МЕЖГОСУДАРСТВЕННЫЙ СОВЕТ ПО СТАНДАРТИЗАЦИИ, МЕТРОЛОГИИ И СЕРТИФИКАЦИИ
(МГС)
INTERSTATE COUNCIL FOR STANDARDIZATION, METROLOGY AND CERTIFICATION
(ISC)

МЕЖГОСУДАРСТВЕННЫЙ СТАНДАРТ

ГОСТ IEC 61508-3— 2018

ФУНКЦИОНАЛЬНАЯ БЕЗОПАСНОСТЬ СИСТЕМ ЭЛЕКТРИЧЕСКИХ, ЭЛЕКТРОННЫХ, ПРОГРАММИРУЕМЫХ ЭЛЕКТРОННЫХ, СВЯЗАННЫХ С БЕЗОПАСНОСТЬЮ

Часть 3

Требования к программному обеспечению

(IEC 61508-3:2010, IDT)

Издание официальное

Москва
Стандартинформ
2018

Предисловие

Цели, основные принципы и основной порядок проведения работ по межгосударственной стандартизации установлены в ГОСТ 1.0—2015 «Межгосударственная система стандартизации. Основные положения» и ГОСТ 1.2—2015 «Межгосударственная система стандартизации. Стандарты межгосударственные, правила и рекомендации по межгосударственной стандартизации. Правила разработки, принятия, обновления и отмены»

Сведения о стандарте

1 ПОДГОТОВЛЕН Обществом с ограниченной ответственностью «Корпоративные электронные системы» на основе собственного перевода на русский язык англоязычной версии стандарта, указанного в пункте 5

2 ВНЕСЕН Техническим комитетом по стандартизации ТК 058 «Функциональная безопасность»

3 ПРИНЯТ Межгосударственным советом по стандартизации, метрологии и сертификации (протокол от 30 мая 2018 г. № 109-П)

За принятие проголосовали:

Краткое наименование страны по МК (ИСО 3166) 004—97	Код страны по МК (ИСО 3166) 004—97	Сокращенное наименование национального органа по стандартизации
Армения	AM	Минэкономики Республики Армения
Беларусь	BY	Госстандарт Республики Беларусь
Киргизия	KG	Кыргызстандарт
Россия	RU	Росстандарт

4 Приказом Федерального агентства по техническому регулированию и метрологии от 6 сентября 2018 г. № 574-ст ГОСТ IEC 61508-3—2018 введен в действие в качестве национального стандарта Российской Федерации с 1 июля 2019 г.

5 Настоящий стандарт идентичен международному стандарту IEC 61508-3:2010 «Функциональная безопасность систем электрических, электронных, программируемых электронных, связанных с безопасностью. Часть 3. Требования к программному обеспечению» («Functional safety of electrical/electronic/programmable electronic safety-related systems — Part 3: Software requirements», IDT).

Международный стандарт IEC 61508-3:2010 подготовлен подкомитетом 65A «Системные аспекты» Технического комитета по стандартизации IEC 65 «Измерения и управление производственными процессами» Международной электротехнической комиссии (IEC).

При применении настоящего стандарта рекомендуется использовать вместо ссылочных международных стандартов соответствующие им межгосударственные стандарты, сведения о которых приведены в дополнительном приложении ДА

6 ВВЕДЕН ВПЕРВЫЕ

Информация об изменениях к настоящему стандарту публикуется в ежегодном информационном указателе «Национальные стандарты», а текст изменений и поправок — в ежемесячном информационном указателе «Национальные стандарты». В случае пересмотра (замены) или отмены настоящего стандарта соответствующее уведомление будет опубликовано в ежемесячном 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 (www.gost.ru)

Содержание

Введение

Системы, состоящие из электрических и/или электронных элементов, в течение многих лет используются для выполнения функций безопасности в большинстве областей применения. Компьютерные системы (обычно называемые программируемыми электронными системами), применяемые во всех прикладных отраслях для выполнения функций, не связанных с безопасностью, во все более увеличивающихся объемах используются для выполнения функций обеспечения безопасности. Для эффективной и безопасной эксплуатации технологий, основанных на использовании компьютерных систем, чрезвычайно важно, чтобы лица, ответственные за принятие решений, имели в своем распоряжении руководства по вопросам безопасности, которые они могли бы использовать в своей работе.

Настоящий стандарт устанавливает общий подход к вопросам обеспечения безопасности для всех стадий жизненного цикла систем, состоящих из электрических и/или электронных, и/или программируемых электронных (E/E/PE) элементов, которые используются для выполнения функций обеспечения безопасности. Этот унифицированный подход принят для того, чтобы разработать рациональную и последовательную техническую политику для всех электрических систем обеспечения безопасности. Основной целью при этом является содействие разработке стандартов для продукции и областей применения на основе стандартов серии IEC 61508.

Примечание — Примерами стандартов для изделий и областей применения, разработанных на основе стандартов серии IEC 61508, являются [1]—[3].

В большинстве ситуаций безопасность достигается за счет использования нескольких систем, в которых применяются различные технологии (например, механические, гидравлические, пневматические, электрические, электронные, программируемые электронные). Любая стратегия безопасности должна, следовательно, учитывать не только все элементы, входящие в состав отдельных систем (например, датчики, управляющие устройства и исполнительные механизмы), но также и все подсистемы безопасности, входящие в состав общей системы обеспечения безопасности. Таким образом, хотя настоящий стандарт рассматривает электрические/электронные/программируемые (E/E/PE) системы, связанные с безопасностью, предлагаемый в нем подход можно использовать также при рассмотрении систем, связанных с безопасностью, базирующихся на других технологиях.

Признанным фактом является существование огромного разнообразия использования E/E/PE систем в различных областях применений, отличающихся различной степенью сложности, возможными рисками и опасностями. В каждом конкретном применении необходимые меры безопасности будут зависеть от многочисленных факторов, которые являются специфичными для этого применения. Настоящий стандарт, являясь базовым стандартом, позволит формулировать такие меры в будущих международных стандартах для продукции и областей применения, а также в последующих редакциях уже существующих стандартов.

Настоящий стандарт:

- рассматривает все соответствующие стадии жизненных циклов всей системы безопасности, E/E/PE системы безопасности и программного обеспечения системы безопасности (например, от первоначальной концепции, далее проектирование, реализация, эксплуатация, техническое обслуживание вплоть до снятия с эксплуатации), в ходе которых E/E/PE системы используются для выполнения функций безопасности;
- был разработан с учетом быстрого развития технологий; его основа является в значительной мере устойчивой и полной для применения во время будущих разработок;
- делает возможной разработку стандартов для конкретных продукции и областей применения, где используются E/E/PE системы, связанные с безопасностью; разработка стандартов для продукции и областей применения в рамках общей структуры, вводимой настоящим стандартом, должна привести к более высокому уровню согласованности (например, основных принципов, терминологии, и т. д.) как для отдельных областей применения, так и для их совокупностей; что даст преимущества, как для обеспечения безопасности, так и в плане экономики;
- устанавливает метод разработки спецификации требований к безопасности, необходимых для достижения заданной функциональной безопасности E/E/PE систем, связанных с безопасностью;
- применяет для определения требований к уровням полноты безопасности подход, основанный на оценке рисков;

- вводит уровни полноты безопасности при задании целевого уровня полноты безопасности для функций безопасности, которые должны быть реализованы E/E/PE системами, связанными с безопасностью.

Примечание — Настоящий стандарт не устанавливает требования к уровню полноты безопасности для любой функции безопасности, и не определяет, как устанавливается уровень полноты безопасности. Однако настоящий стандарт формирует основанный на риске концептуальный подход и предлагает примеры методов обеспечения функциональной безопасности;

- устанавливает целевые меры отказов для функций безопасности, реализуемых E/E/PE системами, связанными с безопасностью, и связывает эти меры с уровнями полноты безопасности;

- устанавливает нижнюю границу для целевых мер отказов для функции безопасности, реализуемой одиночной E/E/PE-системой, связанной с безопасностью. Для E/E/PE-систем, связанных с безопасностью, работающих в:

- режиме низкой интенсивности запросов на обслуживание, нижняя граница для выполнения функции, для которой система предназначена, устанавливается в соответствии со средней вероятностью опасного отказа по запросу, равной 10^{-5};

- режиме высокой интенсивности запросов на обслуживание или режиме с непрерывным запросом, нижняя граница устанавливается в соответствии с вероятностью 10^{-9} опасных отказов в час.

Примечание 1 — Одиночная E/E/PE-система, связанная с безопасностью, не обязательно предполагает одноканальную архитектуру.

Примечание 2 — В проектах систем, связанных с безопасностью и имеющих низкий уровень сложности, можно достигнуть более низких значений целевой полноты безопасности, но предполагается, что в настоящее время указанные предельные значения целевой полноты безопасности могут быть достигнуты для относительно сложных систем (например, программируемые электронные системы, связанные с безопасностью);

- устанавливает требования к предотвращению и управлению систематическими отказами, основанные на опыте и заключениях из практического опыта. Учитывая, что вероятность возникновения систематических отказов в общем случае не может быть определена количественно, настоящий стандарт позволяет утверждать для специфицируемой функции безопасности, что целевая мера отказов, связанных с этой функцией, может считаться достигнутой, если все требования стандарта были выполнены;

- вводит стойкость к систематическим отказам, применяемую к элементу, характеризующую уверенность в том, что полнота безопасности, касающаяся систематических отказов элемента, соответствует требованиям заданного уровня полноты безопасности;

- применяет широкий диапазон принципов, методов и средств для достижения функциональной безопасности E/E/PE систем, связанных с безопасностью, но не использует явно понятие «безопасный отказ». В то же время, понятия «безопасный отказ» и «безопасный в своей основе» могут быть использованы, но для этого необходимо обеспечить подходящие требования в соответствующих разделах настоящего стандарта, которым эти понятия должны соответствовать.

МЕЖГОСУДАРСТВЕННЫЙ СТАНДАРТ

ФУНКЦИОНАЛЬНАЯ БЕЗОПАСНОСТЬ СИСТЕМ ЭЛЕКТРИЧЕСКИХ, ЭЛЕКТРОННЫХ, ПРОГРАММИРУЕМЫХ ЭЛЕКТРОННЫХ, СВЯЗАННЫХ С БЕЗОПАСНОСТЬЮ

Часть 3

Требования к программному обеспечению

Functional safety of electrical, electronic, programmable electronic safety-related systems.
Part 3. Software requirements

Дата введения — 2019—07—01

1 Область применения

1.1 Настоящий стандарт:

a) применяется совместно с IEC 61508-1 и IEC 61508-2;

b) применяется к любому программному обеспечению, являющемуся частью системы, связанной с безопасностью, либо используемому для разработки системы, связанной с безопасностью, в области применения IEC 61508-1 и IEC 61508-2. Такое программное обеспечение называется программным обеспечением, связанным с безопасностью.

Программное обеспечение, связанное с безопасностью, включает в себя операционные системы, системное программное обеспечение, программы, используемые в коммуникационных сетях, интерфейсы пользователей и обслуживающего персонала, встроенные программно-аппаратные средства, а также прикладные программы;

c) задает конкретные требования к инструментальным средствам поддержки, используемым для разрабатываемой и конфигурируемой системы, связанной с безопасностью, в соответствии с IEC 61508-1 и IEC 61508-2;

d) предусматривает определение функции безопасности и стойкости к систематическим отказам программного обеспечения.

Примечание 1 — То, что уже было сделано как часть спецификации E/E/PE систем, связанных с безопасностью (подраздел 7.2 IEC 61508-2), не следует повторять в настоящем стандарте.

Примечание 2 — Определение функций безопасности и стойкости к систематическим отказам программного обеспечения представляет собой итеративную процедуру; см. рисунки 3 и 6.

Примечание 3 — Структуру документации см. в IEC 61508-1 (пункт 5 и приложение A). Структура документации может учитывать процедуры, используемые в компаниях, а также реальную практику, сложившуюся в отдельных областях применений.

Примечание 4 — Определение термина «стойкость к систематическим отказам» см. 3.5.9 IEC 61508-4;

e) устанавливает требования к стадиям жизненного цикла системы безопасности и действиям, которые должны предприниматься в процессе проектирования и разработки программного обеспечения, связанного с безопасностью (модель жизненного цикла программного обеспечения системы безопасности). Эти требования включают в себя применение мероприятий и методов, ранжированных по уровням требуемой стойкости к систематическим отказам и предназначенных для предотвращения и управления ошибками и отказами в программном обеспечении;

Издание официальное

f) задает требования к информации, относящейся к вопросам подтверждения соответствия аспектов программного обеспечения безопасности системы, которая должна передаваться в организацию, выполняющую интеграцию E/E/PE системы;

g) предоставляет требования к подготовке информации и процедур, касающихся программного обеспечения, которое необходимо пользователям для эксплуатации и технического обслуживания E/E/PE системы, связанной с безопасностью;

h) предоставляет требования, предъявляемые к организациям, выполняющим модификацию программного обеспечения, связанного с безопасностью;

i) предоставляет вместе с IEC 61508-1 и IEC 61508-2 требования к инструментальным средствам поддержки, таким как средства разработки и проектирования, трансляции, тестирования и отладки, управления конфигурацией.

Примечание — Взаимосвязь между IEC 61508-2 и настоящим стандартом показана на рисунке 5;

j) не применяется для медицинского оборудования в соответствии с серией стандартов IEC 60601 [4].

1.2 IEC 61508-1, IEC 61508-2, IEC 61508-3 и IEC 61508-4 являются базовыми стандартами по безопасности, хотя этот статус не применим в контексте E/E/PE систем, связанных с безопасностью, имеющих низкую сложность (пункт 3.4.4 IEC 61508-4). В качестве базовых стандартов по безопасности, они предназначены для использования техническими комитетами при подготовке стандартов в соответствии с принципами, изложенными в руководстве IEC 104 и руководстве ISO/IEC 51. IEC 61508-1, IEC 61508-2, IEC 61508-3 и IEC 61508-4 предназначены для использования в качестве самостоятельных стандартов. Функция безопасности настоящего стандарта не применима к медицинскому оборудованию, соответствующему требованиям серии горизонтальных стандартов IEC 60601 [4].

1.3 В круг обязанностей технического комитета входит использование (там, где это возможно) основополагающих стандартов по безопасности при подготовке собственных стандартов. В этом случае требования, методы проверки или условия проверки настоящего основополагающего стандарта по безопасности не будут применяться, если на них нет конкретной ссылки, или они не включены в стандарты, подготовленные этими техническими комитетами.

1.4 Общая структура IEC 61508-1 — IEC 61508-7 и роль IEC 61508-3 в достижении функциональной безопасности E/E/PE систем, связанных с безопасностью, показана на рисунке 1. Структура жизненного цикла всей системы безопасности показана на рисунке 2.

2 Нормативные ссылки

В настоящем стандарте использованы нормативные ссылки на следующие международные документы и стандарты (для датированных ссылок применяют только указанное издание ссылочного документа):

ISO/IEC Guide 51:1999, Safety aspects — Guidelines for their inclusion in standards (Аспекты безопасности. Руководящие указания по включению в стандарты)

IEC Guide 104:1997, The preparation of safety publications and the use of basic safety publications and group safety publications (Подготовка публикаций по безопасности и использование базовых публикаций по безопасности и публикаций по безопасности групп)

IEC 61508-1:2010, Functional safety of electrical/electronic/programmable electronic safety-related systems — Part 1: General requirements (Функциональная безопасность систем электрических, электронных, программируемых электронных, связанных с безопасностью. Часть 1. Общие требования)

IEC 61508-2:2010, Functional safety of electrical/electronic/programmable electronic safety-related systems — Part 2: Requirements for electrical / electronic / programmable electronic safety-related systems (Функциональная безопасность систем электрических, электронных, программируемых электронных, связанных с безопасностью. Часть 2. Требования к системам электрическим/электронным/программируемым электронным, связанным с безопасностью)

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

3 Термины и определения

В настоящем стандарте применены термины по IEC 61508-4.

4 Соответствие настоящему стандарту

Требования, которые следует выполнять для соответствия настоящему стандарту, приведены в IEC 61508-1 (раздел 4).

5 Документация

Задачи и требования, предъявляемые к документации, приведены в IEC 61508-1 (раздел 5).

6 Дополнительные требования к управлению программным обеспечением, связанным с безопасностью

6.1 Цели

Цели подробно рассмотрены в 6.1 IEC 61508-1.

6.2 Требования

6.2.1 В дополнение к требованиям, описанным в 6.2 IEC 61508-1, предъявляются следующие требования.

6.2.2 Планирование функциональной безопасности должно определять стратегию поставок, разработки, интеграции, верификации, подтверждения соответствия и модификации программного обеспечения в той мере, в какой этого требует уровень полноты безопасности функций, реализуемых E/E/PE системой, связанной с безопасностью.

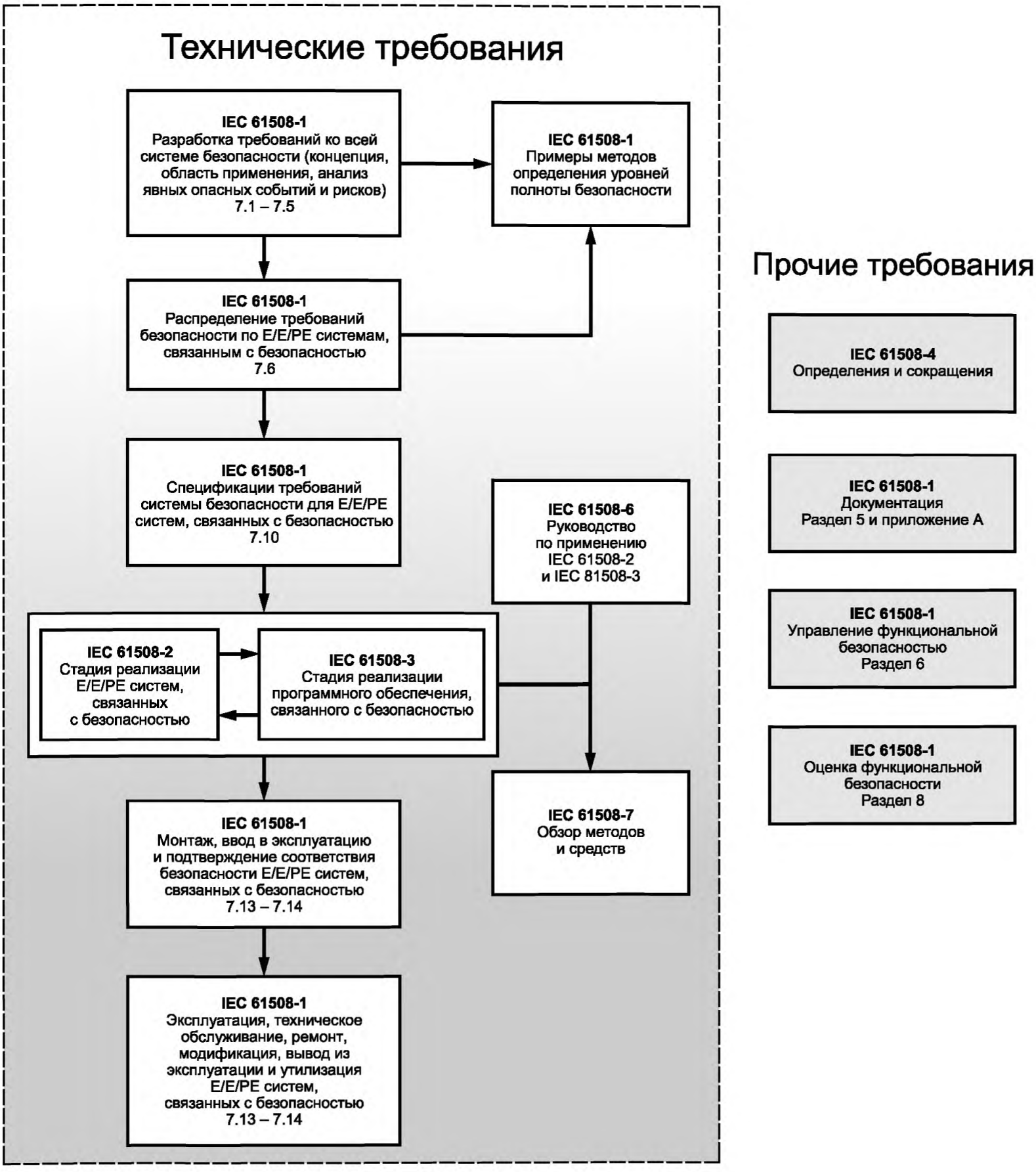

Рисунок 1 — Общая структура стандартов серии IEC 61508

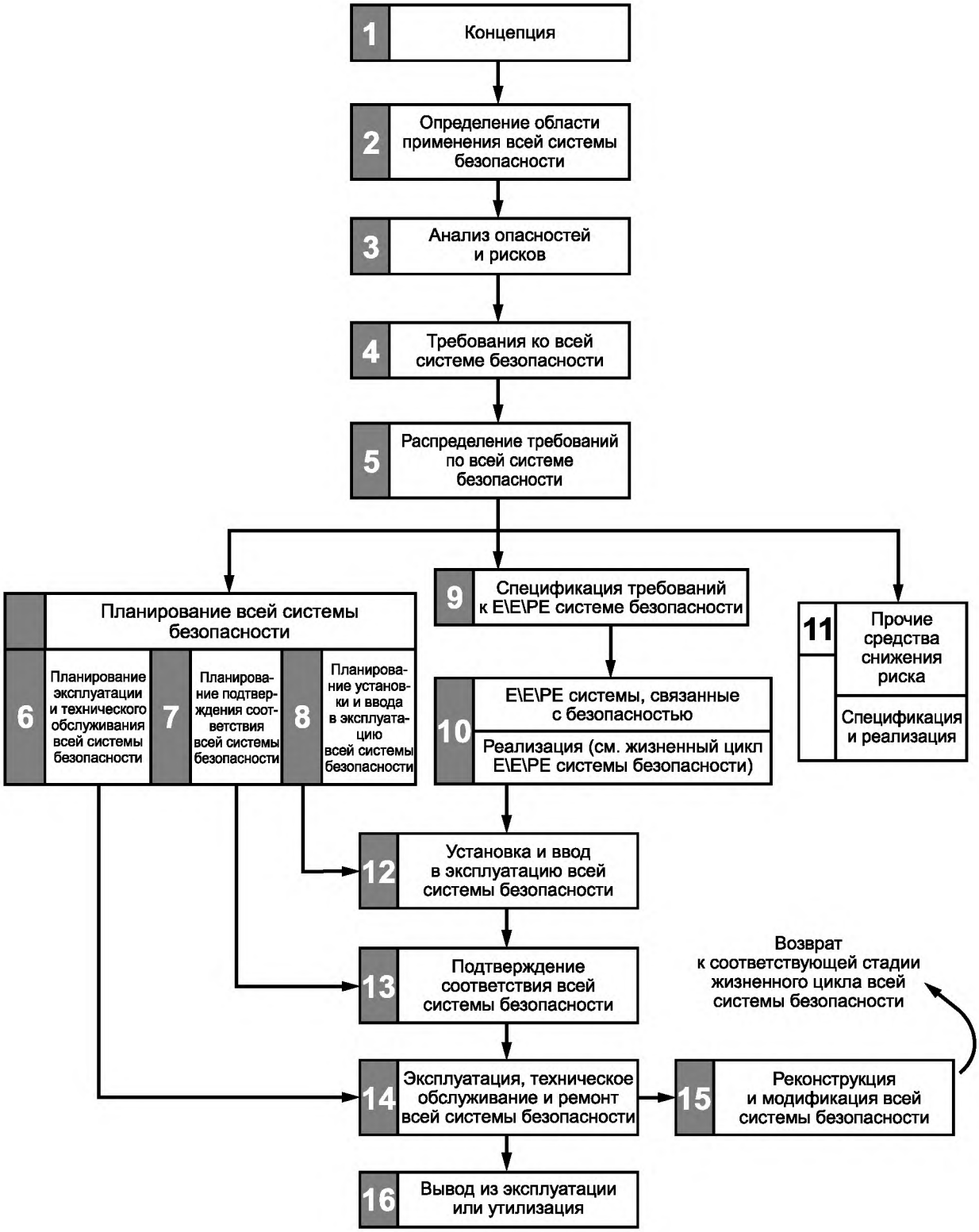

Рисунок 2 — Структура жизненного цикла всей системы безопасности

Примечание — Философия настоящего подхода состоит в использовании планирования функциональной безопасности в качестве возможности для приспособления настоящего стандарта для учета требуемой полноты безопасности для каждой функции безопасности, реализуемой E/E/PE системой, связанной с безопасностью.

6.2.3 Система управления конфигурацией программного обеспечения должна:

a) использовать административные и технические средства контроля на протяжении жизненного цикла программного обеспечения системы безопасности для того, чтобы управлять изменениями в программах и таким образом гарантировать непрерывное выполнение указанных в спецификациях требований к программному обеспечению, связанному с безопасностью;

b) гарантировать выполнение операций, необходимых для того, чтобы продемонстрировать достижение заданной стойкости к систематическим отказам программного обеспечения;

c) осуществлять аккуратную поддержку с использованием уникальной идентификации всех элементов конфигурации, которые необходимы для обеспечения требований полноты безопасности E/E/PE системы, связанной с безопасностью. Элементы конфигурации должны включать в себя, как минимум, следующее: анализ системы безопасности и требования к системе безопасности; спецификацию программного обеспечения и проектную документацию; исходный текст программ; план и результаты тестирования; документацию о проверках; ранее разработанные программные элементы и пакеты, которые должны быть включены в E/E/PE систему, связанную с безопасностью; все инструментальные средства и системы разработки, которые использовались при создании, тестировании или выполнении иных действий с программным обеспечением E/E/PE системы, связанной с безопасностью;

d) использовать процедуры контроля над внесением изменений для того, чтобы:

- предотвращать несанкционированные модификации,
- документально оформлять запросы на выполнение модификаций,
- анализировать влияние предлагаемых модификаций и утверждать либо отвергать модификации,
- подробно документально оформлять модификации и выдавать полномочия на выполнение всех утвержденных модификаций,
- устанавливать основные параметры конфигурации системы для этапов разработки программного обеспечения и документально оформлять (частичное) тестирование интеграции системы,
- гарантировать объединение и встраивание всех подсистем программного обеспечения (включая переработку более ранних версий).

Примечание 1 — Для осуществления руководства и применения административных и технических средств контроля необходимы наличие полномочий и принятие управленческих решений.

Примечание 2 — С одной стороны, анализ влияния может включать неформальную оценку. С другой стороны, анализ влияния может включать в себя строгий формальный анализ возможного неблагоприятного воздействия всех предложенных изменений, которые могут быть неверно поняты или неверно осуществлены. Руководство по анализу влияния см. [5];

e) гарантировать, что осуществлены соответствующие меры, чтобы корректно загружать прошедшие подтверждение соответствия элементы программного обеспечения и данные в систему во время ее выполнения.

Примечание — Допускается рассматривать отдельные целевые системы, а также общие системы. Программному обеспечению, кроме приложений, возможно, понадобился бы безопасный метод загрузки, например, для встроенных программ программируемых устройств;

f) документально оформлять перечисленную ниже информацию, для того чтобы обеспечить возможность последующего аудита функциональной безопасности: состояние конфигурации, текущее состояние системы, обоснование (с учетом результатов анализа влияния) и утверждение всех модификаций, подробное описание всех модификаций;

h) строго документально оформлять каждую версию программного обеспечения, связанного с безопасностью. Обеспечить хранение всех версий программного обеспечения и всей относящейся к ним документации, а также версий данных для обеспечения возможности сопровождения и выполнения модификаций на протяжении всего периода использования разработанного программного продукта.

Примечание — Дополнительную информацию по управлению конфигурацией см. [5].

7 Требования к жизненному циклу программного обеспечения системы безопасности

7.1 Общие положения

7.1.1 Цели

Целью требований, излагаемых в настоящем подразделе, является разделение процесса разработки программного обеспечения на этапы и процессы (см. таблицу 1 и рисунки 3—6).

7.1.2 Требования

7.1.2.1 В соответствии с IEC 61508-1 (раздел 6) при планировании системы безопасности должен быть выбран и специфицирован жизненный цикл для разработки программного обеспечения системы безопасности.

7.1.2.2 Может использоваться любая модель жизненного цикла программного обеспечения системы безопасности при условии, что она соответствует всем целям и требованиям настоящего раздела.

7.1.2.3 Каждая стадия жизненного цикла программного обеспечения системы безопасности должна быть разделена на элементарные процессы. Для каждой стадии должны быть определены область применения, входные данные и выходные данные.

Примечание — См. рисунки 3, 4 и таблицу 1.

7.1.2.4 При условии, что жизненный цикл программного обеспечения системы безопасности соответствует требованиям таблицы 1, приемлемо адаптировать V-модель (см. рисунок 6) с учетом полноты безопасности и сложности проекта.

Примечание 1 — Модель жизненного цикла программного обеспечения системы безопасности, которая соответствует требованиям данного раздела, может быть соответственно настроена для конкретных потребностей проекта или организации. Полный список стадий жизненного цикла, приведенный в таблице 1, относится к большим заново разрабатываемым системам. Для небольших систем может оказаться целесообразным, например, объединить стадии проектирования системы программного обеспечения и проектирования архитектуры.

Примечание 2 — Характеристики систем, управляемых данными, описание которых можно найти в приложении G (например, языки программирования с полной или ограниченной изменчивостью, степень конфигурации данных), могут использоваться для настройки жизненного цикла программного обеспечения системы безопасности.

7.1.2.5 Любая настройка жизненного цикла программного обеспечения системы безопасности должна быть обоснована функциональной безопасностью.

7.1.2.6 Процедуры обеспечения качества и безопасности должны быть интегрированы в процессы жизненного цикла систем безопасности.

7.1.2.7 Для каждой стадии жизненного цикла следует использовать соответствующие методы и мероприятия. В приложениях A и B приведены рекомендации по выбору методов и средств, а также ссылки на [5] и [6]. В [5] и [6] приведены рекомендации по выбору конкретного метода для обеспечения свойств, требуемых для систематической полноты безопасности. Методы, выбранные в соответствии с этими рекомендациями, сами по себе не гарантируют достижения необходимой полноты безопасности.

Примечание — Успех в достижении систематической полноты безопасности зависит от выбора методов с учетом следующих факторов:

- согласованность и взаимодополняющий характер выбранных методов, языков и инструментов для всего цикла разработки;
- полностью ли понимают разработчики, используемые ими методы, языки и инструментальные средства;
- насколько соответствуют методы, языки и инструментальные средства конкретным проблемам, с которыми сталкиваются разработчики в процессе разработки.

7.1.2.8 Результаты процессов жизненного цикла программного обеспечения системы безопасности должны быть документально оформлены (см. раздел 5).

Примечание — IEC 61508-1 (раздел 5) рассматривает документальное оформление результатов стадий жизненного цикла системы безопасности. При разработке некоторых E/E/PE систем, связанных с безопасностью, результаты некоторых стадий жизненного цикла системы безопасности могут быть оформлены в виде отдельных документов, тогда как результаты от других стадий могут объединяться в один документ. Существенным является требование, чтобы результаты стадии жизненного цикла системы безопасности соответствовали ее предназначению.

7.1.2.9 Если на какой-либо стадии жизненного цикла программного обеспечения системы безопасности возникает необходимость внести изменение, относящееся к более ранней стадии жизненного цикла, то, во-первых, используя анализ влияния, необходимо определить, какие модули программного обеспечения затрагиваются изменениями, и во-вторых, какие действия на более ранней стадии жизненного цикла системы безопасности должны быть выполнены повторно.

Примечание — С одной стороны, анализ влияния может представлять собой неформальную оценку. С другой стороны, он может включать в себя строгий формальный анализ возможных неблагоприятных последствий всех предполагаемых изменений, которые, возможно, были плохо продуманы или неверно реализованы. Руководящие указания по анализу влияния см. [5].

