)
[39] Интеллектуальный дисковый массив RAID 6. Портал «NStor». http://www.nstor.ru/ru/catalog/76/88.html (дата обращения 12.07.2009)
[40] Ватье Ж.-К. Таблицы принятия решений: техника проведения тестирования с использованием Functional Tester от IBM Rational. Software Services, IBM: Пер. с англ. http://www.interface.ru/home.asp?artId=1170 (дата обращения 07.07.2009)
[41] Ематин В., Закис А., Новичков А., Шкляева Н., Подоляк О. Автоматизация процесса тестирования при помощи методологии и инструментальных средств IBM Rational. Ч. 1. http://www.software-testing.ru/library/7-vendor-papers/156-ibm-rational (дата обращения 08.06.2009)
[42] Новичков А. Автоматизация процесса тестирования при помощи методологии и инструментальных средств IBM Rational. Ч. 2. http://www.software-testing.ru/library/7-vendor-papers/155----------ibm-rational--2- (дата обращения 08.07.2009)
[43] Новичков А. Автоматизация процесса тестирования при помощи методологии и инструментальных средств IBM Rational. Ч. 3. http://www.software-testing.ru/library/vendors/154----------ibm-rational--3- (дата обращения 08.06.2009).
[44] Денисенко В. В. Компьютерное управление технологическим процессом, экспериментом, оборудованием. М.: Горячая линия-Телеком, 2009, 608 с., ил.
[45] Шнитман В. З., Кузнецов С. Д. Серверы корпоративных баз данных. Информационно-аналитические материалы. Портал «Сервер» On-Line. http://www.ods.com.ua/win/rus/db/skbd/contents.htm. (дата обращения 12.07.2009)
[46] FAQ по активному воздушному охлаждению. Портал «Перегрева НЕТ!» http://www.peregreva.net/fan_basics1.html (дата обращения 15.07.2009)
[47] Баранов В. Термоэлектрический кулер Titan Amanda TEC. Портал «3DNews». http://www.3dnews.ru/cooling/titan_amanda_tec/ (дата обращения 14.07.2009)
[48] Новый твердотельный вентилятор посрамляет традиционные кулеры! http://glamurnenko.com/archives/173 (дата обращения 15.07.2009)
[49] Задерновский А. А., Ривлин Л. А. Лазерное охлаждение полупроводника (оптическая тепловая машина) // Квант. Электроника, 1996, № 23 (12), с. 1131—1133
[50] Метрологическое оборудование для контактных и бесконтактных средств измерений температуры, теплофизических и линейно-угловых измерений. Портал ОАО «Эталон». http://www.omsketalon .ru/?action=poverka (дата обращения 14.07.2009)
[51] Эталонные датчики. Портал «Элемер». http://www.elemer.ru/files/ articles/listovka_pkds_210.pdf. (дата обращения 14.07.2009)
[52] Лукьянченко А., Федоров А., Соколов Д.В., Ломаев Е.Н., Донг Хынг Ч. Газовые пожарные извещатели. Теоретические основы и практическое применение // Системы безопасности. 2007, № 6. http://daily.sec.ru/dailypbl show.cfm?rid=6&pid=20660&pos=2&stp=25 (дата обращения 14.07.2009)
[53] Охранный извещатель Bosch Blue Line P1-P. http://www.fbgroup.ru/ indexshop.php?scid=1164&sgid=8372. (дата обращения 14.07.2009)

[54] Роздин И. А. Безопасность производства и труда на химических предприятиях. М.: Колосс. 2006, 254 с. (Серия: Учебники и учебные пособия для высших учебных заведений)
[55] ISO/IEC 15289:2006 Systems and software engineering — Content of systems and software life cycle process information products (Documentation)
[56] ISO/IEC 90003:2004Software engineering — Guidelines for the application of ISO 9001:2000 to computer software
[57] Шалыто А. А. SWITCH-технология. Алгоритмизация и программирование задач логического управления. СПб.: Наука, 1998, 628 с.
[58] Шалыто А. А. Логическое управление. Методы аппаратной и программной реализации алгоритмов. СПб.: Наука, 2002, 784 с.
[59] Красилов А. А. Информатика в семи томах. Т. 6. Методы информатики (Изобретание, проектирование, разработка и сопровождение). М., 1997, 2003
[60] Сети Петри. Отчет лаб. 11 СИАПУ ДВО РАН. http://www.iacp.dvo.ru/ lab 11/otchet/ot2000/pn3.html (дата обращения 06.04.2009)
[62] Простые сети Петри. Отчет лаб. 11 СИАПУ ДВО РАН. http://www.iacp.dvo.ru/lab 11/otchet/ot2000/pn3.html#simple (дата обращения 06.04.2009)
[63] Цветные сети Петри. Отчет лаб. 11 СИАПУ ДВО РАН. http://www.iacp. dvo.ru/lab 11/otchet/ot2000/pn3.html#color (дата обращения 06.04.2009)
[64] Язык предписаний. Отчет лаб. 11 СИАПУ ДВО РАН. http://www.iacp. dvo.ru/lab 11/otchet/ot2000/Inscriptions.html (дата обращения 06.04.2009)
[65] Голенков Е. А., Соколов А. С. Метод автоматического построения модели параллельной программы в терминах сетей Петри. Вычислительные методы и программирование. Т. 6. № 2. Изд-во Московского университета, 2005, с. 77—82
[66] Прозоров А. Лекция 4. Моделирование сущностей http://rtlab.ru/ lections/ lec04 (дата обращения 07.04.2009)
[67] Константайн Л., Локвуд Л. Разработка программного обеспечения. СПб.: Питер, 2000, 592 с.
[68] Трахтенгерц Э. А. Компьютерная поддержка принятия решений. М.: Наука, 1998. http:// www.masters.donntu.edu.ua/2004/kita/petrov/ library/lec1.htm (дата обращения 16.07.2009)
[69] Чекинов Г. П., Чекинов С. Г. Применение технологии многоагентных систем для интеллектуальной поддержки принятия решения (ИППР). Сетевой электронный научный журнал «СИСТЕМОТЕХНИКА». 2003, № 1. http://systech.miem.edu.ru/2003/n1/Chekinov.htm (дата обращения 07.04.2009)
[70] Йордон Э., Аргила С. Структурные модели в объектно-ориентированном анализе и проектировании. М.: Лори, 1999, 288 с.
[71] Дубинин В. Н., Зинкин С. А. Языки логического программирования в проектировании вычислительных систем и сетей: Учеб. пособие. Пенза: Изд-во Пенз. гос. техн. ун-та, 1997, 88 с.
[72] Ларман К. Применение UNL 2.0 и шаблонов проектирования. Пер. с англ. А. Ю. Шелестова. Изд. 3-е, Издательский дом «Вильямс», 2009, 727 с.
[73] Златин И. Л. Systemview 6.0 (SystemVue). Системное проектирование радиоэлектронных устройств. М.: Горячая линия-Телеком, 2006, 424 с.
[74] Загидуллин Р. Ш., Стешенко В. Б., Карутин С. Н. SystemView. Системотехническое моделирование устройств обработки сигналов. М.: Горячая линия-Телеком, 2005, 294 с.
[75] Потапов Ю. В. Protel DSP. М.: Горячая линия-Телеком, 2006, 276 с., ил. (Серия «Инструменты разработчика»)
[76] Multisim, LabVIEW и Signal Express. Практика автоматизированного проектирования электронных устройств / Р. Ш. Загидуллин. М.: Горячая линия-Телеком, 2009, 366 с., ил. (Серия «Современная электроника») http:// www.altium.com/products/altium-designer/en/ altium-designer_home.cfm (дата обращения 07.04.2009)
[77] Башлы П. Н. Информационная безопасность: Учеб. пособие. М.: Феникс, 2006, 253 с., ил.
[78] Галле К. Полезные советы по разработке и отладке электронных схем. М.: ДМК Пресс, 2008, 208 с.
[79] Бек К. Экстремальное программирование: разработка через тестирование. СПб.: Питер, 2003, 224 с.
[80] Канер С., Фолк Д., Енг Кек Нгуен. Тестирование программного обеспечения. М.: ДиаСофт, 2001, 538 с.
[81] Дюваль П., Гловер Э. Непрерывная интеграция. Улучшение качества программного обеспечения и снижение риска. М.: «Вильямс», 2008, 240 с.
[82] Бейзер Б. Тестирование черного ящика. Технологии функционального тестирования программного обеспечения и систем. СПб.: Питер, 2004, 320 с.
[83] Синицын С. В., Налютин Н. Ю. Верификация программного обеспечения. М.: Бином. Лаборатория знаний «Интуит», 2008, 368 с.
[84] Тэллес М., Хсих Ю. Наука отладки. КУДИЦ — ОБРАЗ, 2003, 560 с.
[85] Терехов С. А. Нейросетевые аппроксимации плотности распределения вероятности в задачах информационного моделирования. Научная сессия МИФИ-2002. IV Всероссийская науч.-техн. конф. «Нейроинформатика-2002»: Лекции по нейроинформатике. Ч. 2. М.: МИФИ, 2002, 172 с.
[86] Казиев В. М., Казиев К. В. Информатика: Задачи и тесты. М.: Просвещение, 2007, 191 с.
[87] Казиев В. М. Введение в практическое тестирование: Курс Интернет-университета информационных технологий (ИНТУИТ.ру) http://www .intuit.ru/department/informatics/practest (дата обращения 02.05.2009)

[88] Казарин О. В. Безопасность программного обеспечения компьютерных систем. М.: МГУЛ, 2003, 212 с. http://scanner.narod.ru/link/Safe/ bezopasnost programmnogo obespecheniya1.htm (дата обращения 27.06.2009)

[89] Техника анализа надежности систем. Метод анализа вида и последствий отказа. http://www.standards.ru/doc.aspx?catalogid=mec& classid=-1&search=60812%962006 (дата обращения 07.11.2009)

[90] Анализ видов и последствий потенциальных отказов (FMEA) (Potential Failure Mode and Effects Analysis). Пер. с англ. М.: Приоритет, 2003, 84 с.

[91] AMDEC. Анализ видов и последствий потенциальных дефектов продукции / системы. Руководство SOGEDAC-IV-1-12, 1994. Пер. с франц. Н. Новгород: СМЦ Приоритет, 2001, 16 с.

[92] Леоненков А. Самоучитель UML. Изд. 2-е, СПб.: БХВ-Петербург, 2004, 432 с.

[93] Трофимов С. А. CASE-технологии: практическая работа в Rational Rose. Изд. 2-е, СПб.: Бином. Торговый Дом, 2002, 288 с.

[94] Алымов В. Т., Тарасова Н. П. Техногенный риск: анализ и оценка. М.: ИКЦ Академкнига, 2007, 118 с.