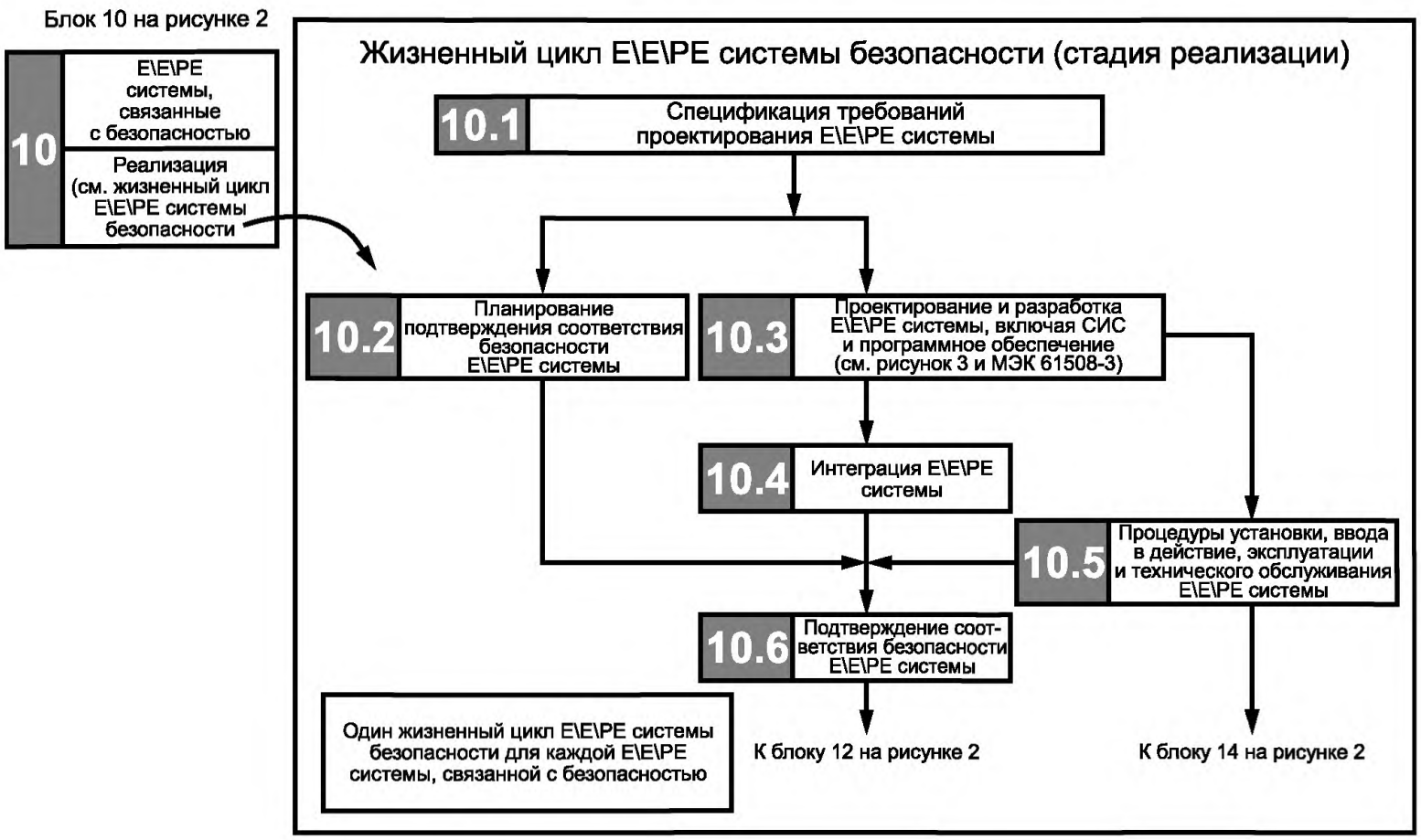

Рисунок 3 — Структура жизненного цикла E/E/PE системы безопасности (стадия реализации)

Рисунок 4 — Структура жизненного цикла программного обеспечения системы безопасности (стадия реализации)

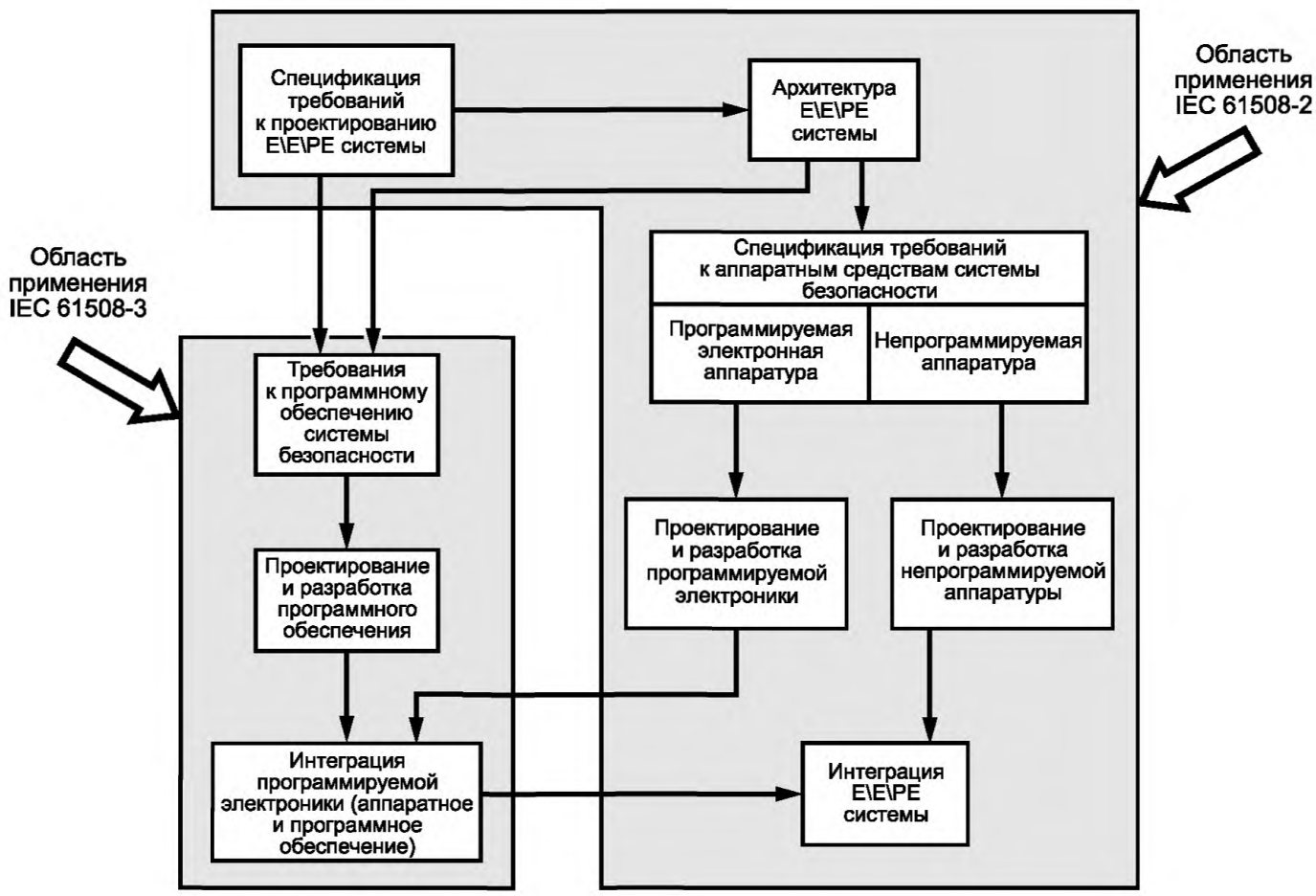

Рисунок 5 — Взаимосвязь и области применения IEC 61508-2 и IEC 61508-3

Рисунок 6 — Стойкость к систематическим отказам и жизненный цикл разработки программного обеспечения (V-модель)

Таблица 1 — Жизненный цикл программного обеспечения (ПО) системы безопасности: обзор

Стадия жизненного цикла ПО системы безопасности (номер стадии соответствует номеру блока на рисунке 4)	Задача	Область применения	Номер пункта или раздела	Входные данные	Выходные данные
10.1 Спецификация требований к ПО системы безопасности	Указать требования: к системе программируемой электроники (ПЭ); к ПО, связанному с безопасностью, в виде требований к функциям безопасности ПО и стойкости к систематическим отказам ПО. Указать необходимые для реализации требуемых функций безопасности требования к функциям безопасности ПО для каждой E/E/PE системы, связанной с безопасностью. Указать требования стойкости к систематическим отказам ПО для каждой E/E/PE системы, связанной с безопасностью, необходимые для достижения уровня полноты безопасности, указанного для каждой функции безопасности, назначенной этой E/E/PE системе, связанной с безопасностью	ПЭ система. Система ПО	7.2.2	Спецификация требований к E/E/PE системе безопасности в результате распределения (см. IEC 61508-1) Спецификация требований к E/E/PE системе безопасности (из IEC 61508-2)	Спецификация требований к ПО системы безопасности
10.2 Планирование подтверждения соответствия безопасности системы для аспектов ПО	Разработать план подтверждения соответствия безопасности системы для аспектов ПО	ПЭ система. Система ПО	7.3.2	Спецификация требований к ПО системы безопасности	План подтверждения соответствия безопасности системы для аспектов ПО
	Архитектура: разработать архитектуру ПО, которая соответствует указанным требованиям к ПО, связанному с безопасностью, для необходимого уровня полноты безопасности; оценить требования, предъявляемые к ПО со стороны архитектуры аппаратных средств E/E/PE системы, связанной с безопасностью, включая значение взаимодействия между программным обеспечением и аппаратными средствами E/E/PE системы, для безопасности управляемого оборудования	ПЭ система. Система ПО	7.4.3	Спецификация требований к ПО системы безопасности. Проект архитектуры аппаратных средств E/E/PE системы (из IEC 61508-2)	Описание проекта архитектуры ПО. Спецификация проверки интеграции архитектуры ПО. Спецификация проверки интеграции ПО / ПЭ (как требует IEC 61508-2)

Продолжение таблицы 1

Стадия жизненного цикла ПО системы безопасности (номер стадии соответствует номеру блока на рисунке 4)	Задача	Область применения	Номер пункта или раздела	Входные данные	Выходные данные
	Инструментальные средства поддержки и языки программирования: выбрать подходящий набор инструментальных средств, включая языки программирования и компиляторы, системные интерфейсы времени выполнения, пользовательские интерфейсы и форматы и представления данных для требуемого уровня полноты безопасности в течение всего жизненного цикла ПО системы безопасности для верификации, подтверждения соответствия, оценки и модификации	ПЭ система. Система ПО. Инструментальные средства поддержки. Язык программирования	7.4.4	Спецификация требований к ПО системы безопасности. Описание проекта архитектуры ПО	Инструментальные средства поддержки и стандарты кодирования. Выбор инструментов разработки
	Детальное проектирование и разработка (проект программной системы): спроектировать и реализовать ПО, которое соответствует указанным требованиям к ПО системы безопасности для необходимого уровня полноты безопасности; ПО должно быть пригодным к анализу и верификации и поддерживать возможность безопасной модификации	Проектирование архитектуры основных компонентов и подсистем ПО	7.4.5	Описание проекта архитектуры ПО. Инструментальные средства поддержки и стандарты кодирования	Спецификация проекта системы ПО. Спецификация тестирования интеграции системы ПО
	Детальное проектирование и разработка (проект отдельных программных модулей): спроектировать и реализовать ПО, которое соответствует указанным требованиям к ПО системы безопасности для необходимого уровня полноты безопасности; ПО должно быть пригодным к анализу и верификации и поддерживать возможность безопасной модификации	Проект системы ПО	7.4.5	Спецификация проекта системы ПО. Инструментальные средства поддержки и стандарты кодирования	Спецификация проекта программного модуля. Спецификация тестирования программного модуля
	Детальная реализация исходного текста: - спроектировать и реализовать ПО, которое соответствует указанным требованиям к ПО системы безопасности для необходимого уровня полноты безопасности; ПО должно быть пригодным к анализу и верификации и поддерживать возможность безопасной модификации	Отдельные программные модули	7.4.6	Спецификация проекта программного модуля. Инструментальные средства поддержки и стандарты кодирования	Листинг исходного текста. Обзорный отчет по исходному тексту

Продолжение таблицы 1

Стадия жизненного цикла ПО системы безопасности (номер стадии соответствует номеру блока на рисунке 4)	Задача	Область применения	Номер пункта или раздела	Входные данные	Выходные данные
	Тестирование программных модулей: - верифицировать выполнение требований к ПО, связанному с безопасностью (в терминах требуемых функций безопасности и стойкости к систематическим отказам ПО); - показать, что каждый программный модуль выполняет предназначенные для него функции и не выполняет непредусмотренных действий; - гарантировать в той мере, насколько это уместно, что конфигурация данных ПЭ систем выполняет указанные требования стойкости к систематическим отказам ПО	Прграммные модули	7.4.7	Спецификация тестирования программного модуля. Листинг исходного текста. Обзорный отчет по исходному тексту	Результаты тестирования программного модуля. Верифицированные и проверенные программные модули
10.3 Проектирование и разработка ПО	Проверка интеграции ПО: - верифицировать выполнение требований к ПО системы безопасности (в терминах требуемых функций безопасности и стойкости к систематическим отказам ПО); - показать, что каждый программный модуль выполняет предназначенные для него функции и не выполняет непредусмотренных действий; - гарантировать в той мере, насколько это уместно, что конфигурация данных ПЭ систем выполняет указанные требования стойкости к систематическим отказам ПО	Архитектура ПО. Система ПО	7.4.8	Спецификация тестирования интеграции программной системы	Результаты тестирования интеграции программной системы. Верифицированная и проверенная программная система
10.4 Интеграция программируемой электроники (аппаратные средства и программное обеспечение)	Интегрировать ПО с выбранной программируемой электронной аппаратурой. Объединить ПО и аппаратные средства в связанных с безопасностью программируемых электронных устройствах для того, чтобы гарантировать их совместимость и выполнение требований к необходимому уровню полноты безопасности	Аппаратное обеспечение прграммируемой электроники. Интегрированное ПО	7.5.2	Спецификация тестирования интеграции архитектуры ПО. Спецификация тестирования интеграции программумой электроники (IEC 61508-2). Интегрированная программумая электроника	Результаты тестирования интеграции архитектуры ПО. Результаты тестирования интеграции программируемой электроники. Верифицированная и проверенная интегрированная программируемая электроника

Продолжение таблицы 1

Стадия жизненного цикла ПО системы безопасности (номер стадии соответствует номеру блока на рисунке 4)	Задача	Область применения	Номер пункта или раздела	Входные данные	Выходные данные
10.5 Процедуры эксплуатации и модификации ПО	Предоставить информацию и процедуры, относящиеся к ПО и необходимые для того, чтобы гарантировать соблюдение функциональной безопасности E/E/PE систем, связанных с безопасностью, во время эксплуатации и модификации	Аппаратное обеспечение программируемой электроники. Интегрированное ПО	7.6.2	По необходимости вся информация, описанная выше	Процедуры эксплуатации и модификации ПО
10.6 Подтверждение соответствия безопасности системы аспектов ПО	Обеспечить гарантию, что интегрированная система соответствует указанным требованиям к ПО, связанному с безопасностью, для заданного уровня полноты безопасности	Аппаратное обеспечение программируемой электроники. Интегрированное ПО	7.7.2	План подтверждения соответствия безопасности системы аспектов ПО	Результаты подтверждения соответствия ПО безопасности системы. Принятое ПО
Модификация ПО	Внести поправки, улучшения или модификации в принятое ПО, гарантируя, что требуемый уровень стойкости к систематическим отказам ПО будет сохранен	Аппаратное обеспечение программируемой электроники. Интегрированное ПО	7.8.2	Процедуры модификации ПО. Результаты модификации ПО	Результаты анализа влияния модификации ПО. Журнал модификации ПО
Верификация ПО	Протестировать и оценить выходные данные для заданной стадии жизненного цикла ПО системы безопасности для того, чтобы гарантировать правильность и соответствие выходным данным и стандартам для этой стадии	Зависит от стадии	7.9.2	Соответствующий план верификации (зависит от стадии)	Соответствующий отчет по верификации (зависит от стадии)
Оценка функциональной безопасности ПО	Изучить и принять решение по функциональной безопасности аспектов ПО, которая обеспечивается E/E/PE системами, связанными с безопасностью	Для всех стадий, перечисленных выше	8	План оценки функциональной безопасности ПО	Отчет по оценке функциональной безопасности ПО

7.2 Спецификация требований к программному обеспечению системы безопасности

Примечание — Данная стадия представлена на рисунке 4 (см. 10.1).

7.2.1 Цели

7.2.1.1 Первой целью настоящего подраздела является определение требований к программному обеспечению, связанному с безопасностью, как требований к функциям безопасности программного обеспечения и требований стойкости к систематическим отказам программного обеспечения.

7.2.1.2 Второй целью настоящего подраздела является определение требований к функциям безопасности программного обеспечения каждой E/E/PE системы, связанной с безопасностью, которые нужны для реализации этих функций безопасности.

7.2.1.3 Третьей целью настоящего подраздела является определение требований стойкости к систематическим отказам программного обеспечения для каждой связанной с безопасностью E/E/PE системы, необходимых для достижения уровня полноты безопасности, назначенного каждой функции безопасности, реализуемой E/E/PE системой, связанной с безопасностью.

7.2.2 Требования

Примечание 1 — В большинстве случаев эти требования выполняются комбинацией базового встраиваемого программного обеспечения и программных модулей, которые разработаны специально для конкретного применения. Именно комбинация этих двух видов программного обеспечения позволяет достигать характеристик, описанных в подразделах, приводимых ниже. Точная граница между базовым и прикладным программным обеспечением зависит от выбранной архитектуры программной системы (см. 7.4.3).

Примечание 2 — Для выбора соответствующих методов и средств (см. приложения A и B) для осуществления требования данного пункта, необходимо рассмотреть следующие свойства (см. приложение C, где даны указания по интерпретации свойств, и приложение F IEC 61508-7, где даны их неформальные определения) спецификации требований к программному обеспечению системы безопасности:

- полнота охвата потребностей безопасности программным обеспечением;
- корректность охвата потребностей безопасности программным обеспечением;
- отсутствие ошибок в самой спецификации, включая отсутствие неоднозначности;
- ясность требований к системе безопасности;
- отсутствие неблагоприятного взаимовлияния функций, не связанных с безопасностью, и функций безопасности, реализуемых программным обеспечением системы безопасности;
- способность обеспечения проведения оценки и подтверждения соответствия.

Примечание 3 — Потребности безопасности, которым должно соотвествовать программное обеспечение, описываются набором функций безопасности и соответствующими требованиями полноты безопасности, определенных для функций программного обеспечения в проекте E/E/PE системы. (Полный набор требований к системе безопасности — гораздо шире, так как включает также функции безопасности, которые выполняются не программным обеспечением, а другими средствами). Полнота спецификации требований к программному обеспечению системы безопасности решающим образом зависит от эффективности более ранних стадий жизненного цикла системы.

7.2.2.1 Если требования к программному обеспечению, связанному с безопасностью, уже были определены в требованиях к E/E/PE системе, связанной с безопасностью (см. 7.2 IEC 61508-2), повторять их не требуется.

7.2.2.2 Спецификация требований к программному обеспечению, связанному с безопасностью, должна быть выработана на основе заданных требований к безопасности E/E/PE системы, связанной с безопасностью (IEC 61508-2), и любых требований к планированию безопасности (см. раздел 6). Эта информация должна быть доступна для разработчика программного обеспечения.

Примечание 1 — Это требование означает, что должно быть тесное взаимодействие между разработчиком E/E/PE системы и разработчиком программного обеспечения (IEC 61508-2 и IEC 61508-3) По мере того как требования к безопасности и архитектура программного обеспечения (см. 7.4.3) становятся более определенными, может проявляться влияние на архитектуру аппаратных средств E/E/PE системы, и по этой причине становится важным тесное взаимодействие между разработчиками аппаратных средств и программного обеспечения (см. рисунок 5).

Примечание 2 — Проект программного обеспечения может включать уже существующее и многократно использованное программное обеспечение. Такое программное обеспечение может быть разработано без учета спецификации требования к создаваемой системе. В 7.4.2.12 представлены требования к уже существующему программному обеспечению, чтобы удовлетворить спецификации требований к программному обеспечению системы безопасности.

7.2.2.3 Спецификация требований к программному обеспечению, связанному с безопасностью, должна быть достаточно подробной для того, чтобы обеспечить стадии проектирования и внедрения информации для реализации требуемой полноты безопасности (включая требования к независимости, см. IEC 61508-2) и позволить выполнить оценку функциональной безопасности.

Примечание — Уровень детальности спецификации может изменяться в зависимости от сложности применения. Соответствующая спецификация функционального поведения может включать требования к точности, синхронизации и производительности, емкости, устойчивости, допустимой перегрузки и другие свойства, характеризующие конкретное применение.

7.2.2.4 Для решения проблемы независимости должен быть выполнен подходящий анализ отказов по общей причине. Если выявлены вероятные механизмы отказа, то должны быть приняты эффективные меры защиты.

Примечание — В приложении F приведены методы достижения одного аспекта независимости программного обеспечения.

7.2.2.5 Разработчик программного обеспечения должен просмотреть информацию, содержащуюся в 7.2.2.2 для того, чтобы гарантировать, что требования определены адекватным образом. В частности, разработчик программного обеспечения должен учесть:

a) функции безопасности;

b) конфигурацию или архитектуру системы;

c) требования к полноте безопасности аппаратных средств (программируемой электроники, датчиков и исполнительных устройств);

d) требования стойкости к систематическим отказам программного обеспечения;

e) производительность и время отклика;

f) интерфейсы оборудования и оператора, включая разумно предвидимые нарушения.

Примечание — Необходимо рассмотреть совместимость с любыми уже существующими применениями.

7.2.2.6 В специфицированных требованиях к программному обеспечению, связанному с безопасностью, должны быть подробно описаны все соответствующие режимы работы УО, E/E/PE системы и любых оборудования или системы, подсоединенных к E/E/PE системе, если только они не были уже адекватно определены в требованиях к системе безопасности, специфицированных для E/E/PE системы, связанной с безопасностью.

7.2.2.7 В спецификации требований к программному обеспечению системы безопасности должны быть определены и документально оформлены все, связанные с безопасностью, и иные необходимые ограничения, связанные с взаимодействием между аппаратными средствами и программным обеспечением.

7.2.2.8 В той степени, в которой этого требует описание проекта архитектуры аппаратных средств E/E/PE систем, и с учетом возможного увеличения сложности спецификация требований к программному обеспечению системы безопасности должна учитывать:

a) самоконтроль программного обеспечения (например, IEC 61508-7);

b) мониторинг программируемой электронной аппаратуры, датчиков и исполнительных устройств;

c) периодическое тестирование функций безопасности во время выполнения программы;

d) предоставление возможности тестирования функций безопасности во время работы УО;

e) функции программного обеспечения для выполнения контрольных испытаний и всех диагностических тестов, чтобы выполнить требование полноты безопасности E/E/PE системы, связанной с безопасностью.

Примечание — Увеличение сложности, которое может возникнуть из вышеупомянутых соображений, может потребовать пересмотра архитектуры.

7.2.2.9 Если E/E/PE система, связанная с безопасностью, должна выполнять функции, не относящиеся к безопасности, эти функции должны быть четко указаны в спецификации требований к программному обеспечению системы безопасности.

Примечание — Требования к отсутствию взаимовлияния функций, связанных и несвязанных с безопасностью, см. 7.4.2.8 и 7.4.2.9.

7.2.2.10 Спецификация требований к программному обеспечению системы безопасности должна выражать необходимые характеристики безопасности продукции, а не его проекта, как это определяется при планировании системы безопасности (см. раздел 6 IEC 61508-1). С учетом 7.2.2.1—7.2.2.9 в зависимости от конкретных обстоятельств должны быть определены следующие положения:

a) требования к функциям программного обеспечения системы безопасности:

1) функции, которые позволяют УО достигать или поддерживать безопасное состояние,

2) функции, связанные с обнаружением, оповещением и обработкой ошибок аппаратных средств программируемой электроники,

3) функции, связанные с обнаружением, оповещением и обработкой ошибок датчиков и исполнительных устройств,

4) функции, связанные с обнаружением, оповещением и обработкой ошибок в самом программном обеспечении (самоконтроль программного обеспечения),

5) функции, связанные с периодическим тестированием функций безопасности в режиме реального времени (в предопределенной операционной среде),

6) функции, связанные с периодическим тестированием функций безопасности в автономном режиме (то есть в условиях, в которых функция безопасности УО не выполняется),

7) функции, обеспечивающие модификацию ПЭ системы безопасности,

8) интерфейсы функций, не связанных с безопасностью,

9) производительность и время отклика,

10) интерфейсы между программным обеспечением и ПЭ системой.

Примечание — Интерфейсы должны включать в себя средства программирования в автономном и неавтономном режимах,

11) средства коммуникации, связанные с безопасностью;

b) требования стойкости к систематическим отказам программного обеспечения:

1) уровень(и) полноты безопасности для каждой функции по перечислению a).

Примечание — Назначение полноты безопасности элементам программного обеспечения описано в приложении A IEC 61508-5,

2) требования независимости между функциями.

7.2.2.11 Если требования к программному обеспечению системы безопасности выражены или выполнены в виде конфигурации данных, то эти данные должны быть:

a) согласованы с требованиями к системе безопасности;

b) выражены значениями из допустимого диапазона и санкционированными комбинациями реализующих их параметров;

c) определены способом, который совместим с базовым программным обеспечением (например, последовательность выполнения, время выполнения, структуры данных, и т. д.).

Примечание 1 — Это требование к прикладным данным относится, в частности, к применениям, управляемым данными. Они характеризуются следующим образом: исходный код уже существует, а главная цель разработки состоит в обеспечении уверенности, что конфигурации данных правильно задает поведение, требуемое от применения. Между элементами данных могут быть сложные зависимости, и достоверность данных может меняться с течением времени.

Примечание 2 — Указания по системам, управляемым данными, см. в приложении G.

7.2.2.12 Если данные определяют интерфейс между программным обеспечением и внешними системами, то в дополнение к п. 7.4.11 IEC 61508-2 необходимо рассмотреть следующие характеристики:

a) необходимость согласованности при определении данных;

b) недостоверные, некорректные или неактуальные значения;

c) время отклика и пропускная способность, в том числе в условиях максимальной загрузки;

d) максимально и минимально возможное время выполнения и зависание;

e) переполнение и потеря данных в памяти.

7.2.2.13 Параметры эксплуатации должны быть защищены от:

а) недостоверных, находящихся вне определенного диапазона или несвоевременных, значений;

б) несанкционированных изменений;

в) искажений.

Примечание 1 — Следует рассмотреть защиту от несанкционированных изменений как программных, так и непрограммных средств. Необходимо отметить, что эффективная защита от несанкционированных изменений программного обеспечения может отрицательно отразиться на безопасности, например, если изменения необходимо выполнить быстро и в напряженных условиях.

Примечание 2 — Хотя человек может быть частью системы, связанной с безопасностью (см. раздел 1 IEC 61508-1), требования человеческого фактора, связанные с проектированием E/E/PE систем, связанных с без-

опасностью, в настоящем стандарте подробно не рассматриваются. Однако там, где это необходимо, должны быть рассмотрены следующие соображения, связанные с человеком:

- информационная система оператора должна использовать пиктографическое представление и терминологию, с которой знакомы операторы. Они должны быть четкими, понятными и лишенными ненужных деталей и/или аспектов;
- информация об УО, выведенная оператору на экран, должна подробно и достоверно отображать физическое состояние УО;
- если на экране дисплея оператору представлена информация о выполняющихся в УО процессах и/или если оператор выполняет возможные действия, последствия которых нельзя сразу заметить, то выведенная на экран в автоматическом режиме информация должна показать состояние, в котором находится представленная на дисплее система, или последовательность действий, в которой указано, какое состояние последовательности достигнуто, какие операции могут быть выполнены и какие могут быть возможные последствия.

7.3 Планирование подтверждения соответствия безопасности системы для аспектов программного обеспечения

Примечание 1 — Эта стадия представлена на рисунке 4 (см. блок 10.2).

Примечание 2 — Подтверждение соответствия для программного обеспечения обычно не может быть выполнено отдельно от используемого им оборудования и системной среды.

7.3.1 Цель

Целью требований настоящего подраздела является разработка плана подтверждения соответствия безопасности системы для аспектов программного обеспечения, связанного с безопасностью.

7.3.2 Требования

7.3.2.1 В ходе планирования должны быть определены процедурные и технические шаги, которые необходимо выполнить для того, чтобы продемонстрировать, что программное обеспечение соответствует требованиям безопасности.

7.3.2.2 План подтверждения соответствия безопасности системы для аспектов программного обеспечения должен содержать следующие положения:

a) точная дата, когда должно происходить подтверждение соответствия;

b) перечень лиц, осуществляющих подтверждение соответствия;

c) идентификацию соответствующих режимов работы УО, включая:

- подготовку к использованию, а также установку и настройку,
- работу в режиме запуска и обучения, в автоматическом, ручном, полуавтоматическом и стационарном режимах,
- переустановку, выключение, сопровождение,
- предполагаемые ненормальные условия и предполагаемые ошибки оператора;

d) идентификация программного обеспечения, связанная с безопасностью, для которого должна быть проведена процедура подтверждения соответствия, для каждого режима работы УО до момента его ввода в эксплуатацию;

e) техническая стратегия для подтверждения соответствия (например, аналитические методы, статистическое тестирование и т. п.);

f) средства (методы) и процедуры в соответствии с перечислением e), которые должны быть использованы для того, чтобы подтвердить, что каждая функция безопасности соответствует установленным требованиям к функциям безопасности и требованиям стойкости к систематическим отказам программного обеспечения;

g) условия, в которых должны происходить процедуры подтверждения соответствия (например, при тестировании может потребоваться использование калиброванных инструментов и оборудования);

h) критерии прохождения/непрохождения подтверждения соответствия;

i) политика и процедуры, используемые для оценки результатов подтверждения соответствия, в частности, при оценке отказов.

Примечание — Эти требования основаны на общих требованиях подраздела 7.8 IEC 61508-1.

7.3.2.3 Подтверждение соответствия должно дать обоснование выбранной стратегии. Техническая стратегия для подтверждения соответствия программного обеспечения, связанного с безопасностью, должна содержать следующую информацию:

a) выбор ручных или автоматических методов, или и тех и других;

b) выбор статических или динамических методов, или и тех и других;

c) выбор аналитических или статистических методов, или и тех и других;

d) выбор критериев приемки на основе объективных факторов или экспертной оценки, или и тех и другой.

7.3.2.4 В рамках процедуры подтверждения соответствия аспектов программного обеспечения, связанного с безопасностью, если этого требует уровень полноты безопасности (раздел 8 IEC 61508-1), область применения и содержание плана подтверждения соответствия безопасности системы аспектов программного обеспечения должны быть изучены экспертом или третьей стороной, представляющей эксперта. Эта процедура должна также включать в себя заявление о присутствии эксперта при испытаниях.

7.3.2.5 Критерии прохождения/непрохождения при завершении подтверждения соответствия программного обеспечения должны включать в себя:

a) необходимые входные сигналы, включая их последовательность и значения;

b) предполагаемые выходные сигналы, включая их последовательность и значения, и

c) другие критерии приемки, например, использование памяти, синхронизацию, допустимые интервалы значений.

7.4 Проектирование и разработка программного обеспечения

Примечание — Эта стадия представлена на рисунке 4 (см. 10.3).

7.4.1 Цели

7.4.1.1 Первой целью требований настоящего подраздела является создание такой архитектуры программного обеспечения, которая соответствовала бы заданным требованиям к программному обеспечению, связанному с безопасностью, в отношении необходимого уровня полноты безопасности.

7.4.1.2 Второй целью требований настоящего подраздела является оценка требований, предъявляемых к программному обеспечению со стороны архитектуры аппаратных средств E/E/PE системы, связанной с безопасностью, включая значение взаимодействия между аппаратными средствами и программным обеспечением E/E/PE систем для обеспечения безопасности управляемого оборудования.

7.4.1.3 Третьей целью требований настоящего подраздела является выбор подходящего набора инструментальных средств, включая языки программирования и компиляторы, интерфейсы системы времени выполнения, интерфейсы пользователя и форматы и представления данных, который соответствовал бы заданному уровню полноты безопасности на протяжении всего жизненного цикла программного обеспечения системы безопасности и способствовал бы выполнению процессов верификации, подтверждения соответствия, оценки и модификации.

7.4.1.4 Четвертой целью требований настоящего подраздела является проектирование и реализация программного обеспечения, которое соответствовало бы специфицированным требованиям к программному обеспечению, связанному с безопасностью, для необходимого уровня полноты безопасности. Это программное обеспечение должно быть пригодным для анализа и верификации и обладать способностью к безопасной модификации.

7.4.1.5 Пятой целью требований настоящего подраздела является проверка выполнения требований к программному обеспечению, связанному с безопасностью (в отношении необходимых функций безопасности и стойкости к систематическим отказам программного обеспечения).

7.4.1.6 Шестой целью требований настоящего подраздела является гарантирование, в той мере, насколько это уместно, того, что конфигурирование данными ПЭ систем соответствует указанным в настоящем подразделе требованиям стойкости к систематическим отказам программного обеспечения.

7.4.2 Общие требования

7.4.2.1 В зависимости от природы процесса разработки программного обеспечения ответственными за соответствие требованиям 7.4 могут быть: или только поставщик связанного с безопасностью программного окружения (например, поставщик PLS), или только пользователь этого окружения (например, разработчик прикладных программ), или поставщик и пользователь. Распределение ответственности должно быть определено во время планирования системы безопасности (см. раздел 6).

Примечание — О характеристиках системы и архитектуры программного обеспечения, для которых необходима определенность при выборе подразделения, ответственного за соответствие требованиям 7.4, см. 7.4.3.

7.4.2.2 В соответствии с требуемым уровнем полноты безопасности и конкретными техническими требованиями к функции безопасности выбранный метод проектирования должен обладать характеристиками, которые облегчают:

a) абстрактное представление, разделение на модули и другие характеристики, контролирующие уровень сложности;

b) выражение:

1) выполняемых функций,

2) обмена данными между элементами,

3) информации, относящейся к последовательности и времени выполнения программ,

4) ограничений синхронизации,

5) параллельного и синхронизированного доступа к совместно используемым ресурсам,

6) структур данных и их свойств,

7) проектных предположений и их зависимостей,

8) обработки исключений,

9) проектных предположений (предварительных условий, постусловий, инвариантов),

10) комментариев;

c) возможность описания нескольких представлений проекта, включая представление структуры и представление поведения;

d) понимание разработчиками и другими лицами, которые должны иметь дело с проектом;

e) верификацию и оценку соответствия.

7.4.2.3 Тестируемость и способность к модификации системы безопасности должны быть предусмотрены на этапе проектирования для того, чтобы облегчить реализацию этих характеристик в окончательной версии системы, связанной с безопасностью.

Примечание — Наример, режимы эксплуатации в машиностроении и на обрабатывающих предприятиях.

7.4.2.4 Выбранный метод проектирования должен обладать характеристиками, облегчающими модификацию программного обеспечения. К числу таких характеристик относят модульность, скрытие информации и инкапсуляцию.

7.4.2.5 Представление проекта должно основываться на нотации, которая является однозначно определенной или ограничена до однозначно определенных свойств.

7.4.2.6 Проект должен, насколько это возможно, минимизировать ту часть программного обеспечения, которая связана с безопасностью.

7.4.2.7 Проект программного обеспечения должен включать в себя (соразмерно требуемому уровню полноты безопасности) средства самоконтроля потоков управления и потоков данных. При обнаружении отказа должны быть выполнены соответствующие действия.

7.4.2.8 Если программное обеспечение должно реализовать функции как относящиеся, так и не относящиеся к безопасности, оно в целом должно рассматриваться как относящееся к безопасности, если в проекте не предусмотрены соответствующие меры, гарантирующие, что отказы функций, не относящихся к безопасности, не могут оказать негативное влияние на функции, относящиеся к безопасности.

7.4.2.9 Если программное обеспечение должно реализовать функции безопасности, имеющие различный уровень полноты безопасности, то следует считать, что все программное обеспечение имеет уровень наивысший среди этих уровней, если только в проекте не будет продемонстрирована достаточная независимость функций, имеющих различный уровень полноты безопасности. Должно быть продемонстрировано, что либо независимость обеспечена в пространстве и во времени, либо любые нарушения независимости находятся под контролем. Обоснование независимости должно быть документально оформлено.

Примечание — Методы достижения одного аспекта независимости программного обеспечения приведены в приложении F.

7.4.2.10 Если стойкость к систематическим отказам элемента программного обеспечения ниже, чем уровень полноты безопасности функции безопасности, к которой он относится, то этот элемент должен использоваться в сочетании с другими элементами, такими, что стойкость к систематическим отказам такого сочетания элементов будет равна уровню полноты безопасности функции безопасности.

7.4.2.11 Если функция безопасности реализуется с использованием комбинации элементов программного обеспечения, для которых известны их значения стойкости к систематическим отказам, то к такой комбинации элементов должны применяться требования стойкости к систематическим отказам, представленные в 7.4.3 IEC 61508-2.

Примечание — Существует различие между функцией безопасности, от начала до конца реализуемой одним или более элементами, и функцией безопасности элемента, то есть каждого из элементов, участвующего в реализации. Если два элемента объединяются для достижения более высокой стойкости к систематическим отказам, то в такой комбинации каждый из этой пары элементов должен быть способен к предотвращению/смягчению опасного события, при этом функции безопасности каждого из этих элементов не обязательно должны быть идентичны, и не требуется, чтобы каждый из элементов комбинации был способен независимо обеспечивать функциональную безопасность, которая задана для всей комбинации.

Пример — В управлении электронного дросселя автомобиля функция безопасности «предотвращение нежелательного ускорения» полностью реализуется на двух процессорах. Функция безопасности элемента, реализуемая основным контроллерм, управляет поведением дросселя в режиме запрос/ответ. Функция безопасности элемента, реализуемая вторым процессором, выполняет разнообразный контроль (см. С.3.4 IEC 61508-7) и аварийный останов в случае необходимости. Комбинация этих двух процессоров дает более высокую уверенность в том, что выполнение всей функции безопасности «предотвращение нежелательного ускорения» будет обеспечено.

7.4.2.12 Если для реализации всей или части функции безопасности используется вновь уже существующий элемент программного обеспечения, то этот элемент должен соответствовать обоим следующим требованиям систематической полноты безопасности:

a) соответствовать требованиям одного из следующих способов обеспечения соответствия:

- способ 1_S: разработка, соответствующая требованиям. Соответствие требованиям настоящего стандарта для предотвращения и управления систематическими отказами в программном обеспечении;
- способ 2_S: проверка в эксплуатации. Представить свидетельства, что элемент проверен в эксплуатации. См. 7.4.10 IEC 61508-2;
- способ 3_S: оценка разработки, не соответствующей требованиям. Соблюдение требований 7.4.2.13;

Примечание 1 — Способы 1_S, 2_S и 3_S соответствуют способам, описанным в перечислении c) 7.4.2.2 IEC 61508-2, для элементов программного обеспечения. Они воспроизведены только, чтобы минимизировать обращение к IEC 61508-2.

Примечание 2 — В соответствии с 3.2.8 IEC 61508-4 уже существующее программное обеспечение может быть доступным коммерческим продуктом, или оно, возможно, было разработано конкретной организацией для предыдущего изделия или системы. Уже существующее программное обеспечение могло или не могло быть разработано в соответствии с требованиями настоящего стандарта.

Примечание 3 — Требования к уже существующим элементам применяются также к библиотеке времени выполнения или интерпретатору;

b) разработать руководство по безопасности (см. приложение D IEC 61508-2 и приложение D настоящего стандарта), которое дает достаточно точное и полное описание элемента для обеспечения оценки полноты конкретной функции безопасности, полностью или частично зависящей от уже существующего элемента программного обеспечения.

Примечание 1 — Руководство по безопасности может быть получено из собственной документации поставщика элемента и описания процесса разработки поставщика элемента или создано либо расширено дополнительными квалифицированными действиями, выполненными разработчиком системы, связанной с безопасностью, или третьей стороной. В некоторых случаях может понадобиться инженерный анализ для создания спецификации или разработки документации, соответствующей требованиям данного пункта с учетом сложившихся правовых условий (например, авторское право или права интеллектуальной собственности).

Примечание 2 — Обоснование элемента может быть разработано во время планирования безопасности (см. раздел 6).

7.4.2.13 В соответствии со способом обеспечения соответствия 3_S, уже существующий элемент программного обеспечения должен соответствовать всем следующим требованиям перечислений a)—i):

a) Спецификация требований к программному обеспечению системы безопасности для элемента в его новом приложении должна быть документально оформлена подробно, в соответствии с требованиями настоящего стандарта для любого элемента, связанного с безопасностью, с той же стойкостью к систематическим отказам. Спецификация требований к программному обеспечению системы без-

опасности должна охватывать функциональное и безопасное поведение, и применена к элементу в его новом применении, как определено в 7.2. (см. таблицу 1).

b) Обоснование для использования элемента программного обеспечения должно представить свидетельства о том, что были рассмотрены требуемые свойства системы безопасности, определенные в 7.2.2, 7.4.3, 7.4.4, 7.4.5, 7.4.6, 7.4.7, 7.5.2, 7.7.2, 7.8.2, 7.9.2 и разделе 8 с учетом требований приложениия С.

c) В достаточно подробно документально оформленном проекте элемента должны быть представлены свидетельства соответствия со спецификацией требований и требуемой стойкостью к систематическим отказам. См. 7.4.3, 7.4.5 и 7.4.6 и таблицы А.2 и А.4 приложения А.

d) Свидетельства по 7.4.2.13, перечисление a) и по 7.4.2.13, перечисление b) должны охватывать интеграцию программного обеспечения и аппаратных средств. См. 7.5 и таблицу А.6 приложения А.

e) Требуется доказательство, что для элемента были выполнены процедуры проверки и подтверждения соответствия, используя систематический подход с документально оформленным тестированием и анализом всех частей проекта элемента и кода. См. 7.4.7, 7.4.8, 7.5, 7.7, 7.9 и таблицы А.5—А.7 и А.9 приложения А, а также, связанные с ними таблицы приложения В.

Примечание — Для того, чтобы удовлетворить требованиям тестирования может быть использован положительный опыт применения вероятностных методов и метода «черного ящика» (см. таблицы А.7 приложения А и В.3 приложения В).

f) Если элемент программного обеспечения выполняет функции, которые не требуются системе, связанной с безопасностью, то должно быть представлено свидетельство о том, что ненужные функции не мешают E/E/PE системе соответствовать требованиям безопасности.

Примечание — Способы, соответствующие данному требованию, включают в себя:

- устранение таких функций из проекта;
- отключение таких функций;
- использование соответствующей архитектуры системы (например, декомпозиция на части, упаковка в отдельный файл, разнообразие, проверка достоверности выходов);
- увеличение объема тестирования.

g) Должно быть доказано, что все вероятностные механизмы отказа элемента программного обеспечения были идентифицированы и реализованы соответствующие меры их ослабления.

Примечание — Соответствующие меры ослабления включают в себя:

- использование соответствующей архитектуры системы (например, декомпозиция на части, упаковка в отдельный файл, разнообразие, проверка достоверности выходов),
- обработка исключений.

h) При планировании использования элемента должны быть идентифицированы конфигурация элемента программного обеспечения, среды выполнения программного и аппаратного обеспечения и (при необходимости) конфигурация системы компиляции/редактирования связей.

i) Обоснованием использования элемента должно быть проведение для него процедуры подтверждения соответствия только для тех применений, которые соответствуют предположениям руководства для этого элемента по безопасности применяемых изделий (см. приложение D IEC 61508-2 и приложение D настоящего стандарта).

7.4.2.14 Пункт 7.4.2, насколько это уместно, должен применяться к данным и языкам генерации данных.

Примечание — Руководящие указания по системам, управляемым данными, см. в приложении G.

a) Если ПЭ система обладает уже существующей функциональностью, которая сконфигурирована данными и соответствует конкретным требованиям применения, то проект прикладного программного обеспечения должен соответствовать степени конфигурируемости применения, существующей у предварительно поставляемой функциональности и сложности ПЭ системы, связанной с безопасностью.

b) Если функциональность ПЭ системы, связанной с безопасностью, определена в значительной степени или в основном конфигурационными данными, то для предотвращения появления отказов во время проектирования, производства, загрузки и модификации данных конфигурации и уверенности, что конфигурационные данные правильно формируют логику применения, должны использоваться соответствующие методы и средства.

c) Спецификация структур данных должна быть:

1) не противоречивой функциональным требованиям системы, включая данные применения,

2) полной,

3) внутренне непротиворечивой,

4) такой, чтобы структуры данных были защищены от изменения или повреждения.

d) Если ПЭ система обладает уже существующей функциональностью, которая сконфигурирована данными и соответствует определенным требованиям применения, то сам процесс конфигурации должен быть соответственно документально оформлен.

7.4.3 Требования к проектированию архитектуры программного обеспечения

Примечание 1 — Архитектура программного обеспечения определяет основные элементы и подсистемы программного обеспечения, их взаимосвязь, способ реализации необходимых характеристик и, в частности, полноты безопасности. Архитектура программного обеспечения также определяет общее поведение программного обеспечения, и как элементы программного обеспечения реализуют интерфейс и взаимодействуют между собой. Примерами основных компонентов программного обеспечения являются операционные системы, базы данных, подсистемы ввода и вывода УО, коммуникационные подсистемы, прикладные программы, инструментальные средства программирования и диагностики и т. п.

Примечание 2 — В некоторых отраслях промышленности архитектура программного обеспечения может называться «описание функций или спецификация функций проекта» (хотя эти документы могут также включать в себя вопросы, относящиеся к аппаратным средствам).

Примечание 3 — В некоторых случаях пользовательского прикладного программирования, в частности, в языках, используемых в программируемых логических контроллерах (ПЛК) (IEC 61508-6, приложение E), архитектура определяется поставщиком как стандартная характеристика ПЛК. Однако в соответствии с требованиями настоящего стандарта к поставщику может быть предъявлено требовани, гарантировать пользователю соответствие поставляемого продукта требованиям 7.4. Пользователь приспосабливает ПЛК, используя стандартные возможности программирования, например, многозвенные логические схемы. Требования 7.4.3—7.4.8 остаются в силе. Требование определения и документирования архитектуры может рассматриваться как информация, которую пользователь может использовать при выборе ПЛК (или эквивалентного ему устройства) для применения.

Примечание 4 — С точки зрения системы безопасности стадия разработки архитектуры программного обеспечения соответствует периоду, когда для программного обеспечения разрабатывается базовая стратегия безопасности.

Примечание 5 — Хотя стандарты серии IEC 61508 устанавливают числовые значения целевых мер отказов для функций безопасности, выполняемых E/E/PE системами, связанными с безопасностью, систематическая полнота безопасности, обычно не определяется количественно (см. 3.5.6 IEC 61508-4), но полнота безопасности программного обеспечения (см. 3.5.5 IEC 61508-4) определяется как стойкость к систематическим отказам со шкалой уверенности от 1 до 4 (см. 3.5.9 IEC 61508-4). Для целей настоящего стандарта принято, что программная ошибка может быть безопасной или опасной в зависимости от специфики использования программного обеспечения. Архитектура системы/программного обеспечения должна быть такой, чтобы опасные отказы элемента были ограничены каким-либо архитектурным ограничением, а методы разработки должны эти ограничения учитывать. В соответствии с требованиями настоящего стандарта методы разработки и подтверждения соответствия применяют со всей строгостью, которая является качественно согласованной с требуемой стойкостью к систематическим отказам.

Примечание 6 — Для выбора соответствующих методов и средств (см. приложения A и B), выполняющих требования настоящего пункта, должны быть рассмотрены следующие свойства (см. руководство по интерпретации свойств в приложении C и неформальные описания методов и средств в приложении F IEC 61508-7) архитектуры программного обеспечения:

- полнота спецификации требований к программному обеспечению системы безопасности;
- корректность спецификации требований к программному обеспечению системы безопасности;
- отсутствие собственных ошибок проекта;
- простота и ясность;
- предсказуемость поведения;
- верифицируемость и тестируемость проекта;
- отказоустойчивость;
- защита от отказов по общей причине, вызванной внешним событием.

7.4.3.1 В зависимости от характера разработки программного обеспечения ответственность за соответствие 7.4.4 может лежать на нескольких сторонах. Распределение ответственности должно быть документально оформлено во время планирования системы безопасности (см. раздел 6 IEC 61508-1).

7.4.3.2 Проект архитектуры программного обеспечения должен быть создан поставщиком программного обеспечения и/или разработчиком, описание архитектуры должно быть подробным. Описание должно:

a) содержать выбор и обоснование (см. 7.1.2.7) интегрированного набора методов и мероприятий, которые будут необходимы в течение жизненного цикла программного обеспечения системы безопасности для того, чтобы соответствовать требованиям к программному обеспечению системы безопасности для заданного уровня полноты безопасности. Эти методы и мероприятия включают в себя стратегию проектирования программного обеспечения для обеспечения устойчивости к отказам (совместимую с аппаратными средствами) и избежания отказов, в том числе, включая в себя (при необходимости) избыточность и разнообразие;

b) основываться на разделении на элементы/подсистемы, для каждой из которых должна предоставляться следующая информация:

1) проводилась ли верификация и если проводилась, то при каких условиях,

2) связан ли каждый из этих компонентов/подсистем с безопасностью или нет,

3) существует ли стойкость к систематическим отказам для компонента/подсистемы программного обеспечения;

c) определять все взаимодействия между программным обеспечением и аппаратным средствами, а также оценивать и детализировать их значение.

Примечание — Если взаимодействие между программным обеспечением и аппаратными средствами уже определено архитектурой системы, то достаточно сослаться на архитектуру системы;

d) использовать для представления архитектуры нотацию, которая является однозначно определенной или ограничена до подмножества однозначно определенных характеристик;

e) содержать набор проектных характеристик, которые должны использоваться для поддержания полноты безопасности всех данных. В число таких данных допускается включать входные и выходные производственные, коммуникационные данные, данные интерфейса оператора, сопровождения и хранящиеся во внутренних базах данных;

f) определять соответствующие тесты интеграции архитектуры программного обеспечения для обеспечения выполнения спецификации требований к программному обеспечению системы безопасности для заданного уровня полноты безопасности.

7.4.3.3 Любые изменения, которые может потребоваться внести в спецификацию требований к E/E/PE системе безопасности (см. 7.4.3.2), после использования мероприятий по 7.4.3.2 должны быть согласованы с разработчиком E/E/PE систем и документально оформлены.

Примечание — Итерационное взаимодействие между архитектурой аппаратных средств и программного обеспечения является неизбежным (см. рисунок 5), поэтому существует необходимость в обсуждении с разработчиком аппаратуры таких вопросов, как спецификация тестирования интеграции программируемой электронной аппаратуры и программного обеспечения (см. 7.5).

7.4.4 Требования к инструментальным средствам поддержки, включая языки программирования

Примечание — Для выбора соответствующих методов и средств (см. приложения A и B) для выполнения требований настоящего пункта должны быть рассмотрены следующие свойства (см. руководство по интерпретации свойств в приложении C и неформальные описания методов и средств в приложении F IEC 61508-7) архитектуры программного обеспечения:

- уровень, до которого инструментальные средства поддерживают разработку программного обеспечения с требуемыми свойствами программного обеспечения;
- четкость работы и функциональность инструментальных средств;
- корректность и воспроизводимость результата.

7.4.4.1 Программное обеспечение инструментальных средств поддержки, работающее в неавтономном режиме, должно рассматриваться как элемент программного обеспечения системы, связанной с безопасностью.

Примечание — Примеры инструментальных средств поддержки, работающие в автономном и неавтономном режиме см. в 3.2.10 и 3.2.11 IEC 61508-4.

7.4.4.2 Действие по выбору программного обеспечения инструментальных средств поддержки, работающих в автономном режиме, должно быть тесно связанным с частью действий по разработке программного обеспечения.

Примечание 1 — Требования к жизненному циклу разработки программного обеспечения см. в 7.1.2.

Примечание 2 — Для увеличения целостности программного обеспечения за счет уменьшения вероятности появления или необнаружения отказов во время его разработки должны использоваться соответствующие инструментальные средства, работающие в неавтономном режиме, поддерживающие разработку программного обеспечения. Примерами инструментальных средств, использующихся на стадиях разработки жизненного цикла программного обеспечения, являются:

a) инструментальные средства преобразования или трансляции, которые преобразуют программное обеспечение или представление проекта (например, текст или схему) из одного уровня абстрактного представления в другой уровень: инструментальные средства усовершенствования проекта, компиляторы, ассемблеры, компоновщики, редакторы связей, загрузчики и средства генерации кода;

b) средства оценки и подтверждения соответствия, такие как статические анализаторы кода, средства контроля тестового охвата, средства доказательства теорем и средства моделирования;

c) инструментальные средства диагностики для поддержки и контроля программного обеспечения в процессе эксплуатации;

d) инструментальные средства инфраструктуры, такие как системы поддержки разработок;

e) инструментальные средства управления конфигурацией, такие как средства управления версиями;

f) инструментальные средства данных применения, которые создают или поддерживают данные, необходимые для определения параметров и создания экземпляров функций системы. Такие данные включают в себя параметры функций, диапазоны инструментальных средств, уровни срабатывания и отключения аварийных сигналов, состояния выхода, которые будут восприняты как отказы.

Примечание 3 — Инструментальные средства поддержки, работающие в автономном режиме, должны быть выбраны как интегрируемые. Инструментальные средства интегрированы, если они работают совместно так, чтобы, выходы одного инструментального средства имели соответствующее содержание и формат для автоматической передачи на вход к следующему инструментальному средству, минимизируя таким образом возможность появления ошибки человека при повторной обработке промежуточных результатов.

Примечание 4 — Инструментальные средства поддержки, работающие в автономном режиме, должны быть выбраны как совместимые с потребностями применения системы, связанной с безопасностью, и интегрированного комплекса инструментальных средств.

Примечание 5 — Необходимо рассмотреть наличие подходящих инструментальных средств, чтобы предоставить сервисы, необходимые для всего жизненного цикла E/E/PE системы, связанной с безопасностью, (например, средства поддержки спецификаций, проектирования, реализации, документирования, модификации).

Примечание 6 — Необходимо уделить внимание компетентности пользователей выбранных инструментальных средств. Требования к компетентности см. раздел 6 IEC 61508-1.

7.4.4.3 Выбор инструментальных средств поддержки, работающих в автономном режиме, должен быть обоснован.

7.4.4.4 Все инструментальные средства поддержки классов Т2 и Т3, работающие в автономном режиме, должны иметь спецификацию или документацию на это средство, в которой ясно определено поведение инструментального средства и любые инструкции или ограничения при их использовании. Требования к жизненному циклу разработки программного обеспечения см. в 7.1.2, а для категорий программного обеспечения инструментальных средств поддержки, работающих в автономном режиме, см. в 3.2.11 IEC 61508-4.

Примечание — Такая «спецификация или документация на инструментальное средство» не является руководством по безопасности применяемого элемента (см. приложение D IEC 61508-2 и настоящий стандарт) непосредственно для самого инструментального средства. Руководство по безопасности для применяемого элемента касается уже существующего элемента, который включен в исполняемую систему, связанную с безопасностью. Если уже существующий элемент был сгенерирован инструментальным средством класса Т3 и затем включен в исполняемую систему, связанную с безопасностью, то любая релевантная информация (например, если в документации для оптимизирующего компилятора указано, что порядок оценки параметров функции не гарантируется) из «спецификации или документации на инструментальное средство» должна быть включена в руководство по безопасности для применяемого элемента, что позволяет провести оценку полноты конкретной функции безопасности, которая полностью или частично зависит от элемента, включенного в исполняемую систему, связанную с безопасностью.

7.4.4.5 Для того, чтобы определить уровень доверия к инструментальным средствам и возможные механизмы отказа инструментальных средств, которые могут повлиять на исполняемое программное обеспечение, должна быть выполнена оценка инструментальных средств поддержки классов Т2 и Т3, работающих в автономном режиме. Если механизмы отказа идентифицированы, то должны быть использованы соответствующие меры по их ослаблению.

Примечание 1 — Программное обеспечение HAZOP реализует один из методов анализа последствий возможных отказов, который можно использовать для программного обеспечения инструментального средства.

Примечание 2 — Примерами мер по ослаблению являются: предотвращение известных ошибок, ограниченное использование функциональности инструментального средства, проверка выходных результатов инструментального средства, использование разнообразных инструментальных средств для той же цели.

7.4.4.6 Для каждого инструментального средства в классе Т3 должно быть в наличии свидетельство о том, что инструментальное средство соответствует спецификации или документации на него. Такое свидетельство может быть основано на подходящей комбинации информации о предыдущем успешном использовании в подобных средах и для подобных применений (в данной организации или в других организациях), и подтверждении соответствия инструментального средства, как определено в 7.4.4.7.

Примечание 1 — История версий может обеспечивать уверенность в полноте готовности инструментального средства, а также должны быть учтены записи ошибок / несоответствий, когда инструментальное средство используется для разработки в новой среде.

Примечание 2 — Свидетельство для инструментального средства класса Т3 может также использоваться для инструментальных средств класса Т2 для оценки правильности их результатов.

7.4.4.7 Результаты подтверждения соответствия инструментальных средств должны быть документально оформлены и содержать:

a) хронологическую запись действий по подтверждению соответствия;

b) версию используемого руководства по инструментальному средству;

c) функции инструментального средства, для которых проводится процедура подтверждения соответствия;

d) используемые инструментальные средства и оборудование;

e) результаты действия по подтверждению соответствия; документально оформленные результаты подтверждения соответствия должны установить, что подтверждено соответствие программного обеспечения или существуют причины для отказа;

f) контрольные примеры и их результаты для последующего анализа;

g) несоответствия между ожидаемыми и фактическими результатами.

7.4.4.8 Если свидетельство соответствия по 7.4.4.6 недоступно, то должны быть предприняты эффективные меры для управления отказами рассматриваемой системы, связанной с безопасностью, которые являются следствием ошибок инструментального средства.

Примечание — Примером такой меры является средство генерации разнообразного избыточного кода, которое позволяет обнаружить, и управлять отказами рассматриваемой системы, связанной с безопасностью, которые произошли в ней из-за ошибок транслятора.

7.4.4.9 Должна быть проверена совместимость инструментальных средств в интегрированном комплексе инструментальных средств.

Примечание — Инструментальные средства интегрированы, если они работают совместно так, чтобы выходы одного инструментального средства имели соответствующее содержание и формат для автоматической передачи на вход к следующему инструментальному средству, минимизируя таким образом возможность появления ошибки человека при повторной обработке промежуточных результатов. См. В.3.5 приложения В IEC 61508-7.

7.4.4.10 В той степени, в которой этого требует уровень полноты безопасности, выбранное представление программного обеспечения или проекта (включая язык программирования) должно:

a) иметь транслятор, оцененный для проверки его пригодности (если необходимо), включая подтверждение соответствия требованиям национальных или международных стандартов;

b) использовать свойства языка, определенные только для него;

c) соответствовать характеристикам применения;

d) обладать свойствами, облегчающими обнаружение ошибок при проектировании или программировании;

e) поддерживать характеристики, соответствующие методу проектирования.

Примечание 1 — Языки программирования являются классом программного обеспечения и используются для представлений проекта. Транслятор преобразует программное обеспечение или представление проекта (например, текст или блок-схему) из одного уровня абстракции в другой уровень. Примерами трансляторов являются: инструменты усовершенствования проекта, компиляторы, ассемблеры, компоновщики, редакторы связей, загрузчики и инструменты генерации кода.

Примечание 2 — Оценка транслятора может быть выполнена для конкретного проекта применения или класса применений. В последнем случае вся необходимая информация об инструментальном средстве («спецификации или руководстве по инструментальному средству», см. 7.4.4.4), его назначении и надлежащем использовании должна быть доступной пользователю инструментального средства. В таком случае оценка инструментального средства для конкретного проекта может быть сокращена до проверки общей пригодности инструментального средства для проекта и соответствия со «спецификацией или руководством по инструментальному средству» (то есть оценивают правильное использование инструментального средства). Правильное использование инструментального средства может включать в себя дополнительные действия по проверке в рамках конкретного проекта.

Примечание 3 — Для того, чтобы оценить пригодность транслятора для выполнения своей цели согласно заданным критериям, которые должны включать в себя функциональные и нефункциональные требования, может использоваться процедура подтверждения соответствия (то есть набор тестовых программ, корректная трансляция которых известна заранее). Основным методом подтверждения соответствия транслятора его функциональным требованиям может быть динамическое тестирование. По возможности должна использоваться автоматическая процедура тестирования.

7.4.4.11 Если требования 7.4.4.10 не могут быть полностью выполнены, то необходимо обосновать пригодность языка для выполнения своей цели, а также использовать любые дополнительные меры, направленные на устранение любых идентифицированных недостатков языка.

7.4.4.12 Языки программирования для разработки всего программного обеспечения, связанного с безопасностью, должны использоваться в соответствии со стандартами составления программ для таких языков.

Примечание — Руководящие указания по использованию стандартов кодирования для программного обеспечения системы безопасности см. в IEC 61508-7.

7.4.4.13 Стандарты составления программ должны определять правильные методы программирования, запрещать использование небезопасных возможностей языка (например, неопределенных особенностей языка, неструктурированных конструкций и т. п.), упрощать проверку и тестирование и определять процедуры для документирования исходного текста. Документация, относящаяся к исходному тексту, должна содержать, по меньшей мере, следующую информацию:

a) юридическое лицо (например, компания, автор(ы) и т. д.);

b) описание;

c) входные и выходные данные;

d) историю управления конфигурацией.

7.4.4.14 Если выполняется автоматическая генерация кода или применяется автоматический транслятор, то необходимо провести оценку пригодности автоматического транслятора для разработки системы, связанной с безопасностью, для тех стадий жизненного цикла разработки, на которых применяют инструментальные средства поддержки разработки.

7.4.4.15 Если инструментальные средства поддержки классов Т2 и Т3, работающие в автономном режиме, генерируют элементы для базовой конфигурации, то управление конфигурацией должно гарантировать, чтобы информация об инструментальных средствах была записана в базовой конфигурации. В частности, информация об инструментальных средствах должна включать в себя:

a) идентификацию инструментального средства и его версии;

b) идентификацию элементов базовой конфигурации, для которых использовалась данная версия инструментального средства;

c) последовательность использования инструментального средства (включая параметры инструментального средства, опции и выбранные сценарии) для каждого базового элемента конфигурации.

Примечание — Цель данного подпункта состоит в том, чтобы позволить реконструировать базовую конфигурацию.

7.4.4.16 Управление конфигурацией должно гарантировать, что для инструментальных средств в классах Т2 и Т3 используются только квалифицированные версии.

7.4.4.17 Управление конфигурацией должно гарантировать, что используются только инструментальные средства, совместимые друг с другом и с системой, связанной с безопасностью.

Примечание — Аппаратные средства системы, связанной с безопасностью, могут также наложить ограничения на совместимость программного обеспечения инструментальных средств, например, эмулятор процессора должен быть точной моделью реальной электроники процессора.

7.4.4.18 Каждая новая версия инструментального средства поддержки, работающего в автономном режиме, должна быть квалифицирована. Эта квалификация может опираться на доказательства, представленные для более ранней версии, при наличии достаточных доказательства при условии, что:

a) функциональные различия (при наличии) не будут влиять на совместимость инструментального средства с остальной частью набора инструментальных средств и

b) новая версия вряд ли будет содержать принципиально новые, неизвестные отказы.

Примечание — Доказательство того, что новая версия вряд ли будет содержать принципиально новые, неизвестные отказы, может быть основано на ясной идентификации выполненных изменений, анализе действий по проверке и подтверждению соответствия, выполняемых для новой версии, и любом существующем опыте работы других пользователей с новой версией.

7.4.4.19 В зависимости от характера разработки программного обеспечения ответственность за соответствие требованиям 7.4.4 может лежать на нескольких сторонах. Распределение ответственности должно быть документально оформлено во время планирования системы безопасности (см. раздел 6 IEC 61508-1).

7.4.5 Требования к детальному проектированию и разработке — проектирование системы программного обеспечения

Примечание 1 — Под детальным проектированием понимается разделение основных элементов архитектуры на систему программных модулей, проектирование отдельных программных модулей и их программирование. В небольших приложениях проектирование программных систем и архитектуры может быть объединено.