[95] МЭК 60812—2006 Техника анализа надежности систем. Метод анализа вида и последствий отказа. http://www.standards.ru/doc. aspx?catalogid=mec&classid=-1&search=60812 (дата обращения 07.11.2009)

[96] Мушик Э., Мюллер П. Методы принятия технических решений. Пер. с нем. М.: Мир, 1990, 208 с.

[97] Макконелл Д. Анализ алгоритмов. Вводный курс. М.: Техносфера, 200, 304 с.

[98] Шафер Д. Ф., Фатрелл Р. Т., Шафер Л. И. Управление программными проектами: достижение оптимального качества при минимуме затрат. Пер. с англ. М.: Издательский дом «Вильямс», 2004, 1135 с., ил.

[99] Казарин О. В. Безопасность программного обеспечения компьютерных систем. М.: МГУЛ, 2003, 212 с.

[100] Платонов В. В. Программно-аппаратные средства обеспечения информационной безопасности вычислительных сетей. М.: Академия, 2006, 240 с.

[101] Шнайер Б. Секреты и ложь. Безопасность данных в цифровом мире. СПб.: Питер, 2003, 368 с.

[102] Липаев В. Функциональная безопасность программных средств. Информационный бюллетень «Jet Info» 08(135)/2004. Публикация от 27.01.2005

[103] Липаев В. Функциональная безопасность программных средств. http://daily.sec.ru/dailypblshow.cfm?rid=45&pid=11751&pos=7&stp=10&cd=18&cm=5&cy=2005 (дата обращения 05.05.2009)

[104] Состав нормативной базы, регламентирующей процесс разработки, эксплуатации, сопровождения и развития информационной системы железнодорожного транспорта. Информационная система железнодорожного транспорта. Системный проект. Кн. 2 (приложение 2), тема 10.00.76/95.00.00 НИОКР МПС № 29 от 29.01.96 г. М.: НИИЖА, 1997

[105] BS EN 50128:2001 Системы телекоммуникационные, сигнализационные и системы для обработки данных, применяемые на железных дорогах. Программное обеспечение для систем управления и защиты на железных дорогах. Пер. с англ. (Railway applications. Communications, signalling and processing systems. Software for railway control and protection systems) http://emc.belsut.info/docs/standards/DIN_EN/ EN_50128.pdf (дата обращения 05.05.2009)

[106] Йордан Э. Путь камикадзе. Как разработчику выжить в безнадежном море проектов. М.: Лори, 2000, 255 с.

[107] Зотов В. Формирование описаний компонентов для внутрикристальной отладки цифровых устройств и встраиваемых микропроцессорных систем на основе программируемых модулей Xinix CORE Generator Tool // Компоненты и технологии. 2008, № 11

[108] Зотов В. Формирование описаний компонентов для внутрикристальной отладки цифровых устройств и встраиваемых микропроцессорных систем на основе программируемых модулей Xinix CORE Generator Tool. Ч. 2 // Компоненты и технологии. 2008, № 12

[109] Зотов В. Формирование описаний компонентов для внутрикристальной отладки цифровых устройств и встраиваемых микропроцессорных систем на основе программируемых модулей Xinix CORE Generator Tool. Ч. 2 // Компоненты и технологии. 2009, № 2

[110] Зотов В. Формирование описаний компонентов для внутрикристальной отладки цифровых устройств и встраиваемых микропроцессорных систем на основе программируемых модулей Xinix CORE Generator Tool. Ч. 2 // Компоненты и технологии. 2009, № 3

[111] Мейер Б. Объектно-ориентированное конструирование программных систем. М.: Издательско-торговый дом «Русская Редакция», «Интернет-университет информационных технологий», 2005, 1232 с., ил.

[112] Гайсарян С. С. Объектно-ориентированные технологии проектирования прикладных программных систем. http://wm-help.net/books-online/print-page/55841/55841.html (дата обращения 16.07.2009)

[113] Методология JSD. http://wm-help.net/books-online/print-page/55841/55841-37.html (дата обращения 15.05.2009)

[114] Пайл Я. Ада — язык встроенных систем / Пер. под ред. А. А. Красилова. М.: Мир, 1984

[115] Вендров А. Case-технологии. Современные методы и средства проектирования информационных систем. М.: Финансы и статистика,1998, 176 с.

[116] Марка Д.-А., Мак Гоуэн К. Методология структурного аннализа и проектирования SADT (Structured Analysis & Design Technique). Электронная библиотека. http://www.interface.ru/fset.asp?Url=/CASE/ prvv.htm (дата обращения 15.05.2009)