Примечание 2 — Характер детального проектирования и разработки могут изменяться в зависимости от характера процессов разработки программ и архитектуры программного обеспечения (см. 7.4.3). Если прикладное программирование выполняется, например, с помощью языков многозвенных логических схем и функциональных блоков, то детальное проектирование может рассматриваться скорее как конфигурирование, чем как программирование. Тем не менее, правильный стиль программирования состоит в структурировании программного обеспечения, включая организацию модульной структуры, которая выделяет (насколько это возможно) блоки, связанные с безопасностью; в использовании проверок на попадание в интервал допустимых значений и других возможностей защиты от ошибок при вводе исходных данных; в использовании ранее верифицированных программных модулей; в применении проектных решений, которые облегчают выполнение будущих модификаций программного обеспечения.

Примечание 3 — При выборе соответствующих методов и средств (см. приложения А и В настоящего стандарта) для выполнения требований настоящего пункта должны быть рассмотрены следующие свойства (см. руководство по интерпретации свойств в приложении С настоящего стандарта и неформальные описания методов и средств в приложении F IEC 61508-7) проектирования и разработки:

- полнота спецификации требований к программному обеспечению системы безопасности;
- корректность спецификации требований к программному обеспечению системы безопасности;
- отсутствие собственных ошибок проекта;
- простота и ясность;
- предсказуемость поведения;
- поддающийся проверке и тестированию проект;
- отказоустойчивость/обнаружение неисправностей;
- отсутствие отказов по общей причине.

7.4.5.1 В зависимости от характера разработки программного обеспечения ответственность за соответствие требованиям 7.4.4 может лежать на нескольких сторонах. Распределение ответственности должно быть документально оформлено во время планирования системы безопасности (см. раздел 6 IEC 61508-1).

7.4.5.2 До начала детального проектирования должна быть подготовлена следующая информация: спецификация требований к E/E/PE системе, связанной с безопасностью, описание проекта архитектуры программного обеспечения, план подтверждения соответствия аспектов программного обеспечения системы безопасности.

7.4.5.3 Программное обеспечение следует разрабатывать так, чтобы достигалась модульность, тестируемость и способность к модификации системы безопасности.

7.4.5.4 Дальнейшее уточнение проекта для каждого главного элемента/подсистемы в описании проекта архитектуры программного обеспечения должно основываться на декомпозиции системы на программные модули (т.е. на спецификации проекта программной системы). Необходимо специфицировать проект каждого программного модуля и проверки этих модулей.

Примечание 1 — Об уже существующих элементах программного обеспечения см. 7.4.2.

Примечание 2 — Верификация состоит из тестирования и анализа.

7.4.5.5 Должны быть определены соответствующие проверки интеграции программной системы, показывающие, что программная система соответствуют требованиям к программному обеспечению системы безопасности для заданного уровня полноты безопасности.

7.4.6 Требования к реализации исходных текстов программ

Примечание — В соответствии с требуемым уровнем полноты безопасности исходный код должен обладать следующими свойствами (о конкретных методах см. приложения А и В, руководящие указания по интерпретации свойств см. в приложении С к настоящему стандарту):

- быть читаемым, понятным и пригодным к проверке;
- соответствовать специфицированным требованиям к проекту программного модуля (см. 7.4.5);
- соответствовать специфицированным требованиям к стандартам составления программ (см. 7.4.4);
- соответствовать требованиям, определенным при планировании системы безопасности (см. раздел 6).

7.4.6.1 Каждый модуль программного обеспечения должен быть просмотрен. Если код создан с помощью автоматических средств, то он должен соответствовать требованиям 7.4.4. Если исходный код состоит из повторно используемого уже существующего программного обеспечения, то он должен соответствовать требованиям 7.4.2.

Примечание — Просмотр кода относится к процессам верификации (см. 7.9). Просмотр кода может быть выполнен с помощью контроля кода (в порядке увеличения строгости): (1) человеком; (2) сквозным контролем программного обеспечения (см. С.5.15 приложения С IEC 61508-7); или (3) формальной проверкой (см. С.5.14 приложения С IEC 61508-7).

7.4.7 Требования к тестированию программных модулей

Примечание 1 — Процесс проверки того, что программный модуль корректно выполняет все требования, содержащиеся в спецификации тестирования, относится к процессам верификации (см. 7.9). Сочетание просмотра исходных текстов и тестирования программных модулей дает гарантию того, что программный модуль соответствует требованиям своей спецификации, то есть верифицирует модуль.

Примечание 2 — При выборе соответствующих методов и средств (см. приложения А и В настоящего стандарта) для выполнения требований настоящего пункта должны быть рассмотрены следующие свойства (см. руководство по интерпретации свойств в приложении С настоящего стандарта и неформальные описания методов и средств в приложении F IEC 61508-7) тестирования программных модулей:

- полнота тестирования в соответствии со спецификацией проекта программного обеспечения;
- корректность тестирования в соответствии со спецификацией проекта программного обеспечения (успешное выполнение);
- воспроизводимость;
- точно определенная конфигурация тестирования.

7.4.7.1 Каждый программный модуль должен быть протестирован в соответствии со спецификацией, разработанной при проектировании программного обеспечения (см. 7.4.5).

Примечание — Верификация состоит из тестирования и анализа.

7.4.7.2 Эти проверки должны продемонстрировать, что каждый программный модуль выполняет функции, для которых он предназначен, и не выполняет функций, которые не были для него предусмотрены.

Примечание 1 — Сказанное выше не означает тестирования всех комбинаций входных и выходных данных. Достаточным может быть тестирование всех классов эквивалентности или структурное тестирование. Анализ граничных значений или потоков управления может сократить число проверок до приемлемого уровня. Программы, пригодные для анализа, могут помочь в достижении более быстрого выполнения требований. Перечисленные методы см. в приложении С IEC 61508-7.

Примечание 2 — Если при разработке используют формальные методы, формальные доказательства или операторы проверки условий, то область применения подобных проверок может быть уменьшена. Перечисленные методы см. в приложении С IEC 61508-7.

Примечание 3 — Хотя систематическая полнота безопасности обычно количественно не определяется (см. 3.5.6 IEC 61508-4), определенные количественно статистические данные (например, статистическое тестиро-

вание, повышение надежности) являются приемлемыми, если удовлетворены все соответствующие условия для статистически достоверного доказательства, например, см. приложение D IEC 61508-7.

Примечание 4 — Если модуль настолько прост, что для него реально провести исчерпывающий тест, то это может быть самым эффективным способом демонстрации соответствия.

7.4.7.3 Результаты тестирования программных модулей должны быть документально оформлены.

7.4.7.4 Должны быть определены процедуры для коррекции при непрохождении теста.

7.4.8 Требования к тестированию интеграции программного обеспечения

Примечание — Проверка того, что интеграция программного обеспечения является корректной, относится к процессам верификации (см. 7.9).

7.4.8.1 Проверки интеграции программного обеспечения должны разрабатываться на этапе проектирования и разработки (см. 7.4.5).

7.4.8.2 Проверки интеграции системы программного обеспечения должны определять:

a) разделение программного обеспечения на контролируемые интегрируемые подмножества;

b) контрольные примеры и контрольные данные;

c) типы проверок, которые должны быть проведены;

d) условия тестирования, используемые инструменты, конфигурацию и программы;

e) условия, при которых проверка считается выполненной, и

f) процедуры, которые необходимо выполнить, если проверка дала отрицательный результат.

7.4.8.3 Программное обеспечение должно быть проверено в соответствии с тестами интеграции программ, определенными в спецификации тестирования интеграции системы программного обеспечения. Эти тесты должны продемонстрировать, что все программные модули и программные элементы/подсистемы корректно взаимодействуют для выполнения функций, для которых они предназначены, и не выполняют непредусмотренных функций.

Примечание 1 — Сказанное выше не означает тестирования всех комбинаций входных и выходных данных. Достаточным может быть тестирование всех классов эквивалентности или структурное тестирование. Анализ граничных значений или анализ потоков управления могут сократить число проверок до приемлемого уровня. Программы, пригодные для анализа, могут помочь в достижении более быстрого выполнения требований. Перечисленные методы см. в приложении C IEC 61508-7.

Примечание 2 — Если при разработке используются формальные методы, формальные доказательства или операторы проверки условий, то область применения подобных проверок может быть уменьшена. Перечисленные методы см. в приложении C IEC 61508-7.

Примечание 3 — Хотя систематическая полнота безопасности обычно количественно не определяется (см. 3.5.6 IEC 61508-4), определенные количественно статистические данные (например, статистическое тестирование, повышение надежности) являются приемлемыми, если удовлетворены все соответствующие условия для статистически достоверного доказательства, например, см. приложение D IEC 61508-7.

7.4.8.4 Результаты проверки интеграции программного обеспечения должны быть документально оформлены; в документации должны быть сформулированы результаты проверки и должно быть указано, были ли выполнены цели и критерии проверки. Если тестирование окончилось неудачно, должны быть описаны причины.

7.4.8.5 При интеграции программного обеспечения все модификации должны быть объектом анализа влияния, который должен определить, какие программные модули затрагиваются изменениями, и установить необходимость повторной верификации и проектирования.

7.5 Интеграция программируемой электроники (аппаратных средств и программного обеспечения)

Примечание — Эта стадия представлена на рисунке 4 (см. 10.4).

7.5.1 Цели

7.5.1.1 Первой целью требований настоящего подраздела является интеграция программного обеспечения с используемой программируемой электронной аппаратурой.

7.5.1.2 Второй целью требований настоящего подраздела является объединение программного обеспечения и аппаратных средств в программируемый электронный комплекс, связанный с безопасностью, проверка их совместимости и выполнения требований назначенного уровня полноты безопасности.

Примечание 1 — Проверка корректности интеграции программного обеспечения с аппаратными средствами программируемой электроники относится к процессам верификации (см. 7.9).

Примечание 2 — В зависимости от характера применения эти проверки могут быть объединены с проверками по 7.4.8.

7.5.2 Требования

Примечание — При выборе соответствующих методов и средств (см. приложения А и В настоящего стандарта) для выполнения требований настоящего пункта должны быть рассмотрены следующие свойства (см. руководство по интерпретации свойств в приложении С настоящего стандарта и неформальные описания методов и средств в приложении F IEC 61508-7) интеграции:

- полнота интеграции в соответствии со спецификациями проекта;
- корректность интеграции в соответствии со спецификациями проекта (успешное выполнение);
- воспроизводимость;
- точно определенная конфигурация интеграции.

7.5.2.1 Проверки интеграции должны быть определены на этапе проектирования и разработки (см. 7.4.3), их целью является проверка совместимости программного обеспечения и аппаратных средств в программируемом электронном устройстве, связанном с безопасностью.

Примечание — При разработке проверок интеграции может потребоваться тесная кооперация с разработчиком E/E/PE системы.

7.5.2.2 Спецификация тестов интеграции для программируемой электроники (аппаратных средств и программного обеспечения) должна определять:

a) разбиение системы на уровни интеграции;

b) тестовые примеры и тестовые данные;

c) типы выполняемых проверок;

d) условия тестирования, включая инструменты, программы поддержки и описание конфигурации;

e) условия, при которых проверка считается выполненной.

7.5.2.3 Тесты интеграции программируемой электроники (аппаратных средств и программного обеспечения) должны различать операции, выполняемые разработчиком на его оборудовании, и операции, требующие доступа к пользовательскому оборудованию.

7.5.2.4 Тесты интеграции программируемой электроники (аппаратных средств и программного обеспечения) должны различать следующие процессы:

a) включение программного обеспечения в целевое программируемое электронное оборудование;

b) интеграцию E/E/PE систем, т.е. добавление интерфейсов, таких как датчики и исполнительные устройства;

c) полную интеграцию УО и E/E/PE системы, связанной с безопасностью.

Примечание — Процессы по перечислениям b) и c) рассматриваются в IEC 61508-1 и IEC 61508-2.

7.5.2.5 Программное обеспечение должно быть интегрировано с программируемой электронной аппаратурой, связанной с безопасностью, в соответствии со специфицированными тестами интеграции для программируемой электроники (аппаратных средств и программного обеспечения).

7.5.2.6 При тестировании интеграции программируемой электроники, связанной с безопасностью (аппаратных средств и программного обеспечения), все изменения в интегрированной системе должны стать объектом анализа влияния, который должен определить, какие программные модули затрагиваются изменениями, и установить необходимость повторной верификации.

7.5.2.7 Тестовые примеры и результаты их выполнения должны быть документально оформлены для последующего анализа.

7.5.2.8 Результаты проверки интеграции программируемой электроники (аппаратных средств и программного обеспечения) должны быть документально оформлены, в документации должны быть сформулированы результаты проверки, а также указано, были ли выполнены цели и критерии проверки. Если тестирование окончилось неудачно, должны быть описаны причины неудачи. Все модификации или изменения, являющиеся результатом тестирования, должны быть объектом анализа влияния, который должен определить, какие программные элементы/модули затрагиваются изменениями, и установить необходимость повторной верификации и проектирования.

7.6 Процедуры эксплуатации и модификации программного обеспечения

Примечание — Эта стадия представлена на рисунке 4 (см. 10.5).

7.6.1 Цели

Целью требований настоящего подраздела является представление информации и процедур, касающихся программного обеспечения, необходимых, чтобы убедиться в том, что функциональная безопасность E/E/PE связанной с безопасностью системы сохраняется при эксплуатации и модификациях.

7.6.2 Требования

Требования приведены в 7.6 IEC 61508-2 и 7.8 настоящего стандарта.

Примечание — В настоящем стандарте программное обеспечение (в отличие от аппаратных средств) не может подвергаться техническому обслуживанию, оно всегда модифицируется.

7.7 Подтверждение соответствия безопасности системы аспектов программного обеспечения

Примечание 1 — Данная стадия представлена на рисунке 4 (см. 10.6).

Примечание 2 — Обычно подтверждение соответствия программного обеспечения не может проводиться отдельно от его аппаратных средств и системного окружения.

7.7.1 Цели

Целью требований настоящего подраздела является гарантировать соответствие интегрированной системы специфицированным требованиям системы безопасности к программному обеспечению для заданного уровня полноты безопасности.

7.7.2 Требования

Примечание — При выборе соответствующих методов и средств (см. приложения А и В) для выполнения требований настоящего подраздела должны быть рассмотрены следующие свойства (см. руководство по интерпретации свойств в приложении С настоящего стандарта и неформальные описания методов и средств в приложении F IEC 61508-7) подтверждения соответствия безопасности системы:

- полнота подтверждения соответствия в соответствии со спецификацией проекта программного обеспечения;
- корректность подтверждения соответствия в соответствии со спецификацией проекта программного обеспечения (успешное выполнение);
- воспроизводимость;
- подтверждение соответствия точно определенной конфигурации.

7.7.2.1 Если для программного обеспечения, связанного с безопасностью, соответствие с требованиями уже было установлено при планировании подтверждения соответствия безопасности для E/E/PE системы, связанной с безопасностью (см. 7.7 IEC 61508-2), то проводить повторное подтверждение соответствия не требуется.

7.7.2.2 Операции подтверждения соответствия должны проводиться в соответствии со спецификациями, разработанными при планировании подтверждения соответствия безопасности системы аспектов программного обеспечения.

7.7.2.3 В зависимости от характера разработки программного обеспечения ответственность за соответствие требованиям 7.7 может лежать на нескольких сторонах. Распределение ответственности должно быть документально оформлено во время планирования безопасности системы (см. раздел 6 IEC 61508-1).

7.7.2.4 Результаты подтверждения соответствия безопасности системы аспектов программного обеспечения должны быть документально оформлены.

7.7.2.5 При проведении подтверждения соответствия программного обеспечения безопасности системы для каждой функции безопасности должны быть документально оформлены следующие результаты:

а) хронологический перечень операций подтверждения соответствия, который позволит восстановить последовательность действий.

Примечание — При записи результатов испытаний важно, чтобы последовательность действий могла быть восстановлена. Главное в этом требовании — восстановить последовательность действий, а не создать упорядоченный по времени/дате список документов;

b) используемая версия плана подтверждения соответствия программного обеспечения безопасности системы (см. 7.3);

c) подтверждаемые функции безопасности (с использованием тестирования или анализа), со ссылками на план подтверждения соответствия безопасности системы аспектов программного обеспечения (см. 7.3);

d) используемые инструменты и оборудование, а также данные калибровки;

e) результаты операций подтверждения соответствия;

f) расхождения между ожидаемыми и фактическими результатами.

7.7.2.6 При наличии расхождений между ожидаемыми и фактическими результатами проводят анализ и принимают решение о продолжении подтверждения соответствия или о подготовке запроса на изменение и возвращении к более ранней стадии жизненного цикла разработки. Это решение должно быть документально оформлено как часть результатов подтверждения соответствия безопасности системы аспектов программного обеспечения.

Примечание — Требования 7.7.2.2—7.7.2.5 основываются на общих требованиях 7.14 IEC 61508-1.

7.7.2.7 Подтверждение соответствия безопасности системы аспектов связанного с безопасностью программного обеспечения должно соответствовать следующим требованиям:

a) основным методом подтверждения соответствия для программного обеспечения должно быть тестирование; анимацию и моделирование допускается использовать как дополнительные методы;

b) прогон программного обеспечения должен проводиться путем имитации:

1) входных сигналов в нормальном режиме работы,

2) предполагаемых случаев,

3) нежелательных условий, требующих вмешательства системы;

c) поставщик и/или разработчик (либо несколько сторон, ответственных за соответствие) должны предоставить документально оформленные результаты подтверждения соответствия безопасности системы для аспектов программного обеспечения и всю имеющую отношение к этой операции документацию в распоряжение разработчика системы, чтобы дать ему возможность выполнить требования IEC 61508-1 и IEC 61508-2.

7.7.2.8 Инструментальные средства программного обеспечения должны соответствовать требованиям 7.4.4.

7.7.2.9 К результатам подтверждения соответствия безопасности системы аспектов связанного с безопасностью программного обеспечения предъявляют следующие требования:

a) проверки должны показать, что все заданные требования, предъявляемые к программному обеспечению, связанному с безопасностью, (см. 7.2), выполняются правильно и что программная система не выполняет непредусмотренных функций;

b) тестовые примеры и их результаты должны быть документально оформлены для последующего анализа и независимо оценены в соответствии с требованиями уровня полноты безопасности (раздел 8 IEC 61508-1);

c) документально оформленные результаты подтверждения соответствия безопасности системы для аспектов программного обеспечения должны содержать либо утверждение о том, что программа получила подтверждение соответствия, либо причины, по которым она его не получила.

7.8 Модификация программного обеспечения

Примечание — Эта стадия представлена на рисунке 4 (см. 10.5).

7.8.1 Цель

Целью требований настоящего подраздела является внесение корректировок, улучшений или изменений в принятое программное обеспечение, гарантирующих сохранение стойкости к систематическим отказам программного обеспечения.

Примечание — В настоящем стандарте программное обеспечение (в отличие от аппаратного обеспечения) не может поддерживаться, оно всегда модифицируется.

7.8.2 Требования

Примечание — При выборе соответствующих методов и средств (см. приложения А и В настоящего стандарта) для выполнения требований настоящего подраздела должны быть рассмотрены следующие свойства (см.

руководство по интерпретации свойств в приложении С настоящего стандарта и неформальные описания методов и средств в приложении F IEC 61508-7) модификации программного обеспечения:

- полнота модификации в соответствии с требованиями модификации;
- корректность модификации в соответствии с требованиями модификации;
- отсутствие собственных ошибок проекта;
- предотвращение нежелательного поведения;
- верифицируемость и тестируемость проекта;
- регрессионное тестирование и охват проверкой.

7.8.2.1 Перед выполнением какой-либо модификации программного обеспечения, должны быть подготовлены процедуры модификации (см. 7.16 IEC 61508-1).

Примечание 1 — Требования 7.8.2.1—7.8.2.9 относятся в первую очередь к изменениям, выполняемым на стадии эксплуатации программного обеспечения. Требования 7.8.2.1—7.8.2.9 могут также применяться на стадии интеграции программируемой электроники, а также на стадиях установки и ввода в эксплуатацию всей системы безопасности (см. 7.13 IEC 61508-1).

Примечание 2 — Пример модели процедуры модификации приведен на рисунке 9 IEC 61508-1.

7.8.2.2 Процесс модификации может начинаться только после появления запроса на санкционированную модификацию программного обеспечения в рамках процедур, определенных на этапе планирования системы безопасности (см. раздел 6), в котором приведена подробная информация:

a) об опасностях, на которые могут повлиять изменения;

b) о предлагаемых изменениях;

c) о причинах изменений.

Примечание — Причины появления запроса на модификацию могут быть, например, связаны:

- с тем, что функциональная безопасность оказалась ниже той, которая была определена в спецификациях;
- накопленным опытом о систематических отказах;
- появлением нового или изменением действующего законодательства, относящегося к безопасности;
- модификацией управляемого оборудования или способа его использования;
- модификацией общих требований к системе безопасности;
- анализом характеристик эксплуатации и технического обслуживания, который показывает, что эти характеристики имеют значения ниже запланированных;
- текущим аудитом систем функциональной безопасности.

7.8.2.3 Должен быть выполнен анализ влияния предлагаемых модификаций программного обеспечения на функциональную безопасность E/E/PE системы, связанной с безопасностью, чтобы определить:

a) необходим ли анализ рисков;

b) какие стадии жизненного цикла программного обеспечения системы безопасности следует повторить.

7.8.2.4 Результаты анализа влияния, полученные в соответствии с 7.8.2.3, должны быть документально оформлены.

7.8.2.5 Все модификации, оказывающие влияние на функциональную безопасность E/E/PE системы, связанной с безопасностью, должны приводить к возврату на соответствующую стадию жизненного цикла программного обеспечения системы безопасности. Все последующие стадии должны выполняться в соответствии с процедурами, определенными для конкретных стадий в соответствии с требованиями настоящего стандарта. При планировании системы безопасности (см. раздел 6) должны быть подробно описаны все последующие процессы.

Примечание — Может потребоваться проведение полного анализа рисков и опасностей, в результате которого может появиться потребность в иных уровнях полноты безопасности, чем те, которые определены для функций безопасности, реализуемых E/E/PE системами, связанными с безопасностью.

7.8.2.6 Планирование системы безопасности для модификации программного обеспечения, связанного с безопасностью, должно соответствовать требованиям, представленным в разделе 6 IEC 61508-1, в частности к:

a) идентификации персонала и определению требований к его квалификации;

b) подробной спецификациюи модификации;

c) планированию верификации;

d) определению области применения процедур повторного подтверждения соответствия и тестирования модификации в той степени, в которой этого требует уровень полноты безопасности.

Примечание — В зависимости от характера применения может быть важным участие экспертов в области данного применения.

7.8.2.7 Модификация должна быть проведена в соответствии с разработанным планом.

7.8.2.8 Все модификации должны быть подробно документированы, включая:

a) запрос на модификацию/корректировку;

b) результаты анализа влияния, которое окажут предлагаемые модификации программного обеспечения на функциональную безопасность, и принятые решения с их обоснованием;

c) сведения об изменениях конфигурации программного обеспечения;

d) отклонения от нормальной работы и нормальных условий работы;

e) документы, которые затрагиваются процессами модификации.

7.8.2.9 Информация о деталях всех проведенных модификаций должна быть документально оформлена. Документация должна включать в себя данные и результаты повторной верификации и повторного подтверждения соответствия.

7.8.2.10 Оценка необходимых модификаций или корректировок должна зависеть от результатов анализа влияния модификаций и стойкости к систематическим отказам программного обеспечения.

7.9 Верификация программного обеспечения

7.9.1 Цель

Целью требований настоящего подраздела является проверка и оценка в соответствии с требуемым уровнем полноты безопасности результатов, полученных на заданной стадии жизненного цикла программного обеспечения системы безопасности, а также гарантия того, что эти результаты являются корректными и соответствуют исходной информации для соответствующей стадии.

Примечание 1 — Настоящий подраздел учитывает базовые аспекты верификации, которые являются общими для нескольких стадий жизненного цикла системы безопасности. Настоящий подраздел не предъявляет дополнительных требований к элементам проверки при верификации в 7.4.7 (проверка программных модулей), 7.4.8 (интеграция программного обеспечения) и 7.5 (интеграция программируемой электроники), которые сами по себе представляют процессы верификации. Данный подраздел не требует также дополнительной верификации для процессов подтверждения соответствия программного обеспечения (см. 7.7), так как подтверждение соответствия в настоящем стандарте определяется как демонстрация соответствия спецификации требований к системе безопасности. Проверка того, является ли корректной сама спецификация, проводится специалистами по предметным областям.

Примечание 2 — В зависимости от архитектуры программного обеспечения ответственность за проведение верификации программного обеспечения может быть поделена между всеми организациями, вовлеченными в разработку и модификацию программного обеспечения.

7.9.2 Требования

Примечание — При выборе соответствующих методов и средств (см. приложения А и В настоящего стандарта) для выполнения требований настоящего подраздела должны быть рассмотрены следующие свойства (см. руководство по интерпретации свойств в приложении С настоящего стандарта и неформальные описания методов и средств в приложении F IEC 61508-7) верификации данных:

- полнота верификации в соответствии с предыдущей стадии;
- корректность верификации в соответствии с предыдущей стадией (успешное выполнение);
- воспроизводимость;
- верификация точно определенной конфигурации.

7.9.2.1 Верификация программного обеспечения для каждой стадии жизненного цикла программного обеспечения системы безопасности должна планироваться (см. 7.4) одновременно с разработкой; вся информация, относящаяся к этому вопросу, должна быть документально оформлена.

7.9.2.2 Планирование верификации программного обеспечения должно касаться критериев, методов и инструментария, используемого при верификации; в ходе его должны быть рассмотрены:

a) оценка требований полноты безопасности;

b) выбор и документирование стратегии, процессов и методов верификации;

c) выбор и использование инструментов верификации (тестовая программа, специальные программные средства для тестирования, имитаторы ввода/вывода и т. п.);

d) оценка результатов верификации;

e) исправления, которые должны быть сделаны.

7.9.2.3 Верификация программного обеспечения должна быть проведена в соответствии с планом.

Примечание — Выбор методов и средств, предназначенных для верификации, а также степень независимости процессов верификации определяются рядом факторов и могут быть определены в стандартах для прикладных отраслей. К числу таких факторов относятся, например:

- размер проекта;
- степень сложности;
- степень новизны проекта;
- степень новизны технологии.

7.9.2.4 Должны быть документально оформлены свидетельства того, что верифицируемая стадия завершена удовлетворительно во всех отношениях.

7.9.2.5 Документация, составляемая после каждой верификации, должна включать в себя:

a) идентификацию параметров, подлежащих верификации;

b) идентификацию информации, необходимой для выполнения верификации.

Примечание — Информация необходимая для проведения верификации, включает в себя: (но не ограничивается): входную информацию, полученую от предыдущей стадии жизненного цикла, стандарты проектирования, стандарты кодирования и используемые инструментальные средства;

c) перечень несоответствий.

Примечание — Например, несоответствия возможны в программных модулях, структурах данных и алгоритмах плохо адаптированных к задаче.

7.9.2.6 Вся существенная информация, относящаяся к стадии *N* жизненного цикла программного обеспечения системы безопасности, необходимая для правильного выполнения следующей стадии *N*+1, должна быть доступна и верифицирована. К выходной информации стадии *N* относятся:

a) информация об адекватности спецификации, описания проекта либо исходного текста программ, разработанных в ходе стадии *N*:

- функциональности,
- полноте безопасности, характеристикам и другим требованиям планирования системы безопасности (см. раздел 6),
- требованию понятности для коллектива разработчиков,
- тестируемости для дальнейшей верификации,
- безопасной модификации, допускающей дальнейшее развитие;

b) информация об адекватности планирования подтверждения соответствия и/или тестов, определенных для стадии *N*, определению и описанию проекта стадии *N*;

c) результаты проверки несовместимости между:

- тестами, определенными для стадии *N*, и тестами, определенными для предыдущей стадии *N*−1,
- выходными данными стадии *N*.

7.9.2.7 В соответствии с выбором жизненного цикла разработки программного обеспечения (см. 7.1) должны быть выполнены:

a) верификация требований к программному обеспечению системы безопасности;

b) верификация архитектуры программного обеспечения;

c) верификация проекта системы программного обеспечения;

d) верификация проектов программных модулей;

e) верификация исходных текстов программ;

f) верификация данных;

g) тестирование программных модулей (см. 7.4.7);

h) тестирование интеграции программного обеспечения (см. 7.4.8);

i) тестирование интеграции программируемой электроники (см. 7.5);

j) подтверждение соответствия аспектов программного обеспечения системы безопасности (см. 7.7).

Примечание — Требования по перечислениям a) — g) представлены ниже.

7.9.2.8 Верификация требований к программному обеспечению системы безопасности: после определения требований к программному обеспечению системы безопасности и перед началом следующей стадии проектирования и разработки программного обеспечения, верификация должна проверять:

a) соответствует ли спецификация требований к программному обеспечению системы безопасности спецификации требований к E/E/PE системе безопасности (см. 7.10 IEC 61508-1 и 7.2 IEC 61508-2) в отношении функциональности, полноты безопасности, характеристик и других требований к планированию системы безопасности;

b) соответствует ли план подтверждения соответствия безопасности системы для аспектов программного обеспечения спецификации требований к программному обеспечению системы безопасности;

c) наличие несовместимости между:

1) спецификацией требований к программному обеспечению системы безопасности и спецификацией требований к E/E/PE системе безопасности (см. 7.10 IEC 61508-1 и 7.2 IEC 61508-2),

2) спецификацией требований к программному обеспечению системы безопасности и планом подтверждения соответствия безопасности системы для аспектов программного обеспечения.

7.9.2.9 Верификация архитектуры программного обеспечения: после того, как выполнен проект архитектуры программного обеспечения, верификация должна проверить:

a) соответствует ли проект архитектуры программного обеспечения спецификации требований к программному обеспечению системы безопасности;

b) адекватны ли проверки интеграции, специфицированные в проекте архитектуры программного обеспечения;

c) адекватность атрибутов каждого основного элемента/подсистемы по отношению к:

1) реализуемости требуемых характеристик системы безопасности,

2) возможности проверки при последующей верификации,

3) пониманию персоналом, выполняющим разработку и верификацию,

4) модификации системы безопасности, позволяющей выполнять дальнейшее развитие программы;

d) наличие несовместимости между:

1) описанием проекта архитектуры программного обеспечения и спецификацией требований к программному обеспечению системы безопасности,

2) описанием проекта архитектуры программного обеспечения и специфицированными тестами интеграции архитектуры программного обеспечения,

3) специфицированными тестами интеграции проекта архитектуры программного обеспечения и планом подтверждения соответствия безопасности системы для аспектов программного обеспечения.

7.9.2.10 Верификация проекта системы программного обеспечения: после завершения проектирования системы программного обеспечения верификация должна проверить:

a) соответствует ли проект системы программного обеспечения (см. 7.4.5) проекту архитектуры программного обеспечения;

b) соответствуют ли специфицированные тесты интеграции системы программного обеспечения (см. 7.4.5) проекту системы программного обеспечения (см. 7.4.5);

c) адекватность спецификации атрибутов каждого основного элемента проекта системы программного обеспечения (см. 7.4.5) по отношению к:

1) реализуемости требуемых характеристик системы безопасности,

2) возможности проверки при последующей верификации,

3) пониманию персоналом, выполняющим разработку и верификацию,

4) модификации системы безопасности, позволяющей выполнять дальнейшее развитие программы.

Примечание — Проверки интеграции системы программного обеспечения могут быть определены как часть проверок интеграции архитектуры программного обеспечения;

d) наличие несовместимости между:

1) спецификацией проекта системы программного обеспечения (см. 7.4.5) и описанием проекта архитектуры программного обеспечения,

2) спецификацией проекта системы программного обеспечения (см. 7.4.5) и спецификацией тестов интеграции системы программного обеспечения (см. 7.4.5),

3) тестами, заданными спецификацией тестов интеграции системы программного обеспечения (см. 7.4.5), и спецификацией тестов интеграции архитектуры программного обеспечения (см. 7.4.3).

7.9.2.11 Верификация проекта модулей программного обеспечения: после того как выполнен проект каждого программного модуля, верификация должна проверить:

a) соответствует ли спецификация проекта программного модуля (см. 7.4.5) спецификации проекта системы программного обеспечения (см. 7.4.5);

b) адекватна ли спецификация проверок каждого программного модуля (см. 7.4.5) спецификации проекта программного модуля (см. 7.4.5);

c) адекватность атрибутов каждого программного модуля по отношению к:

1) реализуемости требуемых характеристик системы безопасности (см. спецификацию требований к программному обеспечению системы безопасности),

2) возможности проверки при последующей верификации,

3) пониманию персоналом, выполняющим разработку и верификацию,

4) модификации системы безопасности, позволяющей выполнять дальнейшее развитие программы;

d) наличие несовместимости между:

1) спецификацией проекта программного модуля (см. 7.4.5) и спецификацией проекта системы программного обеспечения (см. 7.4.5),

2) спецификацией проекта каждого программного модуля (см. 7.4.5) и спецификацией проверок этого программного модуля (см. 7.4.5),

3) спецификацией проверок программных модулей (см. 7.4.5) и спецификацией проверок интеграции системы программного обеспечения (см. 7.4.5).

7.9.2.12 Верификация исходного текста: исходный текст должен быть верифицирован статическими методами для того, чтобы гарантировать соответствие спецификации проекта программных модулей (см. 7.4.5), необходимым стандартам кодирования (см. 7.4.4) и плану подтверждения соответствия безопасности системы для аспектов программного обеспечения.

Примечание — На ранних стадиях жизненного цикла программного обеспечения системы безопасности верификация является статической (например, изучение, просмотр, формальная проверка и т. п.). При верификации исходного текста используют такие методы, как просмотр и прогон программного обеспечения. Сочетание результатов верификации исходных текстов и проверок программного обеспечения гарантирует, что каждый программный модуль будет соответствовать своей спецификации. С этого момента тестирование становится основным средством проверки.

7.9.2.13 Верификация данных:

a) структуры данных должны быть проверены;

b) прикладные данные должны быть проверены на:

1) соответствие структурам данных,

2) полноту по отношении к требованиям применения,

3) совместимость с системным программным обеспечением (например, для организации последовательности, управления во время выполнения и др.) и

4) правильность значений данных;

c) все эксплуатационные параметры должны быть проверены по отношению к требованиям применения;

d) все промышленные интерфейсы и соответствующее программное обеспечение [датчики и исполнительные устройства, а также автономные интерфейсы (см. 7.2.2.12)] должны быть проверены на:

1) выявление предполагаемых отказов интерфейса,

2) устойчивость по отношению к предполагаемым отказам интерфейса;

e) все коммуникационные интерфейсы и соответствующее программное обеспечение должны быть проверены на наличие адекватного уровня:

1) обнаружения ошибок,

2) защиты от повреждения,

3) подтверждения соответствия данных.

7.9.2.14 Верификация временны́х характеристик: должна быть проверена предсказуемость поведения во времени.

Примечание — Поведение во времени может определяться: производительностью средств, наличием ресурсов, временем реакции, временем выполнения в наихудшем случае, случаями переполнения памяти, случайными зависаниями, времнем работы системы.

8 Оценка функциональной безопасности

Примечание — При выборе соответствующих методов и средств (см. приложения А и В настоящего стандарта) для выполнения требований настоящего раздела должны быть рассмотрены следующие свойства (см. руководство по интерпретации свойств в приложении С настоящего стандарта и неформальные описания методов и средств в приложении F IEC 61508-7) оценки функциональной безопасности:

- полнота оценки функциональной безопасности в соответствии с требованиями настоящего стандарта;
- корректность оценки функциональной безопасности в соответствии с проектными спецификациями (успешное завершение);
- доступное для анализа решение всех выявленных проблем;
- возможность модификации оценки функциональной безопасности после изменения проекта без необходимости проведения серьезной переработки оценки;
- воспроизводимость;
- своевременность;
- точно определенная конфигурация.

8.1 Цели и требования раздела 8 IEC 61508-1 относятся к оценке программного обеспечения, связанного с безопасностью.

8.2 Если иное не оговорено в стандартах на область применения, то минимальный уровень независимости для лиц, выполняющих оценку функциональной безопасности, должен быть определен по разделу 8 IEC 61508-1.

8.3 Оценка функциональной безопасности может использовать результаты процессов, приведенных в таблице А.10 (приложение А).

Примечание — Выбор методов, приведенных в приложениях А и В, не гарантирует, что будет достигнута необходимая полнота безопасности (см. 7.1.2.7). Лицо, проводящее оценку, должно также рассмотреть:

- совместимость и взаимное дополнение выбранных методов, языков и инструментальных средств для всего цикла разработки;
- полностью ли разработчики понимают методы, языки и инструментальные средства, которые разработчики используют;
- насколько хорошо адаптированы методы, языки и инструментальные средства к конкретным проблемам, с которыми приходится сталкиваться при разработке.

Приложение А
(обязательное)

Руководство по выбору методов и средств

Некоторые из подразделов настоящего стандарта имеют ассоциированные с ними таблицы, например, подраздел 7.2 (спецификация требований к программному обеспечению системы безопасности) связан с таблицей А.1. Более подробные таблицы, содержащиеся в приложении В, раскрывают содержание некоторых элементов таблиц приложения А, например, таблица В.2 раскрывает содержание динамического анализа и тестирования из таблицы А.5.

Обзор методов и средств, упоминаемых в приложениях А и В, приведен в IEC 61508-7. Для каждого из них приведены рекомендации по уровню полноты безопасности, изменяющемуся от 1 до 4. Эти рекомендации обозначаются следующим образом:

HR	Настоятельно рекомендуется применять этот метод или средство для данного уровня полноты безопасности. Если метод или средство не используется, то на этапе планирования системы безопасности этому должно быть дано подробное объяснение со ссылкой на приложение С, и это объяснение должно быть согласовано с экспертом
R	Метод или средство рекомендуется применять для данного уровня полноты безопасности, но степень обязательности рекомендации ниже, чем в случае рекомендации HR
—	Для данного метода или средства рекомендации ни за, ни против не приводятся
NR	Данный метод или средство решительно не рекомендуется для этого уровня полноты безопасности. Если данный метод или средство применяют, то на стадии планирования системы безопасности этому должно быть дано подробное обоснование со ссылкой на приложение С, которое следует согласовать с экспертом

Методы и средства следует выбирать в соответствии с уровнем полноты безопасности. Альтернативные или эквивалентные методы и средства обозначают буквой, следующей за номером. Следует применять только один из альтернативных или эквивалентных методов/средств.

Могут применяться другие методы и средства при условии, что они соответствуют требованиям и целям стадии разработки программного обеспечения. Руководящие указания по выбору методов см. в приложении С.

Ранжирование методов и средств связано с концепцией эффективности, используемой в IEC 61508-2. При прочих равных условиях методы, имеющие ранг HR, будут более эффективны в предотвращении внесения систематических ошибок при разработке программного обеспечения либо (при разработке архитектуры программ) при выявлении ошибок, оставшихся необнаруженными на стадии выполнения, по сравнению с методами, имеющими ранг R.

При большом числе факторов, влияющих на стойкость к систематическим отказам программного обеспечения, невозможно дать алгоритм, определяющий такую комбинацию методов и средств, которая была бы корректной для любого заданного применения. Тем не менее, в приложении С приведено руководство по выбору конкретных методов для достижения стойкости к систематическим отказам программного обеспечения.

В случае конкретного применения соответствующая комбинация методов или средств должна быть сформулирована при планировании системы безопасности, при этом методы и средства должны применяться, если примечания к таблице не содержат иных требований.

Предварительное руководство в виде двух рабочих примеров по интерпретации таблиц приведено в [6].

Таблица А.1 — Спецификация требований к программному обеспечению системы безопасности (см. 7.2)

Метод/средство[1)]	Ссылка	УПБ1	УПБ2	УПБ3	УПБ4
1a Полуформальные методы	Таблица В.7	R	R	HR	HR
1b Формальные методы	В.2.2, С.2.4	—	R	R	HR
2 Прямая прослеживаемость между требованиями к системе безопасности и требованиями к программному обеспечению системы безопасности	С.2.11	R	R	HR	HR
3 Обратная прослеживаемость между требованиями к системе безопасности и предполагаемыми потребностями безопасности	С.2.11	R	R	HR	HR

Окончание таблицы А.1

Метод/средство[1)]	Ссылка	УПБ1	УПБ2	УПБ3	УПБ4
4 Компьютерные средства разработки спецификаций для поддержки, перечисленных выше подходящих методов/средств	B.2.4	R	R	HR	HR

[1)] Методы/средства следует выбирать в соответствии с уровнем полноты безопасности. Альтернативные или эквивалентные методы/средства обозначают буквами, следующими за числом. Следует применять только один из альтернативных или эквивалентных методов/мероприятий. Выбор альтернативных методов должен быть обоснован в соответствии со свойствами, приведенными в приложении С, желательно для каждого применения.

Примечание 1 — Спецификация требований к программному обеспечению системы безопасности всегда будет требовать описания задачи на естественном языке и использования необходимой системы математических обозначений, отражающих содержание приложения.

Примечание 2 — Таблица отражает дополнительные требования для ясного и точного определения требований к программному обеспечению системы безопасности.

Примечание 3 — См. таблицу С.1.

Примечание 4 — Ссылки (являющиеся справочными, а не обязательными) «B.x.x.x», «C.x.x.x» в столбце 2 указывают на подробные описания методов/средств, изложенные приложениях В и С [5].

Таблица А.2 — Проектирование и разработка программного обеспечения: проектирование архитектуры программного обеспечения (см. 7.4.3)

Метод/средство[1)]	Ссылка	УПБ1	УПБ2	УПБ3	УПБ4
1 Обнаружение ошибок	C.3.1	—	R	HR	HR
2 Коды обнаружения ошибок	C.3.2	R	R	R	HR
3a Программирование с проверкой ошибок	C.3.3	R	R	R	HR
3b Методы контроля (при реализации процесса контроля и контролируемой функции на одном компьютере обеспечивается их независимость)	C.3.4	—	R	R	—
3c Методы контроля (реализация процесса контроля и контролируемой функции на разных компьютерах)	C.3.4	—	R	R	HR
3d Многовариантное программирование, реализующее одну спецификацию требований к программному обеспечению системы безопасности	C.3.5	—	—	—	R
3e Функционально многовариантное программирование, реализующее различные спецификации требований к программному обеспечению системы безопасности	C.3.5	—	—	R	HR
3f Восстановление предыдущего состояния	C.3.6	R	R	—	HR
3g Проектирование программного обеспечения, не сохраняющего состояние (или проектирование ПО, сохраняющего ограниченное описание состояния)	C.2.12	—	—	R	HR
4a Механизмы повторных попыток парирования сбоя	C.3.7	R	R	—	—
4b Постепенное отключение функций	C.3.8	R	R	HR	HR
5 Исправление ошибок методами искусственного интеллекта	C.3.9	—	NR	NR	NR
6 Динамическая реконфигурация	C.3.10	—	NR	NR	NR
7 Модульный подход	Таблица B.9	HR	HR	HR	HR
8 Использование доверительных/проверенных элементов программного обеспечения (при наличии)	C.2.10	R	HR	HR	HR

Окончание таблицы А.2

Метод/средство[1)]	Ссылка	УПБ1	УПБ2	УПБ3	УПБ4
9 Прямая прослеживаемость между спецификацией требований к программному обеспечению системы безопасности и архитектурой программного обеспечения	С.2.11	R	R	HR	HR
10 Обратная прослеживаемость между спецификацией требований к программному обеспечению системы безопасности и архитектурой программного обеспечения	С.2.11	R	R	HR	HR
11a Методы структурных диаграмм[2)]	С.2.1	HR	HR	HR	HR
11b Полуформальные методы[2)]	Таблица В.7	R	R	HR	HR
11c Формальные методы проектирования и усовершенствования[2)]	В.2.2, С.2.4	—	R	R	HR
11d Автоматическая генерация программного обеспечения	С.4.6	R	R	R	R
12 Автоматизированные средства разработки спецификаций и проектирования	В.2.4	R	R	HR	HR
13a Циклическое поведение с гарантированным максимальным временем цикла	С.3.11	R	HR	HR	HR
13b Архитектура с временным распределением	С.3.11	R	HR	HR	HR
13c Управление событиями с гарантированным максимальным временем реакции	С.3.11	R	HR	HR	—
14 Статическое выделение ресурсов	С.2.6.3	—	R	HR	HR
15 Статическая синхронизация доступа к разделяемым ресурсам	С.2.6.3	—	—	R	HR

[1)] Методы/средства следует выбирать в соответствии с уровнем полноты безопасности. Альтернативные или эквивалентные методы/средства обозначают буквами, следующими за числом. Следует применять только один из альтернативных или эквивалентных методов/мероприятий. Выбор альтернативных методов должен быть обоснован в соответствии со свойствами, приведенными в приложении С, желательно для каждого применения.

[2)] Из группы 11 «Структурированные методы» следует применять метод 11a, только если метод 11b не подходит для предметной области с УПБ3 + УПБ4.

Примечание 1 — Некоторые методы в таблице А.2 посвящены концепциям проектирования, другие тому, как проект представляется.

Примечание 2 — Приведенные в настоящей таблице средства, касающиеся устойчивости к ошибкам (контроль ошибок), должны рассматриваться совместно с требованиями по IEC 61508-2 к архитектуре и контролю ошибок для аппаратных средств программируемых электронных устройств.

Примечание 3 — См. таблицу С.2.

Примечание 4 — Группа методов 13a—13c применяется только к системам и программному обеспечению с требованиями к синхронизации системы безопасности.

Примечание 5 — Метод 14: использование динамических объектов (например, при работе со стеком или с неупорядоченным массивом) может наложить требования на доступную память и время выполнения. Метод 14 не должен быть применен, если используется компилятор, который гарантирует, что:

a) перед выполнением будет выделено достаточно памяти для всех динамических переменных и объектов или в случае ошибки при выделении памяти будет достигнуто безопасное состояние;

b) что время отклика соответствует заданным требованиям.

Примечание 6 — Метод 4a: устранение неисправностей с помощью повторных попыток часто подходит при любом УПБ, но должен быть установлен предел для числа повторений.

Примечание 7 — Ссылки (являющиеся справочными, а не обязательными) «B.x.x.x», «C.x.x.x» в столбце 2 указывают на подробные описания методов/средств, изложенных в приложениях В и С [5].

Таблица А.3 — Проектирование и разработка программного обеспечения: инструментальные средства поддержки и языки программирования (см. 7.4.4)

Метод/средство[1)]	Ссылка	УПБ1	УПБ2	УПБ3	УПБ4
1 Выбор соответствующего языка программирования	С.4.5	HR	HR	HR	HR
2 Строго типизированные языки программирования	С.4.1	HR	HR	HR	HR
3 Подмножество языка	С.4.2	—	—	HR	HR
4a Сертифицированные средства и сертифицированные трансляторы	С.4.3	R	HR	HR	HR
4b Инструментальные средства, заслуживающие доверия на основании опыта использования	С.4.4	HR	HR	HR	HR

[1)] Методы/средства следует выбирать в соответствии с уровнем полноты безопасности. Альтернативные или эквивалентные методы/средства обозначают буквами, следующими за числом. Следует выполнять только один из альтернативных или эквивалентных методов/мероприятий. Выбор альтернативных методов должен быть обоснован в соответствии со свойствами, приведенными в приложении С, желательно для каждого применения.

Примечание 1 — См. таблицу С.3.

Примечание 2 — Ссылки (являющиеся справочными, а не обязательными) «B.x.x.x», «C.x.x.x» в столбце 2 указывают на подробные описания методов/средств, изложенных в приложениях В и С [5].

Таблица А.4 — Проектирование и разработка программного обеспечения: детальное проектирование (см. 7.4.5 и 7.4.6) (включает в себя проектирование системы программного обеспечения, проектирование модуля программного обеспечения и кодирование)

Метод/средство[1)]	Ссылка	УПБ1	УПБ2	УПБ3	УПБ4
1a Методы структурных диаграмм[2)]	С.2.1	HR	HR	HR	HR
1b Полуформальные методы[2)]	Таблица В.7	R	R	HR	HR
1c Формальные методы проектирования и усовершенствования[2)]	В.2.2, С.2.4	—	R	R	HR
2 Средства автоматизированного проектирования	В.3.5	R	R	HR	HR
3 Программирование с защитой	С.2.5	—	R	HR	HR
4 Модульный подход	Таблица В.9	HR	HR	HR	HR
5 Стандарты для проектирования и кодирования	С.2.6, таблица В.1	R	HR	HR	HR
6 Структурное программирование	С.2.7	HR	HR	HR	HR
7 Использование доверительных/проверенных программных модулей и компонентов (по возможности)	С.2.10	R	HR	HR	HR
8 Прямая прослеживаемость между спецификацией требований к программному обеспечению системы безопасности и проектом программного обеспечения	С.2.11	R	R	HR	HR

[1)] Методы/средства следует выбирать в соответствии с уровнем полноты безопасности. Альтернативные или эквивалентные методы/средства обозначают буквами, следующими за числом. Следует выполнять только один из альтернативных или эквивалентных методов/средств.

[2)] Из группы 1 «Структурированные методы» следует использовать метод 1a, только если метод 1b не подходит для предметной области с УПБ3 и УПБ4.

Примечание 1 — См. таблицу С.4.

Примечание 2 — Все еще обсуждается пригодность разработки объектноориентированного программного обеспечения для систем, связанных с безопасностью. Руководящие указания по использованию объектноориентированного подхода при разработке архитектуры и в проектировании см. приложение G IEC 61508-7.

Примечание 3 — Ссылки (являющиеся справочными, а не обязательными) «B.x.x.x», «C.x.x.x» в столбце 2 указывают на подробные описания методов/средств, изложенных в приложениях В и С [5].

Таблица А.5 — Проектирование и разработка программного обеспечения: тестирование и интеграция программных модулей (см. 7.4.7 и 7.4.8)

Метод/средство[1)]	Ссылка	УПБ1	УПБ2	УПБ3	УПБ4
1 Вероятностное тестирование	С.5.1	—	R	R	HR
2 Динамический анализ и тестирование	В.6.5, таблица В.2	R	HR	HR	HR
3 Регистрация и анализ данных	С.5.2	HR	HR	HR	HR
4 Функциональное тестирование и тестирование методом «черного ящика»	В.5.1, В.5.2, таблица В.3	HR	HR	HR	HR
5 Тестирование рабочих характеристик	Таблица В.6	R	R	HR	HR
6 Тестирование, основанное на модели	С.5.27	R	R	HR	HR
7 Тестирование интерфейса	С.5.3	R	R	HR	HR
8 Управление тестированием и средства автоматизации	С.4.7	R	HR	HR	HR
9 Прямая прослеживаемость между спецификацией проекта программного обеспечения и спецификациями тестирования модуля и интеграции	С.2.11	R	R	HR	HR
10 Формальная верификация	С.5.12	—	—	R	R

[1)] Методы/средства следует выбирать в соответствии с уровнем полноты безопасности.

Примечание 1 — Тестирование программных модулей и интеграции относится к процессам верификации (см. таблицу А.9).

Примечание 2 — См. таблицу С.5.

Примечание 3 — Формальная проверка может уменьшить размер и объем занимаемой памяти модуля, поэтому необходимо тестирование интеграции.

Примечание 4 — Ссылки (являющиеся справочными, а не обязательными) «В.х.х.х», «С.х.х.х» в столбце 2 указывают на подробные описания методов/средств, изложенных в приложениях В и С [5].

Таблица А.6 — Интеграция программируемых электронных устройств (программное обеспечение и аппаратные средства) (см. 7.5)

Метод/средство[1)]	Ссылка	УПБ1	УПБ2	УПБ3	УПБ4
1 Функциональное тестирование и тестирование методом «черного ящика»	В.5.1, В.5.2, таблица В.3	HR	HR	HR	HR
2 Тестирование рабочих характеристик	Таблица В.6	R	R	HR	HR
3 Прямая прослеживаемость между требованиями проекта системы и программного обеспечения к интеграции программных и аппаратных средств и спецификациями тестирования интеграции программных и аппаратных средств	С.2.11	R	R	HR	HR

[1)] Методы/средства следует выбирать в соответствии с уровнем полноты безопасности.

Примечание 1 — Интеграция программируемых электронных устройств относится к процессам верификации (см. таблицу А.9).

Примечание 2 — См. таблицу С.6.

Примечание 3 — Ссылки (являющиеся справочными, а не обязательными) «В.х.х.х», «С.х.х.х» в столбце 2 указывают на подробные описания методов/средств, изложенных в приложениях В и С [5].

Таблица А.7 — Подтверждение соответствия безопасности системы аспектов программного обеспечения (см. 7.7)

Метод/средство[1)]	Ссылка	УПБ1	УПБ2	УПБ3	УПБ4
1 Вероятностное тестирование	С.5.1	—	R	R	HR
2 Моделирование процесса	С.5.18	R	R	HR	HR
3 Моделирование	Таблица В.5	R	R	HR	HR
4 Функциональное тестирование и тестирование методом «черного ящика»	В.5.1, В.5.2, таблица В.3	HR	HR	HR	HR
5 Прямая прослеживаемость между спецификацией требований к программному обеспечению системы безопасности и планом подтверждения соответствия программного обеспечения системы безопасности	С.2.11	R	R	HR	HR
6 Обратная прослеживаемость между планом подтверждения соответствия программного обеспечения системы безопасности и спецификацией требований к программному обеспечению системы безопасности	С.2.11	R	R	HR	HR

[1)] Методы/средства следует выбирать в соответствии с уровнем полноты безопасности.

Примечание 1 — См. таблицу С.6.

Примечание 2 — Ссылки (являющиеся справочными, а не обязательными) «В.х.х.х», «С.х.х.х» в столбце 2 указывают на подробные описания методов/средств, изложенных в приложениях В и С [5].

Таблица А.8 — Модификация (см. 7.8)

Метод/средство[1)]	Ссылка	УПБ1	УПБ2	УПБ3	УПБ4
1 Анализ влияния	С.5.23	HR	HR	HR	HR
2 Повторная верификация измененных программных модулей	С.5.23	HR	HR	HR	HR
3 Повторная верификация программных модулей, на которые оказывают влияние изменения в других модулях	С.5.23	R	HR	HR	HR
4а Повторное подтверждение соответствия системы в целом	Таблица А.7	—	R	HR	HR
4b Регрессионное подтверждение соответствия	С.5.25	R	HR	HR	HR
5 Управление конфигурацией программного обеспечения	С.5.24	HR	HR	HR	HR
6 Регистрация и анализ данных	С.5.2	HR	HR	HR	HR
7 Прямая прослеживаемость между спецификацией требований к программному обеспечению системы безопасности и планом модификации программного обеспечения (включая повторные верификацию и подтверждение соответствия)	С.2.11	R	R	HR	HR
8 Обратная прослеживаемость между планом модификации программного обеспечения (включая повторную верификацию и подтверждение соответствия) и спецификацией требований к программному обеспечению системы безопасности	С.2.11	R	R	HR	HR

[1)] Методы/средства следует выбирать в соответствии с уровнем полноты безопасности. Альтернативные или эквивалентные методы/средства обозначают буквами, следующими за числом. Следует выполнять только один из альтернативных или эквивалентных методов/мероприятий. Выбор альтернативных методов должен быть обоснован в соответствии со свойствами, приведенными в приложении С, желательно для каждого применения.

Примечание 1 — См. таблицу С.8.

Примечание 2 — Методы группы 4 «Анализ влияния» являются необходимой частью метода «Регрессионного подтверждения соответствия». См. IEC 61508-7.

Примечание 3 — Ссылки (являющиеся справочными, а не обязательными) «В.х.х.х», «С.х.х.х» в столбце 2 указывают на подробные описания методов/средств, изложенных в приложениях В и С [5].

Таблица А.9 — Верификация программного обеспечения (см. 7.9)

Метод/средство[1)]	Ссылка	УПБ1	УПБ2	УПБ3	УПБ4
1 Формальное доказательство	С.5.12	—	R	R	HR
2 Анимация спецификации и тестирования	С.5.26	R	R	R	R
3 Статический анализ	В.6.4, таблица В.8	R	HR	HR	HR
4 Динамический анализ и тестирование	В.6.5,таблица В.2	R	HR	HR	HR
5 Прямая прослеживаемость между спецификацией проекта программного обеспечения и планом верификации (включая верификацию данных) программного обеспечения	С.2.11	R	R	HR	HR
6 Обратная прослеживаемость между планом верификации (включая верификацию данных) программного обеспечения и спецификацией проекта программного обеспечения	С.2.11	R	R	HR	HR
7 Численный анализ в автономном режиме	С.2.13	R	R	HR	HR
Тестирование и интеграция программного модуля	См. таблицу А.5				
Проверка интеграции программируемых электронных устройств	См. таблицу А.6				
Тестирование программной системы (подтверждение соответствия)	См. таблицу А.7				

[1)] Методы/средства следует выбирать в соответствии с уровнем полноты безопасности.

Примечание 1 — Для удобства все процессы, связанные с верификацией, были объединены в настоящей таблице. Это, однако, не накладывает дополнительных требований на элементы верификации, связанные с динамическим тестированием, в таблицах А.5 и А.6, которые сами по себе относятся к процессам верификации. Настоящая таблица также не требует проведения верификационного тестирования в дополнение к подтверждению соответствия программного обеспечения (см. таблицу В.7), которая в настоящем стандарте представляет демонстрацию соответствия спецификации требований к системе безопасности (конечную верификацию).

Примечание 2 — Требования к верификации включены в IEC 61508-1 — IEC 61508-3. Следовательно, первая верификация системы, связанной с безопасностью, относится к ранним спецификациям системного уровня.

Примечание 3 — На ранних стадиях жизненного цикла программного обеспечения системы безопасности верификация является статической, она может включать в себя, например, изучение, просмотр, формальную проверку. Когда программа готова, становится возможным проведение динамического тестирования. Для верификации требуется объединение информации обоих типов. Например, верификация программного модуля статическими средствами включает в себя такие методы, как просмотр программ, прогон, статический анализ, формальная проверка. Верификация программ динамическими средствами включает в себя функциональное тестирование, тестирование методом белого ящика, статистическое тестирование. Использование проверок обоих типов позволяет утверждать, что каждый программный модуль удовлетворяет соответствующей спецификации.

Примечание 4 — См. таблицу С.9.

Примечание 5 — Ссылки (являющиеся справочными, а не обязательными) «В.х.х.х», «С.х.х.х» в столбце 2 указывают на подробные описания методов/средств, изложенных в приложениях В и С [5].

Таблица А.10 — Оценка функциональной безопасности (см. раздел 8)

Метод/средство[1)]	Ссылка	УПБ1	УПБ2	УПБ3	УПБ4
1 Таблица контрольных проверок	В.2.5	R	R	R	R
2 Таблицы решений (таблицы истинности)	С.6.1	R	R	R	R
3 Анализ отказов	Таблица В.4	R	R	HR	HR
4 Анализ отказов по общей причине различного программного обеспечения (если используется различное программное обеспечение)	С.6.3	—	R	HR	HR

Окончание таблицы А.10

Метод/средство[1)]	Ссылка	УПБ1	УПБ2	УПБ3	УПБ4
5 Структурные схемы надежности	С.6.4	R	R	R	R
6 Прямая прослеживаемость между требованиями раздела 8 и планом оценки функциональной безопасности программного обеспечения	С.2.11	R	R	HR	HR

[1)] Соответствующие методы/средства следует выбирать в соответствии с уровнем полноты безопасности.

Примечание 1 — См. таблицу С.10.

Примечание 2 — Ссылки (являющиеся справочными, а не обязательными) «В.х.х.х», «С.х.х.х» в столбце 2 указывают на подробные описания методов / средств, изложенных в приложениях В и С [5].

Приложение B
(справочное)

Подробные таблицы

Таблица B.1 — Стандарты для проектирования и кодирования (см. таблицу A.4 приложения A)

Метод/средство[1)]	Ссылка	УПБ1	УПБ2	УПБ3	УПБ4
1 Использование стандартов кодирования для сокращения вероятности ошибок	C.2.6.2	HR	HR	HR	HR
2 Не использовать динамические объекты	C.2.6.3	R	HR	HR	HR
3a Не использовать динамические переменные	C.2.6.3	—	R	HR	HR
3b Проверка создания динамических переменных в неавтономном режиме	C.2.6.4	—	R	HR	HR
4 Ограниченное использование прерываний	C.2.6.5	R	R	HR	HR
5 Ограниченное использование указателей	C.2.6.6	—	R	HR	HR
6 Ограниченное использование рекурсий	C.2.6.7	—	R	HR	HR
7 Не использовать неструктурированное управление в программах, написанных на языках высокого уровня	C.2.6.2	R	HR	HR	HR
8 Не использовать автоматическое преобразование типов	C.2.6.2	R	HR	HR	HR

[1)] Методы/средства следует выбирать в соответствии с уровнем полноты безопасности. Альтернативные или эквивалентные методы/средства обозначают буквами, следующими за числом. Следует применять только один из альтернативных или эквивалентных методов/мероприятий. Выбор альтернативных методов должен быть обоснован в соответствии со свойствами, приведенными в приложении C, желательно для каждого применения.

Примечание 1 — Методы 2, 3a и 5: использование динамических объектов (например, при реализации стека или динамически распределяемой области памяти) может наложить ограничения на объем доступной памяти и время выполнения. Методы 2, 3a и 5 не должны применяться, если используется компилятор, который обеспечивает, что:

a) для всех динамических переменных и объектов перед выполнением будет выделено достаточно памяти, а в случае ошибки выделения памяти система перейдет в безопасное состояние;

b) время реакции системы соответствует заданным требованиям.

Примечание 2 — См. таблицу C.11.

Примечание 3 — Ссылки (являющиеся справочными, а не обязательными) «B.x.x.x», «C.x.x.x» в столбце 2 указывают на подробные описания методов/средств, изложенных в приложениях B и C [5].

Таблица B.2 — Динамический анализ и тестирование (см. таблицы A.5 и A.9 приложения A)

Метод/средство[1)]	Ссылка	УПБ1	УПБ2	УПБ3	УПБ4
1 Выполнение тестового примера, связанного с анализом граничных значений	C.5.4	R	HR	HR	HR
2 Выполнение тестового примера, связанного с предполагаемой ошибкой	C.5.5	R	R	R	R
3 Выполнение тестового примера, связанного с введением ошибки	C.5.6	–	R	R	R
4 Выполнение тестового примера, сгенерированного на основе модели	C.5.27	R	R	HR	HR
5 Моделирование реализации	C.5.20	R	R	R	HR
6 Разделение входных данных на классы эквивалентности	C.5.7	R	R	R	HR
7a Структурный тест со 100 %-ным охватом (точки входа)[2)]	C.5.8	HR	HR	HR	HR

Окончание таблицы В.2

Метод/средство[1)]	Ссылка	УПБ1	УПБ2	УПБ3	УПБ4
7b Структурный тест со 100 %-ным охватом (операторы)[2)]	С.5.8	R	HR	HR	HR
7c Структурный тест со 100 %-ным охватом (условные переходы)[2)]	С.5.8	R	R	HR	HR
7d Структурный тест со 100 %-ным охватом (составные условия, MC/DC)[2)]	С.5.8	R	R	R	HR

[1)] Методы/средства следует выбирать в соответствии с уровнем полноты безопасности.

[2)] Если 100 %-ный охват не может быть достигнут (например, охват оператора кода защиты), то должно быть дано соответствующее объяснение.

Примечание 1 — Анализ с использованием тестовых примеров проводят на уровне подсистем, он основывается на спецификациях и/или спецификациях и текстах программ.

Примечание 2 — См. таблицу С.12.

Примечание 3 — Ссылки (являющиеся справочными, а не обязательными) «В.х.х.х», «С.х.х.х» в столбце 2 указывают на подробные описания методов/средств, изложенных в приложениях В и С [5].