[117] Леоненков А. В. UML 2 Самоучитель. СПб.: БХВ-Петербург, 2007, 576 с.
[118] Энсор Д., Стивенсон Й. Oracle. Проектирование баз данных. BHV-Киев, 2000, 560 с.
[119] Системная информатика. Сб. науч. тр. Вып. 9. Формальные методы и модели информатики. Новосибирск: Изд-во СО РАН, 2004
[120] Макконнелл Д. Основы современных алгоритмов. М.: Техносфера, 2006, 368 с.
[121] Топорков В. В. Модели распределенных вычислений. М.: Физматлит, 2004, 320 с.
[122] Бэкон Д, Харрис Т. Операционные системы. Параллельные и распределенные системы. СПб.: Питер, Издательская группа BHV, 2004, 800 с.
[123] Ступников С. А. Моделирование композитных уточняющих спецификаций. Диссертация канд. техн. наук. М.: Ин-т проблем информатики РАН, 2006. На правах рукописи
[124] Никифоров А. Ю. Язык описания взаимодействия иерархических систем и его персонализация // Программные продукты и системы. 2009, № 1
[125] Дорошенко А. Е., Шевченко Р. С. Система символьных вычислений для программирования динамических приложений. http://oai.org.ua/ index.php/record/view/3144 (дата обращения 25.12.2009)
[126] Блейхут Р. Быстрые алгоритмы цифровой обработки сигналов. М.: Мир, 1989, 448 с.
[127] Григорьев О. М. Аналитико-табличные процедуры для временных логик // Logical Studies. 2000, № 4
[128] Стемпковский А. Л. Методы логического и логико-временного анализа цифровых КМОП СБИС. М.: Наука, 2007
[129] Гуц А. К. Математическая логика и теория алгоритмов. Электронная библиотека. http://mat-ua.narod.ru/mat/Guz-Logika-Algoritmi.htm (дата обращения 16.05.2009)
[130] Лаврищева Е. М., Петрухин В. А. Методы и средства инженерии программного обеспечения: Уч. М.: МФТИ (ГУ), 2006, 304 с.
[131] Разработка технологии верификации управляющих программ со сложным поведением, построенных на основе автоматного подхода. Этап 1. Выбор направления исследований и базовых методов. Отчет № 2007.08.31. СПб.: СПбГУ ИТМО, 2007. http://is.ifmo.ru/verification / 2007 01 patent-verification.pdf (дата обращения 16.05.2009)
[132] Якушин. Программирование с защитой от ошибок. http://www.tspu .tula.ru/ivt/old site/umr/ trpo/node74.html (дата обращения 16.05.2009)
[133] Колесов А. "Go to" — выражение из четырех букв // BYTE Россия, 2001, № 8 (37)
[134] Петраков А. В. Основы практической защиты информации: Учеб. пособие. М.: Салон-Пресс, 2005, 384 с.
[135] Корт С. С. Теоретические основы защиты информации. М.: Гелиос АРВ, 2004, 240 с.
[136] Домашев А., Попов В., Правиков Д., Грунтович М. Программирование алгоритмов защиты информации. Изд. 2-е., М.: НОЛИДЖ, 2002, 416 с.
[137] Липаев В. В. Выбор и оценивание характеристик качества программных средств. Методы и стандарты. М.: СИНТЕГ, 2001, 228 с.
[138] Саттер Г., Александреску А. Стандарты программирования на С ++. 101 правила и рекомендации. М.: «Вильямс», 2005, 224 с.
[139] Сухомлин В. Система программирования тройного стандарта (3С++). Науч.-иссл. вычисл. центр МГУ им. М. В. Ломоносова. http://www. citforum.ru/programming/prg96/94.shtml (дата обращения 16.05.2009)
[140] Шпаковский Г. И., Серикова Н. В. Программирование для многопроцессорных систем в стандарте MPI. Минск: Изд-во БГУ, 2002, 323 с.
[141] Липаев В. В. Программная инженерия. Методологические основы: Учеб. / В. В. Липаев. Гос. ун-т — Высшая школа экономики. — М.: ТЕИС, 2006, 608 с.
[142] Липаев В. В. Системное проектирование сложных программных средств для информационных систем. М.: СИНТЕГ, 2002, 268 с.
[143] Структурное проектирование и структурное программирование. http://www.ssti.ru/kpi/informatika/Content/biblio/b1/inform man/gl 18 2.html (дата обращения 16.05.2009)
[144] Синицын С. В., Налютин Н. Ю. Верификация программного обеспечения. М.: Бином. Лаборатория знаний «Интуит», 2008, 368 с.
[145] Кулямин В. В. Перспективы интеграции методов верификации программного обеспечения: Труды Института системного программирования РАН. http://www.citforum.ru/SE/testing/integration (дата обращения 15.07.2009)
[146] Липаев В. В. Тестирование крупных комплексов программ на соответствие требованиям: Учеб. М.: ИПЦ «Глобус», 2008, 376 с.
[147] Липаев В. В. Системное проектирование сложных программных средств для информационных систем. М: СИНТЕГ, 2002, 268 с.
[148] Липаев В. В. Методы обеспечения качества крупномасштабных программных средств. М.: СИНТЕГ, 2003, 520 с., ил.
[149] Липаев В. В. Документирование сложных программных средств. М.: СИНТЕГ, 2005, 216 с.
[150] Липаев В. В., Филинов Е. Н. Мобильность программ и данных в открытых информационных системах. М.: Научная книга, 1997, 368 с.