Таблица В.3 — Функциональное тестирование и проверка методом черного ящика (см. таблицы А.5, А.6 и А.7 приложения А)

Метод/средство[1)]	Ссылка	УПБ1	УПБ2	УПБ3	УПБ4
1 Выполнение тестового примера, на основе причинно-следственных диаграмм	В.6.6.2	—	—	R	R
2 Выполнение тестового примера, сгенерированного на основе модели	С.5.27	R	R	HR	HR
3 Макетирование/анимация	С.5.17	...	—	R	R
4 Разделение входных данных на классы эквивалентности, включая анализ граничных значений	С.5.7, С.5.4	R	HR	HR	HR
5 Моделирование процесса	С.5.18	R	R	R	R

[1)] Методы/средства следует выбирать в соответствии с уровнем полноты безопасности.

Примечание 1 — Анализ с использованием тестовых примеров выполняется на уровне систем программного обеспечения и он основывается только на спецификациях.

Примечание 2 — Полнота моделирования будет зависеть от уровня полноты безопасности, сложности и применения.

Примечание 3 — См. таблицу С.13.

Примечание 4 — Ссылки (являющиеся справочными, а не обязательными) «В.х.х.х», «С.х.х.х» в столбце 2 указывают на подробные описания методов/средств, изложенных в приложениях В и С [5].

Таблица В.4 — Анализ отказов (см. таблицу А.10 приложения А)

Метод/средство[1)]	Ссылка	УПБ1	УПБ2	УПБ3	УПБ4
1а Причинно-следственные диаграммы	В.6.6.2	R	R	R	R
1в Анализ методом дерева событий	В.6.6.3	R	R	R	R
2 Анализ методом дерева отказов	В.6.6.5	R	R	R	R
3 Анализ функциональных отказов программного обеспечения	В.6.6.4	R	R	R	R

[1)] Методы/средства следует выбирать в соответствии с уровнем полноты безопасности. Альтернативные или эквивалентные методы/средства обозначают буквами, следующими за числом. Следует применять только один из альтернативных или эквивалентных методов/мероприятий. Выбор альтернативных методов должен быть обоснован в соответствии со свойствами, приведенными в приложении С, желательно для каждого применения.

Примечание 1 — Предварительно должен быть проведен анализ рисков для определения, к какому уровню полноты безопасности следует отнести программное обеспечение.

Примечание 2 — См. таблицу С.14.

Примечание 3 — Ссылки (являющиеся справочными, а не обязательными) «В.х.х.х», «С.х.х.х» в столбце 2 указывают на подробные описания методов/средств, изложенных в приложениях В и С [5].

Таблица В.5 — Моделирование (см. таблицу А.7 приложения А)

Метод/средство[1)]	Ссылка	УПБ1	УПБ2	УПБ3	УПБ4
1 Диаграммы потоков данных	С.2.2	R	R	R	R
2a Метод конечных автоматов	В.2.3.2	—	R	HR	HR
2b Формальные методы	В.2.2, С.2.4	—	R	R	HR
2c Моделирование во времени сетями Петри	В.2.3.3	—	R	HR	HR
3 Моделирование реализации	С.5.20	R	HR	HR	HR
4 Макетирование/анимация	С.5.17	R	R	R	R
5 Структурные диаграммы	С.2.3	R	R	R	HR

[1)] Методы/средства следует выбирать в соответствии с уровнем полноты безопасности. Альтернативные или эквивалентные методы/средства обозначают буквами, следующими за числом. Следует выполнять только один из альтернативных или эквивалентных методов/мероприятий. Выбор альтернативных методов должен быть обоснован в соответствии со свойствами, приведенными в приложении С, желательно для каждого применения.

Примечание 1 — Если какой-то конкретный метод не перечислен в таблице, не следует считать, что он был исключен из рассмотрения. Этот метод должен соответствовать требованиям настоящего стандарта.

Примечание 2 — Количественное значение вероятностей не требуется.

Примечание 3 — См. таблицу С.15.

Примечание 4 — Ссылки (являющиеся справочными, а не обязательными) «В.х.х.х», «С.х.х.х» в столбце 2 указывают на подробные описания методов/средств, изложенных в приложениях В и С [5].

Таблица В.6 — Тестирование рабочих характеристик (см. таблицы А.5 и А.6 приложения А)

Метод/средство[1)]	Ссылка	УПБ1	УПБ2	УПБ3	УПБ4
1 Проверка на критические нагрузки и стресс-тестирование	С.5.21	R	R	HR	HR
2 Ограничения на время ответа и объем памяти	С.5.22	HR	HR	HR	HR
3 Требования к реализации	С.5.19	HR	HR	HR	HR

[1)] Методы/средства следует выбирать в соответствии с уровнем полноты безопасности.

Примечание 1 — См. таблицу С.16.

Примечание 2 — Ссылки (являющиеся справочными, а не обязательными) «В.х.х.х», «С.х.х.х» в столбце 2 указывают на подробные описания методов/средств, изложенных в приложениях В и С [5].

Таблица В.7 — Полуформальные методы см. таблицы А.1, А.2 и А.4 приложения А)

Метод/средство[1)]	Ссылка	УПБ1	УПБ2	УПБ3	УПБ4
1 Логические/функциональные блок-схемы	См. примечание 1	R	R	HR	HR
2 Диаграммы последовательности действий	См. примечание 1	R	R	HR	HR
3 Диаграммы потоков данных	С.2.2	R	R	R	R
4a Конечные автоматы/диаграммы переходов	В.2.3.2	R	R	HR	HR
4b Моделирование во времени сетями Петри	В.2.3.3	R	R	HR	HR
5 Модели данных сущность-связь-атрибут	В.2.4.4	R	R	R	R
6 Диаграммы последовательности сообщений	С.2.14	R	R	R	R
7 Таблицы решений и таблицы истинности	С.6.1	R	R	HR	HR
8 UML	С.3.12	R	R	R	R

Окончание таблицы В.7

[1] Методы/средства следует выбирать в соответствии с уровнем полноты безопасности. Альтернативные или эквивалентные методы/средства обозначают буквами, следующими за числом. Следует выполнять только один из альтернативных или эквивалентных методов/мероприятий. Выбор альтернативных методов должен быть обоснован в соответствии со свойствами, приведенными в приложении С, желательно для каждого применения.

Примечание 1 — Логические и функциональные блок-схемы и диаграммы последовательности действий приведены в [7].

Примечание 2 — См. таблицу С.17.

Примечание 3 — Ссылки (являющиеся справочными, а не обязательными) «B.x.x.x», «C.x.x.x» в столбце 2 указывают на подробные описания методов/средств, изложенных в приложениях В и С [5].

Таблица В.8 — Статический анализ (см. таблицу А.9 приложения А)

Метод/средство[1]	Ссылка	УПБ1	УПБ2	УПБ3	УПБ4
1 Анализ граничных значений	С.5.4	R	R	HR	HR
2 Таблица контрольных проверок	В.2.5	R	R	R	R
3 Анализ потоков управления	С.5.9	R	HR	HR	HR
4 Анализ потоков данных	С.5.10	R	HR	HR	HR
5 Предположение ошибок	С.5.5	R	R	R	R
6a Формальные проверки, включая конкретные критерии	С.5.14	R	R	HR	HR
6b Сквозной контроль (программного обеспечения)	С.5.15	R	R	R	R
7 Тестирование на символьном уровне	С.5.11	—	—	R	R
8 Анализ проекта	С.5.16	HR	HR	HR	HR
9 Статический анализ выполнения программы с ошибкой	В.2.2, С.2.4	R	R	R	HR
10 Временной анализ выполнения при наихудших условиях	С.5.20	R	R	R	R

[1] Методы/средства следует выбирать в соответствии с уровнем полноты безопасности. Альтернативные или эквивалентные методы/средства обозначают буквами, следующими за числом. Следует выполнять только один из альтернативных или эквивалентных методов/мероприятий. Выбор альтернативных методов должен быть обоснован в соответствии со свойствами, приведенными в приложении С, желательно для каждого применения.

Примечание 1 — См. таблицу С.18.

Примечание 2 — Ссылки (являющиеся справочными, а не обязательными) «B.x.x.x», «C.x.x.x» в столбце 2 указывают на подробные описания методов/средств, изложенных в приложениях В и С [5].

Таблица В.9 — Модульный подход (см. таблицу А.4 приложения А)

Метод/средство[1]	Ссылка	УПБ1	УПБ2	УПБ3	УПБ4
1 Ограничение размера программного модуля	С.2.9	HR	HR	HR	HR
2 Управление сложностью программного обеспечения	С.5.13	R	R	HR	HR
3 Ограничение доступа/инкапсуляция информации	С.2.8	R	HR	HR	HR
4 Ограниченное число параметров/фиксированное число параметров подпрограммы	С.2.9	R	R	R	R
5 Одна точка входа и одна точка выхода в каждой подпрограмме и функции	С.2.9	HR	HR	HR	HR
6 Полностью определенный интерфейс	С.2.9	HR	HR	HR	HR

Окончание таблицы В.9

[1] Методы/средства следует выбирать в соответствии с уровнем полноты безопасности. Использование одного метода является, по-видимому, недостаточным. Следует рассматривать все подходящие методы.

Примечание 1 — См. таблицу С.19.

Примечание 2 — Ссылки (являющиеся справочными, а не обязательными) «В.х.х.х», «С.х.х.х» в столбце 2 указывают на подробные описания методов/средств, изложенных в приложениях В и С [5].

Приложение С
(справочное)

Свойства стойкости к систематическим отказам программного обеспечения

С.1 Введение

Учитывая большое число факторов, влияющих на стойкость к систематическим отказам программного обеспечения, невозможно создать алгоритм, включающий методы и средства, который обеспечит необходимый результат для любого заданного применения. Целью настоящего приложения является:

- привести указания по выбору конкретных методов из приложений А и В, чтобы обеспечить стойкость к систематическим отказам программного обеспечения;
- помочь в обосновании использования методов, которые не перечислены явно в приложениях А и В.

Настоящее приложение является дополнением к приложениям А и В.

С.1.1 Структура приложения С, связанная с приложениями А и В

Выходные данные каждой стадии жизненного цикла программного обеспечения системы безопасности определены в таблице 1. Например, рассмотрена спецификация требований к программному обеспечению системы безопасности.

Таблица А.1 («Спецификация требований к программному обеспечению системы безопасности») приложения А рекомендует конкретные методы разработки спецификации требований к программному обеспечению системы безопасности:

Метод/средство	Ссылка	УПБ1	УПБ2	УПБ3	УПБ4
1a Полуформальные методы	Таблица В.7	R	R	HR	HR
1b Формальные методы	В.2.2, С.2.4	—	R	R	HR
2 Прямая прослеживаемость между требованиями к системе безопасности и требованиями к программному обеспечению системы безопасности	С.2.11	R	R	HR	HR
3 Обратная прослеживаемость между требованиями к системе безопасности и предполагаемыми потребностями безопасности	С.2.11	R	R	HR	HR
4 Компьютерные средства разработки спецификаций для поддержки, перечисленных выше, подходящих методов/средств	В.2.4	R	R	HR	HR

Таблица С.1 («Свойства систематической полноты безопасности — спецификация требований к программному обеспечению системы безопасности») устанавливает, что спецификация требований к программному обеспечению системы безопасности характеризуется следующими требуемыми свойствами (которые неформально определены в [5], см. приложение F):

Свойство					
Полнота охвата потребностей безопасности программным обеспечением	Корректность охвата потребностей безопасности программным обеспечением	Отсутствие ошибок в самой спецификации, включая отсутствие неоднозначности	Ясность требований к системе безопасности	Отсутствие неблагоприятного взаимовлияния функций, не связанных с безопасностью, и функций безопасности, реализуемых программным обеспечением системы безопасности	Способность обеспечения проведения оценки и подтверждения соответствия

Таблица С.1 с помощью неформальной шкалы R1/R2/R3 также ранжирует эффективность конкретных методов в достижении этих требуемых свойств:

Метод/ средство	Свойство					
	Полнота охвата потребностей безопасности программным обеспечением	Корректность охвата потребностей безопасности программным обеспечением	Отсутствие ошибок в самой спецификации, включая отсутствие неоднозначности	Ясность требований к системе безопасности	Отсутствие неблагоприятного взаимовлияния функций, не связанных с безопасностью, и функций безопасности, реализуемых программным обеспечением системы безопасности	Способность обеспечения проведения оценки и подтверждения соответствия
1а Полуформальные методы	R1 Дружественный или зависящий от предметной области метод спецификации и нотация, используемые специалистами в проблемной области	R1 Дружественный или зависящий от предметной области метод спецификации и нотация, используемые специалистами в проблемной области. R2 Верификация спецификации согласно критерию охвата	R1 Метод и нотация, которые помогают предотвратить или обнаружить внутреннюю несогласованность, отсутствующее поведение или математически несовместимые выражения. R2 Проверка спецификации согласно критериям охвата. R3 Проверка спецификации, основанная на систематическом анализе и/или систематическом предотвращении определенных типов отказов внутри спецификации	R1 Определяемая нотация, ограничивающая возможность для непонимания. R2 Применение пределов сложности в спецификации	—	R1 Определяемая нотация, снижающая неоднозначность в спецификации

Уверенность, которую можно связать со свойствами спецификации требований к программному обеспечению системы безопасности, как основание для обеспечения безопасности программного обеспечения, зависит от строгости методов, с помощью которых были достигнуты требуемые свойства спецификации требований к программному обеспечению системы безопасности. Строгость метода неформально ранжируется по шкале от R1 до R3, где R1 — наименее строгий и R3 — самый строгий метод.

R1	Без объективных критериев приемки или с ограниченными объективными критериями приемки. Например, тестирование методом «черного ящика», основанное на профессиональном суждении, полевые испытания
R2	С объективными критериями приемки, которые могут дать высокий уровень уверенности в том, что необходимое свойство достигнуто (исключения должны быть определены и обоснованы); например, тест или аналитические методы с метриками охвата, охват таблицами контрольных проверок
R3	С объективным, систематическим рассуждением о том, что необходимое свойство достигнуто. Например, формальное доказательство, демонстрирующее соблюдение архитектурных ограничений, которые гарантируют свойство
—	Данный метод не относится к этому свойству

Для каждого метода, обеспечивающего конкретное свойство, определяется одно из ранжированных значений R1/R2/R3 в зависимости от уровня строгости этого метода.

Метод/ средство	Свойство					
	Полнота охвата потребностей безопасности программным обеспечением	Корректность охвата потребностей безопасности программным обеспечением	Отсутствие ошибок в самой спецификации, включая отсутствие неоднозначности	Ясность требований к системе безопасности	Отсутствие неблагоприятного взаимовлияния функций, не связанных с безопасностью, и функций безопасности, реализуемых программным обеспечением системы безопасности	Способность обеспечения проведения оценки и подтверждения соответствия
1a Полуформальные методы				R1 Определяемая нотация, ограничивающая возможность для непонимания. R2 Применение пределов сложности в спецификации		

В этом примере полуформальный метод со строгостью R1 обеспечивает ограниченную нотацию, которая улучшает точность выражений, и со строгостью R2 еще более ограничивает сложность спецификации, что в противном случае могло бы привести к путанице.

С.1.2 Метод использования — 1

Руководящий принцип. Если можно убедительно продемонстрировать, что требуемые свойства были достигнуты при разработке спецификации требований к программному обеспечению системы безопасности, то обоснована уверенность в том, что спецификация требований к программному обеспечению системы безопасности является соответствующей основой для разработки программного обеспечения с достаточным уровнем систематической полноты безопасности.

Из таблицы С.1 видно, что каждый из методов по таблице А.1 приложения А в той или иной степени обычно обеспечивает выполнение одного или более из вышеупомянутых свойств по таблице С.1, которые относятся к спецификации требований к программному обеспечению системы безопасности.

Однако важно отметить, что хотя таблица А.1 рекомендует конкретные методы, эти рекомендации не являются нормативными и фактически приложение А ясно показывает, что «Учитывая большое число факторов, влияющих на стойкость к систематическим отказам программного обеспечения, невозможно создать алгоритм, включающий методы и средства, который обеспечит необходимый результат для любого заданного применения».

Практически методы, использующиеся при разработке спецификации требований к программному обеспечению системы безопасности, выбирают с учетом ряда практических ограничений (см. 7.1.2.7) в дополнение к собственным возможностям методов. Такие ограничения могут включать в себя:

- непротиворечивость и взаимодополняющий характер выбранных методов, языков и инструментов для всего цикла разработки;
- полностью ли понимают разработчики, используемые ими методы, языки и инструменты;
- насколько хорошо адаптированы методы, языки и инструменты к конкретным проблемам, для разработки которых они используются.

Таблица С.1 может использоваться для сравнения относительной эффективности конкретных методов по таблице А.1 в достижении требуемых свойств на стадии жизненного цикла «спецификация требований к программному обеспечению системы безопасности» и в то же время для факторизации практических ограничений конкретного разрабатываемого проекта.

Например, для верификации и подтверждения соответствия использование формального метода (R3) более обосновано, чем полуформального метода (R2), однако другие ограничения проекта (например, доступность сложных компьютерных инструментов поддержки или узкоспециализированное представление формальной нотации) могут повлиять на выбор полуформального подхода.

Таким образом, требуемые свойства таблицы С.1 могут служить основанием для аргументированного и практического сравнения альтернативных методов, которые рекомендованы в таблице А.1 и предназначены для разработки спецификации требований к программному обеспечению системы безопасности. Или (в общем случае) при рассмотрении требуемых свойств, перечисленных в соответствующей таблице приложения С для конкретной стадии жизненного цикла, может быть сделан аргументированный выбор метода из нескольких альтернативных, рекомендуемых приложением А.

Но особо следует обратить внимание на то, что вследствие систематической природы поведения, свойства приложения С, возможно, не будут достигнуты или строго доказаны. Скорее свойства являются целью, к которой необходимо стремиться. Достижение этих целей может даже потребовать компромиссов между различными свойствами, например, между проектом защиты и его простотой.

Наконец, в дополнение к определению критериев R1/R2/R3 полезно в качестве рекомендации сформировать неформальную связь между ростом уровня строгости от R1 к R3 и ростом уверенности в корректности программного обеспечения. Так как рекомендация является общей и неформальной, необходимо стремиться к следующим минимальным уровням строгости, когда приложение А требует соответствующий ему УПБ:

УПБ	Строгость R
1/2	R1
3	R2 (если возможно)
4	Наиболее высокая возможная строгость

С.1.3 Метод использования — 2

Хотя приложение А рекомендует конкретные методы, разрешено также применять другие методы и средства, если они соответствуют требованиям и целям стадии жизненного цикла.

Уже было отмечено, что на стойкость к систематическим отказам программного обеспечения влияет много факторов, и невозможно создать алгоритм для выбора и объединения методов, гарантирующий достижение требуемых свойств для любого заданного применения.

Может существовать несколько эффективных способов обеспечить требуемые свойства и следует признать, что разработчики системы могут быть в состоянии представить альтернативные доказательства. Информация в таблицах настоящего приложения может использоваться в качестве основы для весомого аргумента, чтобы обосновать выбор методов, отсутствующих в таблицах приложения А.

С.2 Свойства для систематической полноты безопасности

Руководящие указания, представленные в настоящем стандарте и [5], определяют конкретные методы обеспечения свойств систематической полноты безопасности и формирования убедительных доказательств. Если метод не способствует достижению свойства, то в таблицах настоящего приложения это показано как —. Если метод может отрицательно влиять на некоторые свойства и положительно — на другие, то о нем приводится пояснение в примечании к соответствующей таблице.

Таблица С.1 — Свойства систематической полноты безопасности — спецификация требований к программному обеспечению системы безопасности (см. 7.2. Упоминается в таблице А.1)

Метод/ средство	Свойство					
	Полнота охвата потребностей безопасности программным обеспечением	Корректность охвата потребностей безопасности программным обеспечением	Отсутствие ошибок в самой спецификации, включая отсутствие неоднозначности	Ясность требований к системе безопасности	Отсутствие неблагоприятного взаимовлияния функций, не связанных с безопасностью, и функций безопасности, реализуемых программным обеспечением системы безопасности	Способность обеспечения проведения оценки и подтверждения соответствия
1а Полуформальные методы	R1 Дружественный или зависящий от предметной области метод спецификации и нотация, используемая специалистами в проблемной области	R1 Дружественный или зависящий от предметной области метод спецификации и нотация, используемая специалистами в проблемной области. R2 Верификация спецификации согласно критерию охвата	R1 Метод и нотация, которые помогают предотвратить или обнаружить внутреннюю несогласованность, отсутствующее поведение или математически несовместимые выражения. R2 Проверка спецификации согласно критериям охвата. R3 Проверка спецификации, основанная на систематическом анализе и/или систематическом предотвращении определенных типов отказов внутри спецификации	R1 Определяемая нотация, ограничивающая возможность для непонимания. R2 Применение пределов сложности в спецификации	—	R2 Определяемая нотация, уменьшающая неоднозначность в спецификации

Продолжение таблицы С.1

Метод/ средство	Свойство					
	Полнота охвата потребностей безопасности программным обеспечением	Корректность охвата потребностей безопасности программным обеспечением	Отсутствие ошибок в самой спецификации, включая отсутствие неоднозначности	Ясность требований к системе безопасности	Отсутствие неблагоприятного взаимовлияния функций, не связанных с безопасностью, и функций безопасности, реализуемых программным обеспечением системы безопасности	Способность обеспечения проведения оценки и подтверждения соответствия
1b Формальные методы	R1 Дружественный или зависящий от предметной области метод спецификации и нотация, используемая специалистами в проблемной области	R1 Дружественный или зависящий от предметной области метод спецификации и нотация, используемая специалистами в проблемной области. R2 Верификация спецификации согласно критерию охвата. R3 Гарантия правильности на ограниченных аспектах поведения	R1 Метод и нотация, которые помогают предотвратить или обнаружить внутреннюю несогласованность, отсутствующее поведение или математически несовместимые выражения. R2 Проверка спецификации согласно критериям охвата. R3 Проверка спецификации, основанная на систематическом анализе и/или систематическом предотвращении определенных типов отказов внутри спецификации	— Примечание — Может усложнить достижение этого свойства, если метод не является дружественным для приложения или не зависящим от предметной области	—	R3 Уменьшает неоднозначность в спецификации
2 Прямая прослеживаемость между спецификацией требований к безопасности системы и требованиями к безопасности программного обеспечения	R1 Уверенность в том, что спецификация требований к безопасности программного обеспечения получает системные требования к безопасности	—	—	—	—	—

Окончание таблицы С.1

Метод/ средство	Свойство					
	Полнота охвата потребностей безопасности программным обеспечением	Корректность охвата потребностей безопасности программным обеспечением	Отсутствие ошибок в самой спецификации, включая отсутствие неоднозначности	Ясность требований к системе безопасности	Отсутствие неблагоприятного взаимовлияния функций, не связанных с безопасностью, и функций безопасности, реализуемых программным обеспечением системы безопасности	Способность обеспечения проведения оценки и подтверждения соответствия
3 Обратная прослеживаемость между требованиями к системе безопасности и предполагаемыми потребностями в безопасности	—	R1 Уверенность в том, что спецификация требований к программному обеспечению системы безопасности не содержит ненужной сложности	—	R1 Прослеживаемость потребностей безопасности управляемого оборудования улучшает понятность	R1	R1
4 Компьютерные средства разработки спецификаций для поддержки, перечисленных выше, подходящих методов/ средств	R1 Инкапсуляция знаний проблемной области УО и программной среды. R2 Если перечень вопросов, которые необходимо учесть в таблице контрольных проверок, определен, обоснован и охватывает проблему	R1 Методы функционального моделирования. R2 Функциональное моделирование в соответствии с определенными и обоснованными критериями охвата	R2 Синтаксическая и семантическая проверки, чтобы убедиться в том, что соответствующие правила выполнены	R1 Анимация или просмотр спецификации	R1 Идентификация связанных и несвязанных с безопасностью функций	R1 Помощь в прослеживаемости и в охвате. R2 Измерение прослеживаемости и охвата

Таблица С.2 — Свойства систематической полноты безопасности — проектирование и разработка программного обеспечения — проектирование архитектуры программ (см. 7.4.3 и таблицу А.2 приложения А)

Метод/средство	Свойство							
	Полнота спецификации требований к программному обеспечению системы безопасности	Корректность спецификации требований к программному обеспечению системы безопасности	Отсутствие собственных ошибок проекта	Простота и ясность	Предсказуемость поведения	Верифицируемость и тестируемость проекта	Отказоустойчивость	Защита от отказов по общей причине, вызванной внешним событием
1 Обнаружение ошибок	—	—	—	— Достижение этого свойства может быть сложным	R1 Контроль логики выполнения программы	—	R1 (R2, если цели охвата определены, обоснованы и выполнены)	R1 или —
2 Коды обнаружения ошибок	—	—	—	— Достижение этого свойства может быть сложным	— Достижение этого свойства может быть сложным	—	R1 (R2, если цели охвата определены, обоснованы и выполнены). Эффективен для конкретных областей применения, например, для передачи данных	R1 Эффективен для конкретных областей применения, например, для передачи данных
3а Программирование с проверкой ошибок	—	R2 С помощью постусловий проверяется соответствие детальным требованиям	—	R2 Предусловия ограничивают входное пространство	R2 Постусловия проверяют выходы, которые должны быть ожидаемыми либо приемлемыми	R2 Предусловия ограничивают входное пространство и, следовательно, необходимое тестируемое пространство	R3 Эффективен для конкретных отказов	R3 Эффективен для конкретных отказов

Продолжение таблицы С.2

Метод/средство	Свойство							
	Полнота спецификации требований к программному обеспечению системы безопасности	Корректность спецификации требований к программному обеспечению системы безопасности	Отсутствие собственных ошибок проекта	Простота и ясность	Предсказуемость поведения	Верифицируемость и тестируемость проекта	Отказоустойчивость	Защита от отказов по общей причине, вызванной внешним событием
3b Методы контроля (при реализации процесса контроля и контролируемой функции на одном компьютере обеспечивается их независимость)	—	—	R2 Независимый контроль реализует минимальные требования безопасности	R2 Независимый контроль обеспечивает неявное разнообразие	R2 Независимый контроль простым способом реализует лишь минимальные требования безопасности	R2 Независимый контроль реализует лишь минимальные требования безопасности	R1 (R2, если цели охвата определены, обоснованы и выполнены)	R1 (R2, если цели охвата определены, обоснованы и выполнены)
3c Методы контроля (реализация процесса контроля и контролируемой функции на разных компьютерах)	—	—	R2 Независимый контроль реализует минимальные требования безопасности	R2 Независимый контроль обеспечивает неявное разнообразие	R2 Независимый контроль простым способом реализует лишь минимальные требования безопасности	R2 Независимый контроль реализует лишь минимальные требования безопасности	R1 (R2, если цели охвата определены, обоснованы и выполнены)	R1 (R2, если цели охвата определены, обоснованы и выполнены)

Продолжение таблицы С.2

Метод/средство	Свойство							
	Полнота спецификации требований к программному обеспечению системы безопасности	Корректность спецификации требований к программному обеспечению системы безопасности	Отсутствие собственных ошибок проекта	Простота и ясность	Предсказуемость поведения	Верифицируемость и тестируемость проекта	Отказоустойчивость	Защита от отказов по общей причине, вызванной внешним событием
3d Многовариантное программирование, реализующее одну спецификацию требований к программному обеспечению системы безопасности	—	—	R1	— Примечание — Достижение этого свойства может быть сложным, если реализуется в одной исполнимой программе	—	—	R1 Если отказ одной программы не оказывает негативное влияние на другие. R2 R2, если цели охвата определены, обоснованы и выполнены. Не защищает от отказов в спецификации требований	R1 Если отказ одной программы не оказывает негативное влияние на другие. R2 R2, если цели охвата определены, обоснованы и выполнены. Не защищает от отказов в спецификации требований

Продолжение таблицы С.2

Метод/средство	Свойство							
	Полнота спецификации требований к программному обеспечению системы безопасности	Корректность спецификации требований к программному обеспечению системы безопасности	Отсутствие собственных ошибок проекта	Простота и ясность	Предсказуемость поведения	Верифицируемость и тестируемость проекта	Отказоустойчивость	Защита от отказов по общей причине, вызванной внешним событием
3e Функционально многовариантное программирование, реализующее различные спецификации требований к программному обеспечению системы безопасности. Обычно это требуется, если применяются датчики, работающие на различных физических принципах	—	—	R1	— Примечание — Достижение этого свойства может быть сложным, если реализуется в одной исполнимой программе	—	—	R1 Если отказ одной программы не оказывает негативное влияние на другие	R1 Если отказ одной программы не оказывает негативное влияние на другие. Защищает от отказов в спецификации требований
3f Восстановление предыдущего состояния	—	—	— Примечание — Достижение этого свойства может быть сложным	—	— Примечание — Достижение этого свойства может быть сложным	—	R2	R1 (R2, если цели охвата определены, обоснованы и выполнены)

Продолжение таблицы С.2

Метод/средство	Свойство							
	Полнота спецификации требований к программному обеспечению системы безопасности	Корректность спецификации требований к программному обеспечению системы безопасности	Отсутствие собственных ошибок проекта	Простота и ясность	Предсказуемость поведения	Верифицируемость и тестируемость проекта	Отказоустойчивость	Защита от отказов по общей причине, вызванной внешним событием
3g Проектирование программного обеспечения, не сохраняющего состояние (или проектирование программного обеспечения, сохраняющего ограниченное описание состояния)	R2 Если несохранение или сохранение ограниченного описания состояния предусмотрено в требованиях к системе безопасности	R2 Если несохранение или сохранение ограниченного описания состояния предусмотрено в требованиях к системе безопасности	R2 Если несохранение или сохранение ограниченного описания состояния предусмотрено в требованиях к системе безопасности	R1 R2 Если определены, обоснованы и выполнены ограничения для определенного возможного числа состояний	R1 R2 Если определены, обоснованы и выполнены ограничения для определенного возможного числа состояний	R1 R2 Если определены, обоснованы и выполнены цели верификации/тестового охвата возможных состояний	R1 Если это приводит к самовосстановлению проекта. R2 Если определены, обоснованы и выполнены цели самовосстановления	R1 Если это приводит к самовосстановлению проекта. R2 Если определены, обоснованы и выполнены цели самовосстановления
4a Механизмы повторных попыток парирования сбоя	—	—	—	—	Достижение этого свойства может быть сложным	—	R1 (R2, если цели охвата определены, обоснованы и выполнены)	R1 (R2, если цели охвата определены, обоснованы и выполнены)
4b Постепенное отключение функций	—	—	— Примечание — Достижение этого свойства может быть сложным	—	—	—	R1 R2, если цели охвата определены, обоснованы и выполнены	R1 R2, если цели охвата определены, обоснованы и выполнены

Продолжение таблицы С.2

Метод/средство	Свойство							
	Полнота спецификации требований к программному обеспечению системы безопасности	Корректность спецификации требований к программному обеспечению системы безопасности	Отсутствие собственных ошибок проекта	Простота и ясность	Предсказуемость поведения	Верифицируемость и тестируемость проекта	Отказоустойчивость	Защита от отказов по общей причине, вызванной внешним событием
5 Исправление ошибок методами искусственного интеллекта	—	— Примечание — Достижение этого свойства может быть сложным	— Примечание — Достижение этого свойства может быть сложным	— Примечание — Достижение этого свойства может быть сложным	R1 Примечание — Достижение этого свойства может быть сложным	— Примечание — Достижение этого свойства может быть сложным	—	—
6 Динамическая реконфигурация	—	— Примечание — Достижение этого свойства может быть сложным	— Примечание — Достижение этого свойства может быть сложным	— Примечание — Достижение этого свойства может быть сложным	— Примечание — Достижение этого свойства может быть сложным	— Примечание — Достижение этого свойства может быть сложным	—	—