[151] Очков В. Принцип неопределенности программирования. http://metod.ce.cctpu.edu.ru/edu/df/se/general/gen_08.html (дата обращения 15.05.2009)
[152] Структурное проектирование и структурное программирование. http://www.ssti.ru/kpi/informatika/Content/biblio/b1/inform_man/gl_18_2.html (дата обращения 16.05.2009)
[153] Липаев В. В. Функциональная безопасность программных средств. М.: СИНТЕГ, 2004, 348 с.
[154] Конопелько В. К., Липницкий В. А. Теория норм синдромов и перестановочное декодирование помехоустойчивых кодов. Изд. 2-е, М.: Эдиториал УРСС, 2004, 176 с.
[155] Керман М. К. Программирование и отладка в Delphi: Учебный курс. М.: «Вильямс», 2003, 672 с.
[156] Власов К. А., Смачёв А. С. Методика автоматизированной проверки возвращаемых кодов ошибок при тестировании программных интерфейсов. Тр. Института системного программирования РАН. М., 2007
[157] Сайков, Б. Сбои компьютера: диагностика, профилактика, лечение. Изд. 2-е, М.: Бином. Лаборатория знаний, 2003, 351 с.
[158] Программный модуль дополнительного мониторинга TSS-2000 Multimonitoring фирмы TSS (Россия). http://www.centers.ru/brands/tss/ model/mod221002280.htm (дата обращения 13.06.2009)
[159] ЕС-16. Блок аварийного резервирования аппаратно-программного комплекса AMS-16/32. Техническое описание и инструкция по эксплуатации. М.: ООО «РОКСТОН», 2005. http://www.escortpro.ru/ data/catalog/instruction/15112005121715.pdf (дата обращения 13.06.2009)
[160] Горбунов-Посадов М. М. Расширяемые программы. М.: Полиптих, 1999, 336 с.
[161] Семенова И. И. Способ построения алгоритмов по многовариантным моделям. Развитие оборонно-промышленного комплекса на современном этапе: Мат. науч.-техн. конф. — Ч. 1. Омск: ОмГУ, 2003, с. 135—137. http://semenova-ii.narod.ru/stat/08.html (дата обращения 13.06.2009)
[162] Семенова И. И. Система автоматизированного построения многовариантных моделей. СибАДИ. http://semenova-ii.narod.ru/stat/08.html (дата обращения 13.06.2009)
[163] Тестирование и отладка приложений на C#. Skillsoft, /DDBS Prospero. http://shop.ddbs.ru/prog _49185.html (дата обращения 19.05.2009)
[164] Роббинс Д. Отладка приложений для Microsoft.NET и Microsoft Windows. Электронная библиотека Brain2life.[com], 42,2 МБ. http://www.brain2life.com/book/597.html (дата обращения 19.05.2009)
[165] Крюков В. А. Операционные системы распределенных вычислительных систем (распределенные ОС). Курс лекций. Лаборатория Параллельных Информационных Технологий. НИВЦ МГУ. Электронная книга. «Parallel.ru». http://www.parallel.ru/krukov/lec7.html (дата обращения 19.05.2009)
[166] Богатырев В. А. Отказоустойчивость распределенных вычислительных систем динамического распределения запросов и размещение функциональных ресурсов. Электронное научно-техническое издание «Наука и образование». Эл. № ФС 77 - 30569. Государственная регистрация № 0420900025. http://technomag.edu. ru/doc/56860.html (дата обращения 19.05.2009)
[167] Таненбаум Э., ван Стенн. Распределенные системы. Принципы и парадигмы. М.: Питер, 2003, 877 с. Формат: pdf, 23.3 MB
[168] Таненбаум Э., Вудхалл А. Операционные системы. Разработка и реализация (+ CD-ROM). СПб.: Питер, 2007, 704 с. Формат: djvu> 9472 кБ
[169] Красилов А. А. Информатика в семи томах. Т. 7. Интеллектуальные системы (Системы решения проблем). «Интеллсист». Интеллектуальные системы общего назначения. М., 1997—2003. http://www. intellsyst.ru/publications/_text/TOM7.shtml (дата обращения 19.05.2009)
[170] Ахо А. В., Лам М.-С., Рави С., Ульман Д.-Д. Компиляторы: принципы, технологии и инструменты. Изд. 2-е, М.: «Вильямс», 2008, 1184 с., ил.
[171] Душкин Р. В. Опыт построения единого комплекса автоматизированных систем управления предприятием. // Инженер. Технолог. Рабочий: Публицистический производственно-технический журнал/ МАШИЗДАТ. Вып. 5, 2009, с. 13—14
[172] Душкин Р. Лекция 1. Вводная лекция. http://roman-dushkin.narod .ru/fp_01.html (дата обращения 07.06.2009)
[173] Ксавье П. Delphi for NET. Руководство разработчика. М.: «Вильямс», 2005, 960 с., ил.
[174] Компилятор полного стандарта языка C++ как ядро систем разработки программного обеспечения. Сб. статей компании «Интерстрон». Приложение к журналу «Компьюлог». 2000, № 3. http://www.interstron.ru/old/pdf/komp_log.pdf (дата обращения 07.06.2009)
[175] Цирлов В., Миронов В., Марков А. Выявление уязвимостей в программном коде // Открытые системы. 2005. № 12. http://www.osp. ru/os/2005/12/380655 (дата обращения 07.06.2009)
[176] Чернов А. В. Анализ запутывающих преобразований программ. Труды ИСП РАН. 2008. http://www.citforum.idknet.com/security /articles/analysis (дата обращения 07.06.2009)
[177] КорменТ.-Х., Лейзерсон Ч.-И., Ривест Р.-Л., Штайн К. Алгоритмы: построение и анализ. Изд. 2-е, М.: Издательский дом «Вильямс», 2008, 1297 с.
[178] Буч Г., Максимчук Р.- А., Энгл М.-У., Янг Б.-Д., Коналлен Д., Хьюстон К.-А. Объектно-ориентированный анализ и проектирование с примерами приложений. Изд. 3-е, М.: Издательский дом «Вильямс», 2008, 720 с.
[179] Макаров А. В., Скоробогатов С. Ю., Чеповский А. М. Common Inter-mediate Language и системное программирование в Microsoft.NET. М.: Интернет-университет информационных технологий, 2006, 314 с. http://www.rus-kniga.biz/tv246-298579.html (дата обращения 07.06.2009)

[180] Елманова Н. Полезные компоненты и утилиты для пользователей Delphi, C++Builder и IB Database: продукты компании BatSoft. Компьютер Пресс — CD. 1999. № 2. http://www.citforum.ru/progra mming/comp/comp02.shtml (дата обращения 07.06.2009)

[181] Элиенс А. Принципы объектно-ориентированной разработки программ. Изд. 2-е, М.: «Вильямс», 2002, 496 с.