Продолжение таблицы С.2

Метод/средство	Свойство							
	Полнота спецификации требований к программному обеспечению системы безопасности	Корректность спецификации требований к программному обеспечению системы безопасности	Отсутствие собственных ошибок проекта	Простота и ясность	Предсказуемость поведения	Верифицируемость и тестируемость проекта	Отказоустойчивость	Защита от отказов по общей причине, вызванной внешним событием
7 Модульный подход	—	R1 R2, если цели декомпозиции определены, обоснованы и выполнены. В противном случае только R1	R1 Если отсутствие конкретных типов собственных ошибок в проекте может быть верифицировано независимо для каждого модуля. R3 Если отсутствие конкретных типов собственных ошибок в проекте может быть строго обосновано при проектировании модуля	R1 R2, если цели декомпозиции определены, обоснованы и выполнены	R1 R2, если цели декомпозиции определены, обоснованы и выполнены	R1 R2, если цели декомпозиции определены, обоснованы и выполнены	R1 Если модули, на которые не влияет отказ модуля, способствуют ослаблению отказа/ восстановлению отказавшего модуля. R3 Если появление конкретных отказов может быть строго обосновано	R1 Если модули, на которые могут влиять внешние события, и которые могут повлиять на несколько каналов одновременно, выявлены и полностью проверены. R3 Если появление конкретных внешних событий может быть строго обосновано

Продолжение таблицы С.2

Метод/средство	Свойство							
	Полнота спецификации требований к программному обеспечению системы безопасности	Корректность спецификации требований к программному обеспечению системы безопасности	Отсутствие собственных ошибок проекта	Простота и ясность	Предсказуемость поведения	Верифицируемость и тестируемость проекта	Отказоустойчивость	Защита от отказов по общей причине, вызванной внешним событием
8 Использование доверительных/проверенных элементов программного обеспечения (при наличии)	—	R1 R2 R3 Если элемент вносит существенный вклад в выполнение требований к системе безопасности и правильно используется	R1 R2 R3 Элементы, проверенные на практике. Такая способность каждого элемента должна быть обоснована	R1 Модульный подход декомпозирует полную сложность на хорошо понятные блоки	R1 R2 R3 Элементы, проверенные на практике	—	R1 R2, если возможности отказоустойчивости незамедлительно предоставляются элементом и правильно используются или если вокруг элемента построен уровень защиты	R1 R2, если защита от внешних событий, которые могут повлиять на несколько каналов одновременно, незамедлительно предоставляется элементом и правильно используется или если вокруг элемента построен уровень защиты
9 Прямая прослеживаемость между спецификацией требований к программному обеспечению системы безопасности и архитектурой программного обеспечения	R1 Уверенность в том, что архитектура соответствует требованиям к программному обеспечению системы безопасности	—	—	—	—	—	—	—

Продолжение таблицы С.2

Метод/средство	Свойство							
	Полнота спецификации требований к программному обеспечению системы безопасности	Корректность спецификации требований к программному обеспечению системы безопасности	Отсутствие собственных ошибок проекта	Простота и ясность	Предсказуемость поведения	Верифицируемость и тестируемость проекта	Отказоустойчивость	Защита от отказов по общей причине, вызванной внешним событием
10 Обратная прослеживаемость между архитектурой программного обеспечения и спецификацией требований к программному обеспечению системы безопасности	—	R1 Уверенность в том, что архитектура излишне не усложнена	—	—	—	—	—	—
11a Методы структурных диаграмм	—	R1	—	R1 (Графические описания легче понять)	—	R1 (Структурируемые проекты легче верифицировать и тестировать)	—	—
11b Полуформальные методы	R1 Дружественные или зависящие от предметной области методы спецификации и нотация	R1 Дружественные или зависящие от предметной области методы спецификации и нотация	R2 Может выявить внутреннюю несогласованность или неработоспособность или математически некорректные выражения	—	R2 (Предоставляет свидетельства для предсказуемости)	R2 (Предоставляет свидетельства для внутренней согласованности модели проекта)	—	—

Продолжение таблицы С.2

Метод/средство	Свойство							
	Полнота спецификации требований к программному обеспечению системы безопасности	Корректность спецификации требований к программному обеспечению системы безопасности	Отсутствие собственных ошибок проекта	Простота и ясность	Предсказуемость поведения	Верифицируемость и тестируемость проекта	Отказоустойчивость	Защита от отказов по общей причине, вызванной внешним событием
11c Методы формального проектирования и усовершенствования	R1 Дружественные или зависящие от предметной области метод спецификации и нотация	R1 Обеспечивает точное определение ограниченных аспектов поведения, которые должны соответствовать предметной области	R3 Может выявить внутреннюю несогласованность или неработоспособность или математически некорректные выражения	— Примечание — Достижение этого свойства может быть сложным	R2 Предоставляет доказательства для предсказуемости	R2	—	—

Продолжение таблицы С.2

Метод/средство	Свойство							
	Полнота спецификации требований к программному обеспечению системы безопасности	Корректность спецификации требований к программному обеспечению системы безопасности	Отсутствие собственных ошибок проекта	Простота и ясность	Предсказуемость поведения	Верифицируемость и тестируемость проекта	Отказоустойчивость	Защита от отказов по общей причине, вызванной внешним событием
11d Автоматическая генерация программного обеспечения	R1 Если исполнимое программное обеспечение автоматически сгенерировано в соответствии со спецификацией требований или проектом, который, как было показано, является корректным. R2 Если у инструментов генерации, как было показано, есть соответствующая история использования	R1 Если исполнимое программное обеспечение автоматически сгенерировано в соответствии со спецификацией требований или проектом, который, как было показано, является полным. R2 Если у инструментов генерации, как было показано, есть соответствующая история использования	R1 Если инструменты генерации гарантируют предотвращение определенных внутренних отказов проекта. R2 Если у инструментов генерации, как было показано, есть соответствующая история использования	—	—	—	R1, R2, R3 Если способность к отказоустойчивости генерируется автоматически	—

Продолжение таблицы С.2

Метод/средство	Свойство							
	Полнота спецификации требований к программному обеспечению системы безопасности	Корректность спецификации требований к программному обеспечению системы безопасности	Отсутствие собственных ошибок проекта	Простота и ясность	Предсказуемость поведения	Верифицируемость и тестируемость проекта	Отказоустойчивость	Защита от отказов по общей причине, вызванной внешним событием
12 Компьютерные средства разработки спецификаций и проектирования	R1 Инкапсуляция знаний проблемной области управляемого оборудования и программной среды R2 Если определена, обоснована и охвачена таблица контрольных проверок, а также учтены ее проблемы	R1 Осуществление обратной прослеживаемости требований. Функциональные методы моделирования. R2 Функциональное моделирование согласно определенным и обоснованным критериям охвата	R2 Семантические и синтаксические проверки, для гарантирования того, что соответствующие правила выполнены	R1 Анимация и просмотр	—	R2 Семантические и синтаксические проверки, чтобы гарантировать, что соответствующие правила выполнены	—	—

Продолжение таблицы С.2

Метод/средство	Свойство							
	Полнота спецификации требований к программному обеспечению системы безопасности	Корректность спецификации требований к программному обеспечению системы безопасности	Отсутствие собственных ошибок проекта	Простота и ясность	Предсказуемость поведения	Верифицируемость и тестируемость проекта	Отказоустойчивость	Защита от отказов по общей причине, вызванной внешним событием
13a Циклическое поведение с гарантированным максимальным временем цикла	—	R1 Для вопросов синхронизации в спецификации. R3 Если максимальное время цикла определено строгим выводом	R1 Для вопросов синхронизации в спецификации. R3 Если максимальное время цикла определено строгим выводом	—	R1 Для вопросов синхронизации в спецификации. R3 Если максимальное время цикла определено строгим выводом	R1 Для вопросов синхронизации в спецификации. R3 Если максимальное время цикла определено строгим выводом	—	—
13b Архитектура с временны́м распределением	R3 Полнота гарантируется распределением (только для свойств синхронизации)	R3 Корректность гарантируется распределением (только для свойств синхронизации)	R3 Строгая гарантия от внутренних отказов синхронизации	R1 Определяемая нотация значительно снижает непонимание, в основе подхода — использование предсказуемости	R3 Неблагоприятное влияние: полное разделение во времени, поэтому влияния отсутствуют	R3 Существенное сокращение усилий, необходимых для тестирования и сертификации системы	R2 Прозрачная реализация отказоустойчивости	R3 Внешние прерывания не могут вмешаться в расписание с распределенным временем, приоритетом для которого являются задачи, критические к безопасности

Окончание таблицы С.2

Метод/средство	Свойство							
	Полнота спецификации требований к программному обеспечению системы безопасности	Корректность спецификации требований к программному обеспечению системы безопасности	Отсутствие собственных ошибок проекта	Простота и ясность	Предсказуемость поведения	Верифицируемость и тестируемость проекта	Отказоустойчивость	Защита от отказов по общей причине, вызванной внешним событием
13c Управление событиями с гарантированным максимальным временем реакции	—	—	—	R1 Архитектуры, управляемые событиями, могут быть непонятными	R2 Архитектуры, управляемые событиями, могут быть непонятными	R1 Делает тестирование более предсказуемым	—	—
14 Статическое выделение ресурсов	R1	R1	R1	R1 Делает проектирование более понятным	R2 С архитектурой, определяющей использование ресурсов	R1 Делает тестирование более предсказуемым	—	—
15 Статическая синхронизация доступа к разделяемым ресурсам	—	R1 Обеспечивает предсказуемость при доступе к ресурсам	R1 R3, если поддерживается строгими выводами, обеспечивающими корректность синхронизации	R1 Делает проектирование более понятным	R1 R3, если поддерживается строгими выводами, обеспечивающими корректность синхронизации	—	—	—

Таблица С.3 — Свойства систематической полноты безопасности — проектирование и разработка программного обеспечения — инструментальные средства поддержки и языки программирования (см. 7.4.4 и таблицу А.3 приложения А)

Метод / средство	Свойство		
	Поддержка разработки программного обеспечения с требуемыми свойствами программного обеспечения	Четкость работы и функциональность инструментальных средств	Корректность и воспроизводимость результата
1 Выбор соответствующего языка программирования	R2 Если строгая типизация, ограниченное преобразование типов. R3 Если определены семантики для формального вывода	—	—
2 Строго типизированные языки программирования	R2	—	—
3 Подмножество языка	R2 В зависимости от выбранного подмножества	R1	R2 В зависимости от выбранного подмножества
4a Сертифицированные средства и сертифицированные трансляторы	—	R2	R2
4b Инструментальные средства, заслуживающие доверия на основании опыта использования	R1 Если класс обнаруживаемых программных ошибок определяется систематически. R2 Если существует объективное доказательство подтверждения соответствия эффективности инструментального средства	R1 Если инструментальное средство поддержки не является специальным для данной проблемной области. R2 Если инструментальное средство поддержки было создано специально для данной проблемной области	R1 R2 Если существует объективное доказательство подтверждения соответствия эффективности инструментального средства, например, набор средств для подтверждения соответствия компиляторов

Таблица С.4 — Свойства систематической полноты безопасности — проектирование и разработка программного обеспечения — детальное проектирование (включает в себя проектирование систем программного обеспечения, проектирование модулей программного обеспечения и кодирование) (см. 7.4.4, 7.4.6 и таблицу А.4 приложения А)

Метод/ средство	Свойство							
	Полнота спецификации требований к программному обеспечению системы безопасности	Корректность спецификации требований к безопасности программного обеспечения	Отсутствие собственных ошибок проекта	Простота и ясность	Предсказуемость поведения	Верифицируемость и тестируемость проекта	Отказоустойчивость	Отсутствие отказов по общей причине
1a Методы структурных диаграмм	R2	R1	R1	—	—	R1 Структурированные проекты более готовы к верификации и тестированию	—	—
1b Полуформальные методы	R2	R2	R2	—	R2	R2	—	—
1c Методы формального проектирования и усовершенствования	—	R3	R3	— Примечание — Достижение этого свойства может быть сложным	R3 Предоставляет свидетельства для предсказуемости	R2	—	—
2 Средства автоматизированного проектирования	R2 Автоматизированное средство разработки спецификации, применяющее семантические и синтаксические проверки и гарантирующее, что соответствующие правила удовлетворены	R1	R2 Автоматизированное средство разработки спецификации, применяющее семантические и синтаксические проверки и гарантирующее, что соответствующие правила удовлетворены	—	—	R2 Основанные на CASE технологии средства поддержки тестового охвата и статической проверки	—	—

Продолжение таблицы С.4

Метод/ средство	Свойство							
	Полнота спецификации требований к программному обеспечению системы безопасности	Корректность спецификации требований к безопасности программного обеспечения	Отсутствие собственных ошибок проекта	Простота и ясность	Предсказуемость поведения	Верифицируемость и тестируемость проекта	Отказоустойчивость	Отсутствие отказов по общей причине
3 Программирование с защитой	—	—	—	— Примечание — Достижение этого свойства может быть сложным	— Примечание — Достижение этого свойства может быть сложным	—	R1 (R2, если цели охвата определены, обоснованы и выполнены)	R1 (R2, если цели охвата определены, обоснованы и выполнены)
4 Модульный подход	—	—	R1	R1	R1	R1	—	—
5 Стандарты для проектирования и кодирования	—	—	R1	R1	R1	R1	—	—
6 Структурное программирование	—	R1	R1	R1	R1	R1	—	—

Окончание таблицы С.4

Метод/ средство	Свойство							
	Полнота спецификации требований к программному обеспечению системы безопасности	Корректность спецификации требований к безопасности программного обеспечения	Отсутствие собственных ошибок проекта	Простота и ясность	Предсказуемость поведения	Верифицируемость и тестируемость проекта	Отказоустойчивость	Отсутствие отказов по общей причине
7 Использование проверенных/верифицированных программных модулей и компонентов (по возможности)	—	—	R1 Элеметы, проверенные на практике	R1 Модульный подход декомпозирует сложность всей системы на хорошо понятные блоки	R1 Поведение элемента уже известно	—	—	—
8 Прямая прослеживаемость между спецификацией требований к программному обеспечению системы безопасности и проектом программного обеспечения	R1 Уверенность в том, что проект соответствует требованиям к программному обеспечению системы безопасности	—	—	—	—	—	—	—

Таблица С.5 — Свойства систематической полноты безопасности — проектирование и разработка программного обеспечения — тестирование и интеграция программных модулей (см. 7.4.7, 7.4.8. и таблицу А.5 приложения А)

Метод/средство	Свойство			
	Полнота тестирования и интеграции в соответствии со спецификацией проекта программного обеспечения	Корректность тестирования и интеграции в соответствии со спецификацией проекта программного обеспечения (успешное выполнение)	Воспроизводимость	Точно определенная тестируемая конфигурация
1 Вероятностное тестирование	R1 (R2, если цели охвата набором тестовых данных, отражающих реальные условия функционирования программы, определены, обоснованы и выполнены)	R1 (R2, если необходимые результаты определены, обоснованы и выполнены)	—	—
2 Динамический анализ и тестирование	R1 (R2, если цели охвата структурированием определены, обоснованы и выполнены)	R1 (R2, если необходимые результаты определены, обоснованы и выполнены)	—	—
3 Регистрация и анализ данных	—	R1	R1 Способствует согласованности процедур тестирования	R2 Если журналы записей об отказах/тестах включают в себя подробную информацию о серии базовых программ
4 Функциональное тестирование и тестирование методом «черного ящика»	R1 (R2, если цели охвата набором тестовых данных, отражающих реальные условия функционирования программы, определены, обоснованы и выполнены)	R1 (R2, если необходимые результаты определены, обоснованы и выполнены)	—	—
5 Тестирование рабочих характеристик	—	R1 (R2, если необходимые результаты определены, обоснованы и выполнены)	—	—

Продолжение таблицы С.5

Метод/средство	Свойство			
	Полнота тестирования и интеграции в соответствии со спецификацией проекта программного обеспечения	Корректность тестирования и интеграции в соответствии со спецификацией проекта программного обеспечения (успешное выполнение)	Воспроизводимость	Точно определенная тестируемая конфигурация
6 Тестирование, основанное на модели (MBT)	R2 MBT позволяет на ранних стадиях выявлять неоднозначности в спецификации и проекте. MBT используется, начиная с требований. R3 Если при моделировании используются формальные выводы и генерация тестовых примеров (TCG)	R2 Оценка результатов и комплектов регрессионных тестов — ключевое преимущество MBT. R3 Если применять формальные модели, то можно получить объективные данные по охвату тестами	R3 MBT (с TCG) направлен на автоматическое выполнение сгенерированных тестов	R2 MBT работает автоматически, тестируемая конфигурация должна быть точно определена; выполнение сгенерированных тестов подобно тестированию методом «черного ящика» с возможностью измерения уровня охвата исходного кода
7 Тестирование интерфейса	—	R1 (R2, если необходимые выходы определены, обоснованы и выполнены)	—	—
8 Управление тестированием и средства автоматизации	R1 (R2, если цели охвата тестами определены, обоснованы и выполнены)	—	R1 Автоматизация способствует согласованности	R2 Обеспечивает воспроизводимость тестирования
9 Прямая прослеживаемость между спецификацией проекта программного обеспечения и спецификациями тестирования модуля и интеграции	R1 Уверенность в том, что спецификация тестов соответствует требованиям к программному обеспечению системы безопасности	—	—	R2 Уверенность в четкой основе тестируемых требований

Окончание таблицы С.5

Метод/средство	Свойство			
	Полнота тестирования и интеграции в соответствии со спецификацией проекта программного обеспечения	Корректность тестирования и интеграции в соответствии со спецификацией проекта программного обеспечения (успешное выполнение)	Воспроизводимость	Точно определенная тестируемая конфигурация
10 Формальная верификация	R3 Если для создания тестовых примеров используется формальный вывод для того, чтобы показать, что все аспекты проекта были реализованы	R3 Дает объективные данные о выполнении всех требований к программному обеспечению системы безопасности	R1 Если средства поддержки недоступны. R2 Если инструмент поддерживается	—

Таблица С.6 — Свойства систематической полноты безопасности — интеграция программируемых электронных устройств (программное обеспечение и аппаратные средства) (см. 7.5 и таблицу А.6 приложения А)

Метод/средство	Свойство			
	Полнота интеграции в соответствии со спецификацией проекта	Корректность интеграции в соответствии со спецификацией проекта (успешное выполнение)	Воспроизводимость	Точно определенная конфигурация интеграции
1 Функциональное тестирование и тестирование методом «черного ящика»	R1 (R2, если цели охвата набором тестовых данных, отражающих реальные условия функционирования программы, определены, обоснованы и выполнены)	R1 (R2, если необходимые результаты определены, обоснованы и выполнены)	—	—
2 Тестирование выполнения	—	R1 (R2, если необходимые результаты определены, обоснованы и выполнены)	—	—
3 Прямая прослеживаемость между требованиями проектирования системы и программного обеспечения к интеграции программных и аппаратных средств и спецификациями тестирования интеграции программных и аппаратных средств	R1 Уверенность в том, что спецификации тестирования интеграции программных и аппаратных средств соответствуют требованиям интеграции	—	—	R2 Уверенность в четкой основе тестируемых требований

Таблица С.7 — Свойства систематической полноты безопасности — подтверждение соответствия для аспектов программного обеспечения безопасности системы (см. 7.7 и таблицу А.7 приложения А)

Метод/средство	Свойство			
	Полнота интеграции в соответствии со спецификацией проекта	Корректность интеграции в соответствии со спецификацией проекта (успешное выполнение)	Воспроизводимость	Точно определенная конфигурация подтверждения соответствия
1 Вероятностное тестирование	R1 (R2, если цели охвата набором тестовых данных, отражающих реальные условия функционирования программы, определены, обоснованы и выполнены)	R1 (R2, если необходимые результаты определены, обоснованы и выполнены)	—	—
2 Моделирование процесса	R1	R1 (R2, если необходимые результаты определены, обоснованы и выполнены)	—	R2 Дает определение внешнего окружения
3 Функциональное тестирование и тестирование методом «черного ящика»	R1 (R2, если цели охвата набором тестовых данных, отражающих реальные условия функционирования программы, определены, обоснованы и выполнены)	R1 (R2, если необходимые результаты определены, обоснованы и выполнены)	—	—
4 Прямая прослеживаемость между спецификацией требований к программному обеспечению системы безопасности и планом подтверждения соответствия программного обеспечения системы безопасности	R1 Уверенность в том, что план подтверждения соответствия программного обеспечения системы безопасности охватывает требования к программному обеспечению системы безопасности	—	—	R2 Уверенность в четкой основе тестируемых требований
5 Обратная прослеживаемость между планом подтверждения соответствия программного обеспечения системы безопасности и спецификацией требований к программному обеспечению системы безопасности	—	R1 Уверенность в том, что план подтверждения соответствия программного обеспечения системы безопасности не содержит излишней сложности	—	R2 Уверенность в четкой основе тестируемых требований

Таблица С.8 — Свойства систематической полноты безопасности — модификация программного обеспечения (см. 7.8 и таблицу А.8 приложения А)

Метод/средство	Свойство					
	Полнота модификации в соответствии с требованиями к модификации	Корректность модификации в соответствии с требованиями к модификации	Отсутствие собственных ошибок проекта	Предотвращение нежелательного поведения	Верифицируемость и тестируемость проекта	Регрессионное тестирование и охват проверкой
1 Анализ влияния	—	—	—	R1	R1	R1
2 Повторная верификация измененных программных модулей	R1 (R2, если существуют объективные цели проверки)	R1 (R2, если существуют объективные цели проверки)	R1 (R2, если существуют объективные цели проверки)	—	—	R1 R2 (R2, если существуют объективные цели проверки)
3 Повторная верификация программных модулей, на которые оказывают влияние изменения в других модулях	R1 (R2, если существуют объективные цели проверки)	R1 (R2, если существуют объективные цели проверки)	R1 (R2, если существуют объективные цели проверки)	—	—	R1 (R2, если существуют объективные цели проверки)
4a Повторная верификация системы в целом	R1 (R2, если существуют объективные цели проверки)	R1 (R2, если существуют объективные цели проверки)	—	R1 (R2, если существуют объективные цели проверки)	—	R1 (R2, если существуют объективные цели проверки)
4b Регрессионное подтверждение соответствия	R1 (R2, если существуют объективные цели проверки)	R1 (R2, если существуют объективные цели проверки)	—	R1 (R2, если существуют объективные цели проверки)	—	R1 (R2, если существуют объективные цели проверки)
5 Управление конфигурацией программного обеспечения	—	—	—	—	—	R1
6 Регистрация и анализ данных	R1	R1	—	—	—	—

Окончание таблицы С.8

Метод/средство	Свойство					
	Полнота модификации в соответствии с требованиями к модификации	Корректность модификации в соответствии с требованиями к модификации	Отсутствие собственных ошибок проекта	Предотвращение нежелательного поведения	Верифицируемость и тестируемость проекта	Регрессионное тестирование и охват проверкой
7 Прямая прослеживаемость между спецификацией требований к программному обеспечению системы безопасности и планом модификации программного обеспечения (включая повторные верификацию и подтверждение соответствия)	R1 Уверенность в том, что план модификации программного обеспечения (включая повторные верификацию и подтверждение соответствия) соответствует требованиям к программному обеспечению системы безопасности	—	—	—	—	—
8 Обратная прослеживаемость между планом модификации программного обеспечения (включая повторные верификацию и подтверждение соответствия) и спецификацией требований к программному обеспечению системы безопасности	—	R1 Уверенность в том, что план модификции программного обеспечения (включая повторные верификацию и подтверждение соответствия) не содержит излишней сложности	—	—	—	—

Таблица С.9 — Свойства систематической полноты безопасности — верификация программного обеспечения (см. 7.9 и таблицу А.9 приложения А)

Метод/средство	Свойство			
	Полнота верификации в соответствии с предыдущей стадией	Корректность верификации в соответствии с предыдущей стадией (успешное выполнение)	Воспроизводимость	Верификация точно определенной конфигурации
1 Формальное доказательство	—	R3	—	—
2 Анимация спецификации и тестирования	R1	R1	—	—
3 Статический анализ	—	R1/R2/R3 (Строгость может меняться от возможностей подмножества языка до формального анализа)	—	—
4 Динамический анализ и тестирование	R1 (R2, если цели охвата структурированием определены, обоснованы и выполнены)	R1 (R2, если необходимые результаты определены, обоснованы и выполнены)	—	—
5 Прямая прослеживаемость между спецификацией проекта программного обеспечения и планом верификации (включающим в себя верификацию данных) программного обеспечения	R1 Уверенность в том, что план верификации программного обеспечения (включающий верификацию данных) соответствует требованиям к программному обеспечению системы безопасности	—	—	R2 Уверенность в четкой основе тестируемых требований
6 Обратная прослеживаемость между планом верификации (включающим верификацию данных) программного обеспечения и спецификацией проекта программного обеспечения	—	R1 Уверенность в том, что план верификации программного обеспечения (включающий верификацию данных) не содержит излишней сложности	—	R2 Уверенность в четкой основе тестируемых требований
7 Численный анализ в автономном режиме	—	R1 Высокая уверенность в ожидаемой точности вычислений. (R2 при объективных критериях приемки. R3, если используется вместе с объективным систематическим доказательством, обосновывающим критерии приемки)	—	—

Таблица С.10 — Свойства систематической полноты безопасности — оценка функциональной безопасности (см. 8 и таблицу А.10 приложения А)

Метод/средство	Свойство						
	Полнота оценки функциональной безопасности в соответствии с требованиями настоящего стандарта	Корректность оценки функциональной безопасности в соответствии с проектными спецификациями (успешное выполнение)	Доступное для анализа решение всех выявленных проблем	Возможность модификации оценки функциональной безопасности после изменения проекта без необходимости проведения серьезной переработки оценки	Воспроизводимость	Своевременность	Точно определенная конфигурация
1 Таблица контрольных проверок	R1	R1	R1	—	R1	—	—
2 Таблицы решений (таблицы истинности)	R1	R2	—	—	R2	—	—
3 Анализ отказов	R2	R2	R1 (Анализ отказов основан на согласованных списках отказов)	—	R1 (Анализ отказов основан на согласованных списках отказов)	—	—
4 Анализ отказов по общей причине различного программного обеспечения (если различное программное обеспечение используется)	R2	R2	R1 (Если анализ отказов по общей причине основан на согласованных списках источников общих причин)	—	R1 (Если анализ отказов по общей причине основан на согласованных списках источников общих причин)	—	—
5 Структурные схемы надежности	R1	R1	—	—	—	—	—
6 Прямая прослеживаемость между требованиями раздела 8 и планом оценки функциональной безопасности программного обеспечения	R1 Уверенность в том, что план оценки функциональной безопасности программного обеспечения соответствует требованиям раздела 8	—	—	—	—	—	—

С.3 Свойства систематической полноты безопасности. Подробные таблицы

Таблица С.11 — Подробные свойства — стандарты проектирования и кодирования (см. таблицу В.1 приложения В)

Метод/средство	Свойство							
	Полнота спецификации требований к программному обеспечению системы безопасности	Корректность спецификации требований к программному обеспечению системы безопасности	Отсутствие собственных ошибок проекта	Простота и ясность	Предсказуемость поведения	Верифицируемость и тестируемость проекта	Отказоустойчивость/ обнаружение отказов	Отсутствие отказов по общей причине
1 Использование стандартов кодирования для сокращения вероятности ошибок	—	—	R1	R1 Запрещает отдельные конструкции языка	R1	R1	—	—
2 Не использовать динамические объекты	—	—	R1/ R2/ R3 В зависимости от используемого языка	—	R1/ R2 В зависимости от используемого языка	R1/ R2 В зависимости от используемого языка	—	—
3a Не использовать динамические переменные	—	—	R1/ R2/ R3 В зависимости от используемого языка	—	R1/ R2 В зависимости от используемого языка	R1/ R2 В зависимости от используемого языка	—	—
3b Проверка создания динамических переменных при выполнении программы	—	—	R1/ R2/ R3 В зависимости от используемого языка	—	R1/ R2 В зависимости от используемого языка	R1/ R2 В зависимости от используемого языка	—	—
4 Ограниченное использование прерываний	—	—	R1/ R2 В зависимости от используемого языка	R1 Повышает ясность логики и последовательностей событий	R1/ R2 В зависимости от используемого языка	R1/ R2 В зависимости от используемого языка	—	—
5 Ограниченное использование указателей	—	—	R1/ R2 В зависимости от используемого языка	R1 Повышает ясность логики	R1/ R2 В зависимости от используемого языка	R1/ R2 В зависимости от используемого языка	—	—

Окончание таблицы С.11

Метод/средство	Свойство							
	Полнота спецификации требований к программному обеспечению системы безопасности	Корректность спецификации требований к программному обеспечению системы безопасности	Отсутствие собственных ошибок проекта	Простота и ясность	Предсказуемость поведения	Верифицируемость и тестируемость проекта	Отказоустойчивость/ обнаружение отказов	Отсутствие отказов по общей причине
6 Ограниченное использование рекурсий	—	—	R1/ R2 В зависимости от используемого языка	—	R1/ R2 В зависимости от используемого языка	R1/ R2 В зависимости от используемого языка	—	—
7 Не использовать неструктурированное управление в программах, написанных на языках высокого уровня	—	—	R1/ R2 В зависимости от используемого языка	R1 Повышает ясность логики	R1/ R2 В зависимости от используемого языка	R1/ R2 В зависимости от используемого языка	—	—
8 Не использовать автоматическое преобразование типов	—	R2 Предотвращает погрешности округления	R2 Предотвращает погрешности округления	R1	R1	—	—	—

Таблица С.12 — Подробные свойства — динамический анализ и тестирование (см. таблицу В.2 приложения В)

Метод/средство	Свойство			
	Полнота тестирования и верификации в соответствии со спецификацией проекта программного обеспечения	Корректность тестирования и верификации в соответствии со спецификацией проекта программного обеспечения (успешное выполнение)	Воспроизводимость	Точно определенная тестируемая и верифицируемая конфигурация
1 Выполнение тестового примера, связанного с анализом граничных значений	—	R1 (R2, если заданы объективные критерии для граничных значений)	—	—
2 Выполнение тестового примера, связанного с предполагаемой ошибкой	—	R1	—	—

Окончание таблицы С.12

Метод/средство	Свойство			
	Полнота тестирования и верификации в соответствии со спецификацией проекта программного обеспечения	Корректность тестирования и верификации в соответствии со спецификацией проекта программного обеспечения (успешное выполнение)	Воспроизводимость	Точно определенная тестируемая и верифицируемая конфигурация
3 Выполнение тестового примера, связанного с введением ошибки	—	R1	—	—
4 Выполнение тестового примера, сгенерированного на основе модели	R2 MBT используется, начиная с требований, и позволяет на ранних стадиях выявлять ошибки при проектировании и разработке программного обеспечения. R3 Если при моделировании используются формальные выводы и генерация тестовых примеров (TCG)	R2 Оценка результатов и комплектов регрессионных тестов — ключевое преимущество MBT, что облегчает понимание последствий указанных требований. R3 Если применяются формальные модели, то можно получить объективные данные по охвату тестами	R3 MBT (с TCG) направлен на автоматическое выполнение сгенерированных тестов	R2 MBT работает автоматически, тестируемая конфигурация должна быть точно определена; выполнение сгенерированных тестов подобно тестированию методом «черного ящика» с возможностью измерения уровня охвата исходного кода
5 Моделирование реализации	—	R1 (R2, если цель — требования к производительности)	—	—
6 Разделение входных данных на классы эквивалентности	R1 Если профиль входных данных четко определен и прост в управлении своей структурой	R1 (Если разбиения, скорее всего, не содержат нелинейности, т. е. все члены класса являются действительно эквивалентными)	—	—
7 Структурное тестирование	—	R1 (R2 — существуют объективные цели охвата структурированием)	—	—

Таблица С.13 — Подробные свойства — функциональное тестирование и проверка методом «черного ящика» (см. таблицу В.3 приложения В)

Метод/средство	Свойство			
	Полнота тестирования, интеграции и подтверждения соответствия в соответствии со спецификациями проекта	Корректность тестирования интеграции и подтверждения соответствия в соответствии со спецификациями проекта (успешное выполнение)	Воспроизводимость	Точно определенная конфигурация для тестирования, интеграции и подтверждения соответствия
1 Выполнение тестового примера на основе причинно-следственных диаграмм	R1	R1	—	—
2 Выполнение тестового примера, сгенерированного на основе модели (MBT)	R2 Тестирование, основанное на модели, обеспечивает автоматическую генерацию эффективных контрольных примеров/процедур, используя модели системных требований и заданной функциональности. MBT облегчает раннее выявление ошибок и понимание последствий указанных требований. R3 Если при моделировании используются формальные выводы и генерация тестовых примеров (TCG)	R2 MBT основан на моделях (в основном функциональных/поведенческих), формируемых на основании требований. R3 Если применяются формальные модели, то можно получить объективные данные по охвату тестами	R3 MBT (с TCG) направлен на автоматическое выполнение сгенерированных тестов	R2 MBT работает автоматически, тестируемая конфигурация должна быть точно определена
3 Макетирование/анимация	—	R1	—	—
4 Разделение входных данных на классы эквивалентности, включая анализ граничных значений	R1 Если профиль входных данных четко определен и прост в управлении своей структурой	R1 (Если разбиения скорее всего не содержат нелинейности, то есть все члены класса являются действительно эквивалентными)	—	—
5 Моделирование процесса	—	R1	—	R2 Дает определение внешнего окружения