[182] Налютин Н. Ю., Синицын С. В. Верификация программного обеспечения. М.: Бином. Лаборатория знаний «Интуит», 2008, 368 с.

[183] Плаксин М. А. Тестирование и отладка программ — для профессионалов будущих и настоящих. М.: Бином. Лаборатория знаний, 2007, 167 с.

[184] Тюрин Ю., Марков А. Анализ данных на компьютере. М.: Инфра-М, 2003, 544 с.

[185] Рубанов В. В., Хорошилов А. В., Шатохин Е. А. T2C: технология автоматизированной разработки тестов базовой функциональности программных интерфейсов. М.: Труды Института системного программирования РАН, 2008 г. http://www.citforum.ru/SE/testing/t2c/ (дата обращения 13.06.2009)

[186] Калбертсон Р., Браун К., Кобб Г. Быстрое тестирование. М.: «Вильямс», 384 с.

[187] Иванова Г. С. Технология программирования. М.: Изд.-во МГТУ им. Н. Э. Баумана, 336 с.

[188] Медина К. Устройства ввода ошибок FBD-памяти для компьютеров IBM System x. http://www.ibm.com/developerworks/ru/library/es-fbd (дата обращения 14.06.2009)

[189] Климант Ю. В. C++. Дистанционное обучение программистов. Уроки по программированию. Урок 8. http://cipg.km.ru/lessons/ci/les08.html (дата обращения 14.06.2009)

[190] Блэк Р. Ключевые процессы тестирования. Планирование, подготовка, проведение, совершенствование. М.: Лори, 544 с.

[191] Макгрегор Д., Сайкс Д. Тестирование объектно-ориентированного программного обеспечения: Практическое пособие. М.: ТИД «ДС», 432 с.

[192] Бейзер Б. Тестирование черного ящика. Технологии функционального тестирования программного обеспечения и систем. СПб.: Питер, 2004, 320 с.

[193] Бек К. Экстремальное программирование: разработка через тестирование. СПб.: Питер, 2003, 223 с.

[194] Майерс Г. Искусство тестирования программ. Пер. с англ. М.: Финансы и статистика, 1982, 176 с.

[195] Синицын С. В., Налютин Н. Ю. Верификация программного обеспечения. Лекция 13: Документация, сопровождающая процесс верификации и тестирования (отчеты) «Интернет университет. Информационные технологии» http://www.intuit.ru/department/se /verify/13 (дата обращения 25.06.2009)

[196] Романюк С. Г. Оценка надежности программного обеспечения. М.: НИИСИ РАН. http://www.uprav.biz/materials/innov/view/2273.html (дата обращения 25.06.2009)

[197] Смагин В. А. Форсированные быстродействием испытания программного обеспечения на надежность. http://sprobv-17.narod.ru (дата обращения 25.06.2009)

[198] Смагин В. А. О форсированных испытаниях программного обеспечения на надежность. http://sprobv-17.narod.ru (дата обращения 25.06.2009)

[199] Смагин В. А. Введение в точностную теорию надежности программного обеспечения. http://sprobv-17.narod.ru (дата обращения 25.06.2009)

[200] Казарин О. В. Безопасность программного обеспечения компьютерных систем. М.: МГУЛ, 2003, 212 с. http://infonet.cherepovets.ru /citforum/security/articles/kazarin (дата обращения 25.06.2009)

[201] Чернов А. В. Анализ запутывающих преобразований программ. Тр. Института системного программирования РАН. М., 2003. http://www.citforum.ru/security/articles/analysis (дата обращения 25.06.2009)

[202] Ковалев В. В., Компаниец Р. И., Маньков Е. В., Дьяченко Д. А., Пустарнаков В. Ф. Анализ и защита потоков управления в исполняемых кодах программ. Информационно-издательский центр CONNECT! Мир связи, 2006, № 4

[203] Ахо А. В., Хопкрофт Д.-Э., Ульман Д.-Д. Структуры данных и алгоритмы. Пер. с англ: Уч. пос. М.: Издательский дом «Вильямс», 2000, 384 с.

[204] Липаев В. Программно-технологическая безопасность информационных систем. http://www.info-system.ru/security/security_pr_tech_ security.html (дата обращения 25.06.2009)

[205] Касперски К. Техника оптимизации программ — эффективное использование памяти. БХВ-Петербург, 2003, 464 с.

[206] Разработка сложного программного обеспечения. http://www.devcomplexsoft.ru (дата обращения 25.06.2009)

[207] Лаврищева Е. М., Петрухин В. А. Методы и средства инженерии программного обеспечения: Уч. М.: МФТИ (ГУ), 2006, 304 с. http://window.edu.ru/window_catalog/files/r41699/lavrishcheva_petrukhin.pdf (дата обращения 26.06.2009)

[208] Тенихин А. Л. Применение формальных методов доказательства при создании безопасных систем. СПбГТУ, кафедра ИБКС. http://www.ssl. stu.neva.ru/ssl/publications/magazine/2000/2/3/tenihin.pdf (дата обращения 26.06.2009)

[209] Боуэн Д.-П., Хинчи М.-Д. Десять заповедей формальных методов. http://www.osp.ru/pcworld/1997/09/157957 (дата обращения 26.06.2009)

[210] Немолочнов О. Ф., Зыков А. Г., Осовецкий Л. Г., Поляков В. И., Петров К. В. Тестирование логических неисправностей вычислительных процессов в программах // Информационные технологии. 2007, № 12, с. 2—5

[211] Колдовский В. Разработка ПО: метрики программных проектов. Киев: Издательский Дом ITC, 2009. http://itc.ua/node/27774 (дата обращения 26.06.2009)

[212] Сбор и публикация проектных метрик в процессе разработки программного обеспечения на базе штатных средств IBM Rational Clear Case. OLAP.ru http://www.olap.ru/home.asp?artId=445 (дата обращения 26.06.2009)

[213] Николс Э., Петерсон Г. Метрики управления качеством защиты приложений. Портал «Открытые системы. СУБД». http://www.osp. ru/os/2007/04/4219959 (дата обращения 26.06.2009)

[214] Кулямин В. В. Компонентный подход в программировании. Лекция 8: Образцы проектирования. БИНОМ. Лаборатория знаний, Интернет-университет информационных технологий — ИНТУИТ.ру. 2006. http:// www.intuit.ru/department/se/compprog/8/1.html (дата обращения 26.06.2009)

[215] Кулямин В. В. Компонентный подход в программировании. Курс лекций. БИНОМ. Лаборатория знаний, Интернет-университет информационных технологий — ИНТУИТ.ру. 2006. http://www.intuit.ru/department/ se/compprog/1 (дата обращения 26.06.2009)

[216] Салливан Э. Время — деньги. Создание команды разработчиков программного обеспечения. Пер. с англ. М.: Издательско-торговый дом «Русская Редакция», 2002, 368 с., ил. http://www.proklondike.com (дата обращения 27.06.2009)

[217] Руководящий документ. Защита от несанкционированного доступа к информации. Часть 1. Программное обеспечение средств защиты информации. Классификация по уровню контроля отсутствия недекларированных возможностей. Утвержден решением председателя Государственной технической комиссии при Президенте Российской Федерации от 4 июня 1999 г., № 114

[218] Бурьяк А. Компактное программирование. Программирование на основе прототипов. Prototype-Based Programming (PBP). http://compact-programming.narod.ru/CP0003.htm (дата обращения 28.06.2009)

[219] Уолш Д. (George Walsh). Создание прототипа программы с помощью библиотеки OpenMP*. Intel, 2006. http://www.intel.com/cd/ids/developer/emea/rus/dc/windows/windows64/191144.htm (дата обращения 28.06.2009)

[220] Павловская Т.А. Программирование на языке высокого уровня. СПб.: Питер, 2007, 432 с.

[221] Дастин Э., Рэшка Д., Пол Д. Автоматизированное тестирование программного обеспечения. Внедрение, управление и эксплуатация. М.: Лори, 2003, 592 с.

[222] Советов Б. Я., Яковлев А. М. Моделирование систем. Изд. 3-е перераб. и доп. Электронная библиотека образовательных и просветительских изданий. М., 2001, 374 с.

[223] ГОСТ 34.602—89 Техническое задание на создание автоматизированной системы

[224] Леффингуэлл Д., Уидриг Д. Принципы работы с требованиями к программному обеспечению. Унифицированный подход. М.: «Вильямс», 2002, 448 с.

[225] Леффингуэлл Д., Уидриг Д. Принципы работы с требованиями к программному обеспечению. Унифицированный подход. М.: «Вильямс», 2002, 448 с.

[226] Пипер Ш., Пол Д., Сколт М. Новая эра в оценке производительности компьютерных систем. (Sean Pieper, Joann Paul, Michael Schulte. A New Era of Performance Evaluation, IEEE Computer, September 2007. IEEE Computer Society, 2007. All rights reserved. Reprinted with permission). Пер. с англ. Портал «Открытые системы — СУБД». http://www.osp.ru/os/2007/09/4569364 (дата обращения 29.06.2009)

[227] Елашкин М. Производительность СУБД и тесты TPC. «BYTE — Платформы и технологии». 2004, № 3 (67). http://www.bytemag.ru/ articles/detail.php?ID=8571 (дата обращения 29.06.2009)

[228] Анализ производительности 64- и 32-разрядных многопроцессорных вычислительных систем в программном комплексе вычислительной гидрогазодинамики STAR-CD. Gjhnfk «iXBT.com». http://www.ixbt. com/ cpu/star-cd-tests.shtml (дата обращения 29.06.2009)

[229] Пушников А. Ю. Введение в системы управления базами данных. Глава 9. Транзакции и целостность баз данных. Портал «CITFORUM.ru». http://www.citforum.ru/database/dblearn/dblearn09.shtml (дата обращения 29.06.2009)

[230] Преодолевая ограничения Windows: физическая память. Портал «Русский хакер.ру». http://rushacker.ru/ index.php?autocom=ibwiki& cmd=article&id=5 (дата обращения 29.06.2009)

[231] Ограничение пользователей Портал «FreeBSD». http://www. freebsd.org.ua/doc/ru_RU.KOI8-R/books/ handbook/users-limiting.html (дата обращения 29.06.2009)

[232] Грудина П. Настройка ограничений пользователям. Портал IT профессионалов «Grudina.ru». http:// grudina.info/articles/freebsd /nastroyka-ogranicheniy-polzovatelyam.html (дата обращения 29.06.2009)

[233] Боне Ш. Эпоха систем, терпимых к изменениям. Портал «Открытые системы — СУБД». http://www.osp.ru/ os/2007/07/4394365 (дата обращения 29.06.2009)

[234] Системы менеджмента качества. Руководящие указания по менеджменту конфигурации. Пер. с англ. Портал «Стандартинформ» http://www.vniiki.ru/doc.aspx?catalogid=iso&classid=-1&search=10007. (дата обращения 29.06.2009)

[235] Белладжио Д., Миллиган Т. Стратегия управления конфигурацией программного обеспечения с использованием IBM Rational ClearCase. Изд. 2-е, М.: ДМК Пресс, 2008, 384 с.

[236] ГОСТ Р 51901.11—2005 (МЭК 61882:2001) Менеджмент риска. Исследование опасности и работоспособности. Прикладное руководство. http://intranet.fastweb.ru/docs/gost/item 566 (дата обращения 29.06.2009)

[237] Шубинский И. Б. Методы анализа рисков нарушения безопасности систем управления. «Евроазия — Вести». 2006, Вып. 6. http://www.eav.ru/publ1p.php?publid=2006-05a14 (дата обращения 29.06.2009)

[238] Общие положения безопасности атомных станций. Приказ ГКЯР Украины от 19.11.2007, № 162. http://www.snrc.gov.ua/nuclear/ru/publish/article/82143 (дата обращения 29.06.2009)

[239] Смит Д.-Д. Безотказность, ремонтопригодность и риск. Практические методы для инженеров, включая вопросы оптимизации надежности и систем, связанных с безопасностью. М.: Группа ИДТ, 2007, 431 с. http://www.centrmag.ru/book2286985.html (дата обращения 29.06.2009)

[240] Мухин О. И. Моделирование систем. Учебник на портале «Stratum.ac.ru» http://stratum.ac.ru/textbooks/modelir/index.html (дата обращения 29.06.2009)

[241] Николенко С. Скрытые марковские модели. Машинное обучение – ИТМО, осень 2006. http://logic.pdmi.ras.ru/~sergey/teaching/mlbayes/06-hmm.pdf (дата обращения 29.06.2009)

[242] Куликов Г. Г., Флеминг П.-Д., Брейкин Т. В., Арьков В. Ю. Марковские модели сложных динамических систем: идентификация, моделирование и контроль состояния (на примере цифровой САУ ГТД). Уфа: Уфим. гос. авиац. техн. ун-т, 1998, 103 с.

[243] Рухман Е. Л., Шеховцов О. И. Марковские модели в задачах помехоустойчивости и надежности передачи информации: Учеб. пособие. Л.: ЛЭТИ, 1981, 78 с.

[244] Моттль В. В., Мучник И. Б. Скрытые марковские модели в структурном анализе сигналов. М.: Физматлит, 1999, 352 с.

[245] Алексеев А. FTA. Дерево отказов как метод структурного анализа. Портал «IT Expert» http://www.itexpert.ru/rus/ITEMS/77-30 (дата обращения 29.06.2009)

[246] ГОСТ Р 51901.14—2007 Менеджмент риска. Структурная схема надежности и булевы методы

[247] Абельсон Х., Сассман Д. Д., Сассман Д. Структура и интерпретация компьютерных программ — язык ЛИСП (Lisp). М.: Добросвет, 2004, 608 с.

[248] Волков И. М., Грачева М. В. Вероятностные методы анализа рисков. http://www.ndc.ru/ru/press/pubs/depo/archive/22/article5.htm (дата обращения 29.06.2009)

[249] Орлов А. И. Теория принятия решений: Учеб. пособие. М.: Март, 2004, 656 с. http://orlovs.pp.ru (дата обращения 27.06.2009)

[250] Орлов А. И. Высокие статистические технологии // Заводская лаборатория. 2003, Т. 69, № 11, с. 55—60. http://orlovs.pp.ru (дата обращения 27.06.2009)

УДК 621.5:814.8:006.354

ОКС 13.110, 13.220.01, 13.310, 13.320, 29.130.20, 35.240

ОКП 43 7000, 43 7100, 43 7200, 43 7280, 70 3000

Ключевые слова: безопасность функциональная, связанные с безопасностью зданий и сооружений системы, методы и средства снижения риска, методы оценки соответствия

Редактор *И. В. Алферова*
Технический редактор *В. Н. Прусакова*
Корректор *Н. И. Гаврищук*
Компьютерная верстка *Т. Ф. Кузнецовой*

Сдано в набор 13.09.2011. Подписано в печать 01.11.2011. Формат 60×84 1/8. Бумага офсетная. Гарнитура Ариал. Печать офсетная. Усл. печ. л. 9,77. Уч.-изд. л. 8,79. Тираж 141 экз. Зак. 1062

ФГУП «СТАНДАРТИНФОРМ», 123995 Москва, Гранатный пер., 4.
www.gostinfo.ru info@gostinfo.ru
Набрано и отпечатано в Калужской типографии стандартов, 248021 Калуга, ул. Московская, 256.