Таблица С.14 — Подробные свойства — анализ отказов (см. таблицу В.4 приложения В)

Метод/средство	Свойство						
	Полнота оценки функциональной безопасности в соответствии с требованиями настоящего стандарта	Корректность оценки функциональной безопасности в соответствии с проектными спецификациями (успешное выполнение)	Доступное для анализа решение всех выявленных проблем	Возможность модификации оценки функциональной безопасности после изменения проекта без необходимости проведения серьезной переработки оценки	Воспроизводимость	Своевременность	Точно определенная конфигурация
1а Причинно-следственные диаграммы	R2	R2	—	—	—	—	—
1в Анализ методом дерева событий	R2	R2	—	—	—	—	—
2 Анализ методом дерева отказов	R2	R2	—	—	—	—	—
3 Анализ функциональных отказов программного обеспечения	R2	R2	—	—	—	—	—

Таблица С.15 — Подробные свойства — моделирование (см. таблицу В.5 приложения В)

Метод/средство	Свойство			
	Полнота подтверждения соответствия в соответствии со спецификацией проекта программного обеспечения	Корректность подтверждения соответствия в соответствии со спецификацией проекта программного обеспечения (успешное выполнение)	Воспроизводимость	Точно определенная конфигурация подтверждения соответствия
1 Диаграммы потоков данных	—	R1	—	—
2а Метод конечных автоматов	R3	R3	—	—
2b Формальные методы	R3	R3	—	—
2с Моделирование во времени сетями Петри	—	R1	—	—
3 Моделирование реализации	—	R1	—	—
4 Макетирование/анимация	—	R1	—	—
5 Структурные диаграммы	—	R1	—	—

Таблица С.16 — Подробные свойства — тестирование характеристик реализации (см. таблицу В.6 приложения В)

Метод/средство	Свойство			
	Полнота тестирования и интеграции в соответствии со спецификациями проекта	Корректность тестирования и интеграции в соответствии со спецификациями проекта (успешное выполнение)	Воспроизводимость	Точно определенная тестируемая и интегрируемая конфигурация
1 Проверка на критические нагрузки и стресс-тестирование	—	R1 (R2, если установлены объективные цели)	—	—
2 Ограничения на время ответа и объем памяти	—	R1 (R2, если установлены объективные цели)	—	—
3 Требования к реализации	—	R1 (R2, если установлены объективные цели)	—	—

Таблица С.17 — Подробные свойства — полуформальные методы (см. таблицу В.7 приложения В)

Метод/средство	Свойство									
	Полнота спецификации требований к программному обеспечению системы безопасности	Корректность спецификации требований к программному обеспечению системы безопасности	Отсутствие собственных ошибок в проекте	Ясность требований к системе безопасности	Отсутствие неблагоприятного взаимовлияния функций, не связанных с безопасностью, с системой безопасности, реализуемой программным обеспечением	Простота и ясность	Предсказуемость поведения	Верифицируемость и тестируемость проекта	Отказоустойчивость/ обнаружение отказов	Отсутствие отказов по общей причине от внешних событий
1 Логические диаграммы/диаграммы функциональных блоков	R2	R2	R2	—	—	R1	R2	—	—	R1
2 Последовательностные диаграммы	R2	R2	R2	—	—	R1	R2	—	—	R2
3 Диаграммы потоков данных	R1	R1	R1	—	—	R1 Подходит для обработки транзакций	—	—	—	R1

Окончание таблицы С.17

Метод/средство	Свойство									
	Полнота спецификации требований к программному обеспечению системы безопасности	Корректность спецификации требований к программному обеспечению системы безопасности	Отсутствие собственных ошибок в проекте	Ясность требований к системе безопасности	Отсутствие неблагоприятного взаимовлияния функций, не связанных с безопасностью, с системой безопасности, реализуемой программным обеспечением	Простота и ясность	Предсказуемость поведения	Верифицируемость и тестируемость проекта	Отказоустойчивость/обнаружение отказов	Отсутствие отказов по общей причине от внешних событий
4a Конечные автоматы/диаграммы переходов	R2	R2	R2	—	—	R1 Математически полная спецификация последовательности событий	R2	—	—	R2
4b Моделирование во времени сетями Петри	R2	R2	R2	—	—	R1 Определяет взаимодействия в реальном времени	R2	—	—	R2
5 Модели данных сущность-связь-атрибут	R1	R1	R1	—	—	R1	—	—	—	R1
6 Диаграммы последовательности сообщений	R2	R2	R2	—	—	R1	R2	—	—	R2
7 Таблицы решений и таблицы истинности	R2	R2	R2	—	—	R1 Для комбинационной логики	R2	—	—	R2

Таблица С.18 — Свойства для систематической полноты безопасности — статический анализ (см. таблицу В.8 приложения В)

Метод/средство	Свойство			
	Полнота верификации в соответствии с предыдущей стадией	Корректность верификации в соответствии с предыдущей стадией (успешное выполнение)	Воспроизводимость	Точно определенная верифицируемая конфигурация
1 Анализ граничных значений	—	R1 (R2, если заданы объективные критерии для граничных значений)	—	—
2 Таблица контрольных проверок	—	R1	—	R1
3 Анализ потоков управления	—	R1	—	—
4 Анализ потоков данных	—	R1	—	—
5 Предположение ошибок	—	R1	—	—
6a Формальные проверки, включая конкретные критерии	R2	R2	—	R2
6b Сквозной контроль (программного обеспечения)	R1	R1	—	R1
7 Тестирование на символьном уровне	—	R2 R3, если применяется для формально определенных предусловий и постусловий и выполняется инструментальным средством, использующим математически строгий алгоритм	—	—
8 Анализ проекта	R2	R1 R2 (с объективными критериями)	—	R2
9 Статический анализ выполнения программы с ошибкой	—	R1 R3 для определенных классов ошибок, если выполняется инструментальным средством, использующим математически строгий алгоритм	—	—
10 Временно́й анализ выполнения при наихудших условиях	R1	R3	—	R2

Таблица С.19 — Подробные свойства — модульный подход (см. таблицу В.9 приложения В)

Метод/средство	Свойство							
	Полнота спецификации требований к программному обеспечению системы безопасности	Корректность спецификации требований к программному обеспечению системы безопасности	Отсутствие собственных ошибок проекта	Простота и ясность	Предсказуемость поведения	Верифицируемость и тестируемость проекта	Отказоустойчивость/ обнаружение отказов	Отсутствие отказов по общей причине
1 Ограничение размера программного модуля	—	—	R1	R1	R1	R1	—	—
2 Управление сложностью программного обеспечения	—	—	R1	R1	R1	R1	—	—
3 Ограничение доступа/инкапсуляция информации	—	—	R1	R1	R1	R1	—	—
4 Ограниченное число параметров / фиксированное число параметров подпрограммы	—	—	R1	R1	R1	R1	—	—
5 Одна точка входа и одна точка выхода в каждой подпрограмме и функции	—	—	R1	R1	R1	R1	—	—
6 Полностью определенный интерфейс	—	—	R2	R1	R1	R1	—	—

Приложение D
(обязательное)

Руководство по безопасности для применяемых изделий. Дополнительные требования к элементам программного обеспечения

D.1 Цель руководства по безопасности

D.1.1 Когда элемент используется вновь или предполагается, что он будет снова использован в одной или более других разрабатываемых системах, необходимо гарантировать, чтобы этот элемент сопровождался достаточно точным и полным описанием (функций, ограничений и доказательств) с тем, чтобы обеспечить возможность оценки полноты безопасности, заданной функции безопасности, которая полностью или частично реализуется этим элементом. Такие действия должны выполняться только с использованием руководства по безопасности.

D.1.2 Руководство по безопасности может состоять из документации поставщика элемента, если она адекватно соответствует требованиям приложения D IEC 61508-2 и настоящего приложения. В противном случае такая документация должна быть создана как часть проекта системы, связанной с безопасностью.

D.1.3 Руководство по безопасности должно определять атрибуты элемента, которые могут включать в себя аппаратные и/или программное ограничения, которые интегратор должен знать и учесть в процессе реализации применения. В частности руководство по безопасности является источником информации для интегратора о свойствах элемента и для чего он был разработан, о поведении элемента и его характеристиках.

Примечание — Область применения и время поставки руководства по безопасности будет зависеть от того, кто его применяет, от типа интегратора, цели элемента и от того, кто его поставляет и поддерживает.

2 Физическое лицо или отдел, или организацию, которые интегрируют программное обеспечение, называют интегратором.

D.2 Содержание руководства по безопасности для элемента программного обеспечения

D.2.1 Руководство по безопасности должно содержать всю информацию, требуемую по приложению D IEC 61508-2, которая относится к элементу. Например, пункты приложения D IEC 61508-2, связанные с аппаратными средствами, не относятся непосредственно к элементу программного обеспечения.

D.2.2 Элемент должен быть идентифицирован, и все необходимые указания по его использованию должны быть доступны интегратору.

Примечание — Для программного обеспечения это может быть продемонстрировано с помощью четкого определения элемента и демонстрацией того, что содержание информации об элементе остается неизменным.

D.2.3 Конфигурация элемента:

a) Конфигурация элемента программного обеспечения, среды выполнения программного и аппаратного обеспечения и, в случае необходимости, конфигурации системы компиляции/связей должны быть документально оформлены в руководстве по безопасности.

b) Рекомендуемая конфигурация элемента программного обеспечения должна быть документально оформлена в руководстве по безопасности, и эта конфигурация должна использоваться в применении, связанном с безопасностью.

c) Руководство по безопасности должно включать в себя все сделанные предположения, от которых зависит обоснование использования элемента.

D.2.4 Руководство по безопасности должно содержать:

a) Компетентность. Должен быть определен минимальный уровень знаний, предполагаемый для интегратора элемента, то есть знание конкретных инструментальных средств применения.

b) Степень доверия, отнесенная к элементу. Информация о любой сертификации элемента, прошедшего независимую оценку, значении полноты, которую интегратор может приписать уже существующему элементу. Эта информация должна включать в себя данные о полноте безопасности, для которой элемент разрабатывается, о стандартах, использовавшихся во время процесса проектирования, и о любых ограничениях, которые интегратор должен реализовать для обеспечения требуемой стойкости к систематическим отказам (в зависимости от функциональности элемента возможно, что некоторые требования могут быть удовлетворены только на стадии интеграции системы. В таких случаях эти требования должны быть идентифицированы для дальнейшего использования интегратором, например время отклика и быстродействие).

Примечание — В отличие от IEC 61508-2, настоящий стандарт не требует, чтобы в руководстве по безопасности для применяемых изделий были указаны режимы отказов программного обеспечения или значения интенсивностей отказов, так как причины отказов программного обеспечения существенно отличаются от причин случайных отказов оборудования, см. приложение D IEC 61508-2.

c) Инструкции по установке. Подробное описание или ссылка на него о том, как установить уже существующий элемент в интегрированную систему.

d) Причина появления релиза элемента. Подробное описание о том, что уже существующий элемент стал предметом очередного релиза, чтобы устранить серьезные отклонения или включить дополнительный функционал.

e) Серьезные отклонения. Должно быть приведено подробное описание всех серьезных отклонений с объяснением того, как происходит каждое отклонение, а также механизмы, которые должен использовать интегратор для ослабления отклонения, влияющего на конкретные функции.

f) Совместимость с предыдущими релизами. Подробное описание того, совместим ли элемент с предыдущими релизами подсистемы, а в противном случае — подробное описание процедуры его обновления, которую необходимо выполнить.

g) Совместимость с другими системами. Уже существующий элемент может зависеть от специально разработанной операционной системы. В таких случаях должны быть подробно описаны детали версии специально разработанной операционной системы.

Должен также быть определен стандарт на создание элемента, включающий в себя идентификацию и версию компилятора, инструменты, используемые для создания уже существующего элемента (идентификацию и версию), и используемый тест для уже существующего элемента (идентификацию и версию).

h) Конфигурация элемента. Для уже существующего элемента должны быть даны имя(имена) и описание(я), включая версию элемента/номер элемента/состояние модификации элемента.

i) Управление изменениями. Механизм, с помощью которого интегратор может инициировать запрос на изменение разработчику программного обеспечения.

j) Невыполненные требования. Могут существовать требования, которые были определены, но не были выполнены в текущей версии элемента. В таких случаях эти требования должны быть идентифицированы для того, чтобы их рассмотрел интегратор.

k) Предусмотренное проектом безопасное состояние. При определенных обстоятельствах в случае появления контролируемого отказа при применении системы элемент может перейти к своему безопасному состоянию, предусмотренному проектом. В таких случаях должно быть приведено точное определение предусмотренного проектом безопасного состояния, которое анализируется интегратором.

l) Ограничения интерфейса. Должны быть подробно описаны любые конкретные ограничения для заданных требований пользовательского интерфейса.

m) Подробное описание любых мер обеспечения безопасности, которые, возможно, были реализованы для предотвращения перечисленных угроз и уязвимостей.

n) Конфигурируемые элементы. Должны быть подробно описаны метод или методы конфигурации, используемые для элемента, их применение и любые ограничения на их применение.

D.3 Обоснование требований для применяемых изделий в руководстве по безопасности

D.3.1 В руководстве по безопасности все требования для применяемых изделий должны быть обоснованы соответствующим доказательством. См. 7.4.9.7 IEC 61508-2.

Примечание 1 — Важно, что требуемая характеристика элемента системы безопасности поддерживается достаточными доказательствами. Неподдержанные требования не позволяют установить правильность и полноту функции безопасности, которую реализует элемент.

Примечание 2 — Доказательство поддержки может быть получено из документации поставщика элемента и разработчика процесса, выполняемого элементом, или может быть создано или расширено квалифицированными специалистами, разрабатывающими систему, связанную с безопасностью, или третьими сторонами.

Примечание 3 — На доступность доказательства могут влиять коммерческие или юридические ограничения (например, авторское право или права интеллектуальной собственности). Эти ограничения выходят за рамки настоящего стандарта.

D.3.2 Доказательство поддержки, которое в руководстве по безопасности обосновывает требования для применяемых изделий, отличается от аналогичного обоснования в руководстве по безопасности элемента.

D.3.3 Если такое доказательство, необходимое для оценки функциональной безопасности, является недоступным, то элемент не подходит для использования в E/E/PE системах, связанных с безопасностью.

Приложение Е
(справочное)

Связь между IEC 61508-2 и настоящим стандартом

Таблицы Е.1 и Е.2 помогают определить, какие разделы IEC 61508-2 необходимо использовать тем, кто имеет дело только с программным обеспечением, а какими разделами можно при этом пренебречь. Известно, что почти все разделы IEC 61508-2 связаны с решением аппаратных проблем. В настоящем стандарте рассмотрены важные вопросы программного обеспечения, однако в IEC 61508-2 сформулировано много требований, связанных с программным обеспечением, которые, как правило, частично дублируют требования настоящего стандарта. Знание IEC 61508-2 главным образом необходимо для тех специалистов по программному обеспечению, которые занимаются совместимостью программного обеспечения и аппаратных средств. Требования IEC 61508-2 можно сгруппировать в категории по таблице Е1:

Таблица Е.1 — Категории требований IEC 61508-2

Программное обеспечение	Для пользователей стандарта, работающих с аппаратными средствами и для пользователей, работающих с программным обеспечением
Прикладное программное обеспечение	Для пользователей, работающих с программным обеспечением, которое непосредственно реализует соответствующую функцию безопасности, но не с программным обеспечением операционной системы или библиотечными функциями
Системное программное обеспечение	Для пользователей, работающих, прежде всего с программным обеспечением операционной системы, библиотечными функциями и т. п.
Только аппаратные средства	Только для пользователей, не работающих с программным обеспечением
В основном аппаратные средства	Для пользователей, проявляющих только незначительный интерес к проблемам программного обеспечения

Требования IEC 61508-2 к программному обеспечению и их связь с определенными в таблице Е.1 категориями требований представлены в таблице Е.2.

Таблица Е.2 — Требования IEC 61508-2 к программному обеспечению и их связь с определенными в таблице Е.1 категориями требований

Требование по IEC 61508-2	Обеспечение/средство важное для пользователей	Примечание
7.2	Программное обеспечение	—
7.2.3.1	Прикладное программное обеспечение	—
7.2.3.2—7.2.3.6	Программное обеспечение	—
7.2.3.3	Только аппаратные средства	—
7.3	Программное обеспечение	7.3.2.2 перечисление f) — только аппаратные средства
7.4	Программное обеспечение	—
7.4.2.1—7.4.2.12	Программное обеспечение	—
7.4.2.13, 7.4.2.14	Только аппаратные средства	—
7.4.3.1—7.4.3.3	Программное обеспечение	—
7.4.3.4	Только аппаратные средства	—
7.4.4	Только аппаратные средства	—
7.4.5	Только аппаратные средства	—

Окончание таблицы Е.2

Требование по IEC 61508-2	Обеспечение/средство важное для пользователей	Примечание
7.4.6	Программное обеспечение	7.4.7.6 — только аппаратные средства
7.4.7	Программное обеспечение	7.4.7.1 перечисления a), b) — только аппаратные средства
7.4.8	Только аппаратные средства	—
7.4.9.1—7.4.9.3	Программное обеспечение	—
7.4.9.4, 7.4.9.5	Только аппаратные средства	—
7.4.9.6, 7.4.9.7	Программное обеспечение	—
7.4.10	Программное обеспечение	—
7.4.11	Только аппаратные средства	—
7.5	Программное обеспечение	—
7.6	Программное обеспечение	—
7.6.2.1 перечисление a)	Аппаратные средства	—
7.6.2.4	В основном аппаратные средства	—
7.7	Программное обеспечение	7.7.2.3, 7.7.2.4 — в основном прикладное программное обеспечение
7.8	Программное обеспечение	—
7.9	В основном прикладное программное обеспечение	—
8	Программное обеспечение	—
Приложение А.1	В основном аппаратные средства	—
Приложение А.2 и таблицы	В основном аппаратные средства	Таблица А.10 — программное обеспечение
Приложение А.3	В основном аппаратные средства	Таблицы А.18, А.17, А.18 — содержат некоторые аспекты программного обеспечения
Приложение В, все таблицы	Программное обеспечение	—
Приложение С	Аппаратные средства	—
Приложение D	Программное обеспечение	D.2.3 — только аппаратные средства
Приложение Е	Только аппаратные средства	—
Приложение F	Только аппаратные средства	—

Приложение F
(справочное)

Методы, не допускающие взаимодействие между элементами программного обеспечения на одном компьютере

F.1 Введение

Независимое выполнение элементов программного обеспечения, работающих в одной компьютерной системе (состоящей из одного или более процессоров с памятью и другими устройствами, совместно используемыми этими процессорами), можно обеспечить и продемонстрировать с помощью различных методов. Настоящее приложение рассматривает некоторые методы, не допускающие взаимодействие (между элементами с различной стойкостью к систематическим отказам, между элементами, разработанными для реализации (или для принятия участия в реализации) одной и той же функции безопасности, или между программным обеспечением, реализующим функции, связанные с безопасностью, и программным обеспечением, реализующим функции, не связанные с безопасностью, на одном компьютере).

Примечание — Термин «независимость выполнения» означает, что элементы в процессе их выполнения не будут неблагоприятно влиять друг на друга, то есть это не приведет к появлению опасного отказа. Этот термин используется, чтобы отличить другие аспекты независимости, которые могут потребоваться между элементами (в частности «разнообразие») и которые соответствуют другим требованиям настоящего стандарта.

F.2 Области поведения

Независимость выполнения должна быть обеспечена и продемонстрирована в областях пространства и времени.

Пространственная область. Данные, используемые одним элементом, не должны быть изменены другим элементом. В частности данные не должны быть изменены элементом, несвязанным с безопасностью.

Временна́я область. Функционирование одного элемента не должно приводить к неправильному функционированию другого элемента, используя слишком большую часть общего времени процессора, или блокируя работу другого элемента, запирая какой-либо совместно используемый ресурс.

F.3 Анализ причин

Для демонстрации независимости выполнения должна быть проведена для предложенного проекта идентификация всех возможных причин взаимовлияний между умозрительно независимыми (невзаимовлияющими) функционирующими элементами в пространственной и временно́й областях. Анализ должен быть проведен при нормальных условиях функционирования и в условиях отказа и должен рассмотреть (но не должен быть ограничен):

a) использование совместно используемой оперативной памяти;

b) использование совместно используемых периферийных устройств;

c) совместное использование времени процессора (где два или более элементов выполняются на одном процессоре);

d) коммуникации между элементами, необходимые для создания проекта всей системы;

e) возможность того, что отказ в одном элементе (такой как переполнение или исключительная ситуация деления на ноль, или неправильное вычисление указателя) может вызвать последовательные отказы в других элементах.

Для обеспечения и обоснования независимости выполнения необходимо рассмотреть все эти идентифицированные источники взаимовлияний.

F.4 Обеспечение пространственной независимости

Методы для обеспечения и демонстрации пространственной независимости включают в себя:

a) Использование аппаратной защиты памяти между различными элементами, включая элементы, различающиеся стойкостью к систематическим отказам.

b) Использование операционной системы, которая для каждого элемента реализует отдельный процесс с его собственным пространством виртуальной памяти, поддерживаемым аппаратной защитой памяти.

c) Использование строгого анализа проекта, исходного кода и, возможно, объектного кода для демонстрации отсутствия любых явных или неявных обращений одного элемента программного обеспечения к памяти другого элемента, которые могут привести к искажению данных, принадлежащих этому элементу (для случая отсутствия аппаратной защиты памяти).

d) Программная защита от недопустимой модификации элементом с более низким уровнем полноты данных элемента с более высоким уровнем полноты.

Данные нельзя передавать от элемента с более низким уровнем полноты к элементу с более высоким уровнем, если элемент с более высоким уровнем полноты не может проверить наличие у данных достаточного уровня полноты.

Если необходимо передать данные между элементами, которые должны выполняться независимо, то должны использоваться однонаправленные интерфейсы, такие, например, как сообщения или каналы, а не совместно используемая память.

Примечание — В идеальном случае независимые элементы не должны взаимодействовать друг с другом. Однако, если проект системы требует, чтобы один элемент передавал данные другому элементу, то проект коммуникационного механизма должен быть таким, чтобы отправляющий и/или получающий сообщение элементы не находились в состоянии отказа или их функционирование не было бы блокировано в случае прекращения или задержки передачи данных.

Кроме переменной информации в оперативной памяти при пространственном разделении должен быть учтен любой резидентный объект данных в постоянных запоминающих устройствах, таких как магнитные диски. Например, защита доступа к файлу, реализованная операционной системой, может использоваться для предотвращения записи одним элементом в принадлежащие другому элементу области данных.

F.5 Обеспечение временно́й независимости

Методы, гарантирующие временну́ю независимость включают в себя:

a) Детерминированные методы планирования, например:

- циклический алгоритм планирования, который дает каждому элементу определенный интервал времени, полученный для каждого элемента в результате анализа времени его работы в наихудшем случае, чтобы статически продемонстрировать выполнение требований синхронизации для каждого элемента;
- архитектуры с временны́м распределением.

b) Планирование, основанное на строгом приоритете, реализованное программой-диспетчером, работающей в реальном времени, со средствами, запрещающими смену приоритетов.

c) Использование временны́х заграждающих меток, которые завершают выполнение элемента, если происходит превышение выделенного для него времени выполнения или максимального времени (в этом случае должен быть проведен анализ риска, показывающий, что завершение выполнения элемента не приведет к опасному отказу и таким образом этот метод может быть лучше всего использован для элемента, не связанного с безопасностью).

d) Использование возможности операционной системы, которая запрещает какому-либо процессу использовать полностью временны́е ресурсы процессора, например, с помощью квантования времени. Такой подход применим, только если элементы, связанные с безопасностью, не задают жестких требований к режиму реального времени и если показано, что алгоритм планирования не приведет к неоправданным задержкам обращения к какому-либо элементу.

Если элементы совместно используют ресурс (например, периферийное устройство), то проект должен гарантировать, что элементы будут функционировать корректно, так как совместно используемый ресурс блокируется другим элементом. При определении отсутствия временно́го взаимовлияния необходимо учесть время получения доступа к совместно используемому ресурсу.

F.6 Требования к программному обеспечению поддержки

Если операционная система, программа-диспетчер, работающая в реальном времени, управление памятью, управление таймером или какое-либо подобное программное обеспечение должны использоваться для обеспечения пространственной или временно́й независимости (или обоих), то в таком программном обеспечении любой из его элементов, которые должны быть независимыми, должен обладать самой высокой стойкостью к систематическим отказам.

Примечание — Однако, в любом таком программном обеспечении для независимых элементов существует возможность отказа по общей причине.

F.7 Независимость программных модулей. Аспекты языка программирования

Таблица F.1 содержит неформальные определения используемых терминов.

Таблица F.1 — Связывание модулей. Определение терминов

Термин	Неформальное определение
Связность	Мера прочности связей между данными и подпрограммами внутри одного модуля
Связывание	Мера прочности связей между модулями
Инкапсуляция	Ограничение внешнего доступа к внутренним (личным) данным и подпрограмм; термин используется главным образом с объектно-ориентированными программами

Окончание таблицы F.1

Термин	Неформальное определение
Независимость	Мера отсутствия связей между частями программы; антоним понятия «связывание»
Модуль	Выполняющая некоторую функцию, ограниченная часть программного обеспечения, которая может иметь собственные данные: класс, иерархию классов, подпрограммы, блок, модуль, пакет и др. в соответствии с языком программирования
Интерфейс	Хорошо определенный набор заголовков подпрограмм, обеспечивающий доступ к модулю
Случайные данные (данные «бродяги»)	Полученные данные, не используемые в модуле, но передающиеся в другой модуль

Как правило, независимость модуля увеличивается, если между модулями существуют слабое связывание и сильная связность внутри модулей. Сильная связность обеспечивает ситуацию, в которой идентифицируемые функциональные модули точно соответствуют идентифицируемым модулям реализации, а слабое связывание модулей обеспечивает незначительное взаимодействие и, следовательно, высокую степень независимости между функционально не связанными модулями.

Слабо связанный модуль обычно формируется в результате сильной связности внутри модуля, объединяя вместе код и используемые данные для выполнения одной конкретной функции. Слабая связность формируется в модулях, если код и данные объединены достаточно свободно, или в результате некоторой временно́й последовательности, или в результате некоторой последовательности потока управления.

Необходимо различать виды связывания модулей (см. таблицу F.2).

Таблица F.2 — Виды связывания модулей

Связывание	Определение	Объяснение	Обоснование	Примечание
Связывание с помощью интерфейсов, икапсуляция	Связывание только для хорошо определенного множества подпрограмм	Доступ к модулю или к его данным только через подпрограммы; любое изменение величины переменной, любой вопрос о величине такой переменной или любой сервис, требуемый от модуля, выполняется через вызов подпрограммы	Заголовки подпрограмм (сигнатуры) модулей объясняют доступные сервисы. Если требуется изменить модуль, то большая часть этих изменений может быть выполнена в самом модуле не затрагивая другие модули. Поддерживает слабое связывание — в целом рекомендуется	В основном для объектно-ориентированных программ, классов, иерархии классов, библиотек; не для подпрограмм
Связывание с помощью данных списка параметров	Передача только данных списка параметров или идентификаторов подпрограмм	Доступ к модулю или к его данным только через переменные или объекты, указанные в заголовке подпрограммы; любое изменение величины переменной, любой вопрос о величине такой переменной являются явными	Заголовок подпрограмм содержит данные или объекты, включенные в вызов подпрограммы. Поддерживает слабое связывание — в целом рекомендуется	Внутри классов объектно-ориентированных программ этот принцип обычно не встречается. К локальным переменным можно получить прямой доступ. Строгое следование этому принципу может также привести к случайным данным. Для того, тобы этого избежать, данный принцип должен быть нарушен

Окончание таблицы F.2

Связывание	Определение	Объяснение	Обоснование	Примечание
Структурное связывание	Передаваемые данные содержат больше данных, чем необходимо	В получающую подпрограмму передается больше данных, чем это необходимо для выполнения требуемой функции	Лишние данные обеспечивают получающий модуль информацией, которая не нужна для его выполнения. Эти данные могут привести к недоразумениям при взаимодействии модулей, что, однако, не осуждается	Обычно этот недостаток может быть легко скорректирован
Связывание по управлению	Связывание, которое осуществляет непосредственное управление получающим модулем	Передача данных, которая может выполнить только передачу управления на другой модуль, и во многих случаях характеризуется передачей одного бита	Более тесное связывание, чем предыдущее, так как требует немедленного действия, предписывая получающей подпрограмме что-либо выполнить. Необходимо проявлять осторожность. В целом не рекомендуется	Не всегда можно избежать. Например, может быть необходима при завершении действия или подтверждении соответствия значения
Глобальное связывание	Связывание с помощью глобальных данных	Модули могут получить доступ к данным, к которым другие модули имеют прямой доступ, или один модуль может получить прямой доступ к данным, принадлежащим другому модулю	Заголовок подпрограмм не указывает на то, какие данные используются и откуда. Трудно понять функции подпрограмм и предсказать последствия любых изменений кода	В целом критикуемый вид связывания. Например, может быть востребован исключительно для того, чтобы избежать случайных данных. Необходимо использовать только очень ограниченно, в соответствии с ясно определенным и документально оформленным стандартом кодирования
Связывание по контенту	Прямой переход в другие модули, оказывающий влияние на условие в условных операторах в этих модулях, или прямой доступ к данным в других модулях	Реализуется в программах на языке ассемблера; не реализуется на всех языках высокого уровня. Может ускорить выполнение программы и уменьшить трудоемкость кодирования	Критикуемый вид связывания. Отдельный модуль можно понять только на том уровне, на котором понятны все, соединенные с ним модули. Делает программу чрезвычайно трудной для понимания и изменения	В некоторых языках программирования даже невозможен. Можно всегда игнорировать

Чтение или анализ кода (см. 7.9.2.12) должны проверить, слабо ли связаны программные модули. Такой анализ обычно требует своего рода понимания цели модулей и способа их выполнения. Поэтому истинное связывание может быть оценено только после прочтения кода и его документации.

Связывания по контенту необходимо избегать. Глобальное связывание может использоваться только в исключительных случаях. Связывания по управлению и структурного связывания необходимо избегать. Если возможно, модули должны быть соединены связыванием с помощью интерфейса (инкапсуляцией) и/или связыванием с помощью данных.

Приложение G
(справочное)

Руководство по адаптации жизненного цикла систем, управляемых данными

G.1 Общие положения

Система, управляемая данными. Системная и прикладная части

Многие системы состоят из двух частей. Одна часть является системной. Другая часть — прикладная, которая адаптирует систему к конкретным требованиям необходимого применения. Прикладная часть может быть создана в виде данных, которые конфигурируют системную часть. В настоящем приложении такие системы называют «управляемыми данными».

Конкретная прикладная часть такого программного обеспечения может быть разработана с использованием множества программных средств и языков программирования. Эти средства и языки могут ограничить способ создания прикладной программы.

Например, если язык программирования достаточно просто описывает функциональность (например, язык многозвенных логических схем для простых систем блокировки), то прикладное программное обеспечение для запрограммированной задачи, вероятно, будет довольно простым. Однако если язык программирования описывает поведение приложения достаточно сложно, то прикладное программное обеспечение для запрограммированной задачи, вероятно, будет сложным. Если разработано очень простое прикладное программное обеспечение, то детальное проектирование можно выполнять как конфигурирование, а не как программирование.

Степень строгости, необходимой для достижения требуемой полноты безопасности, зависит от степени сложности конфигурации, поддерживаемой средствами разработки/конфигурирования, и сложности представляемого поведения применения (см. рисунок G 1).

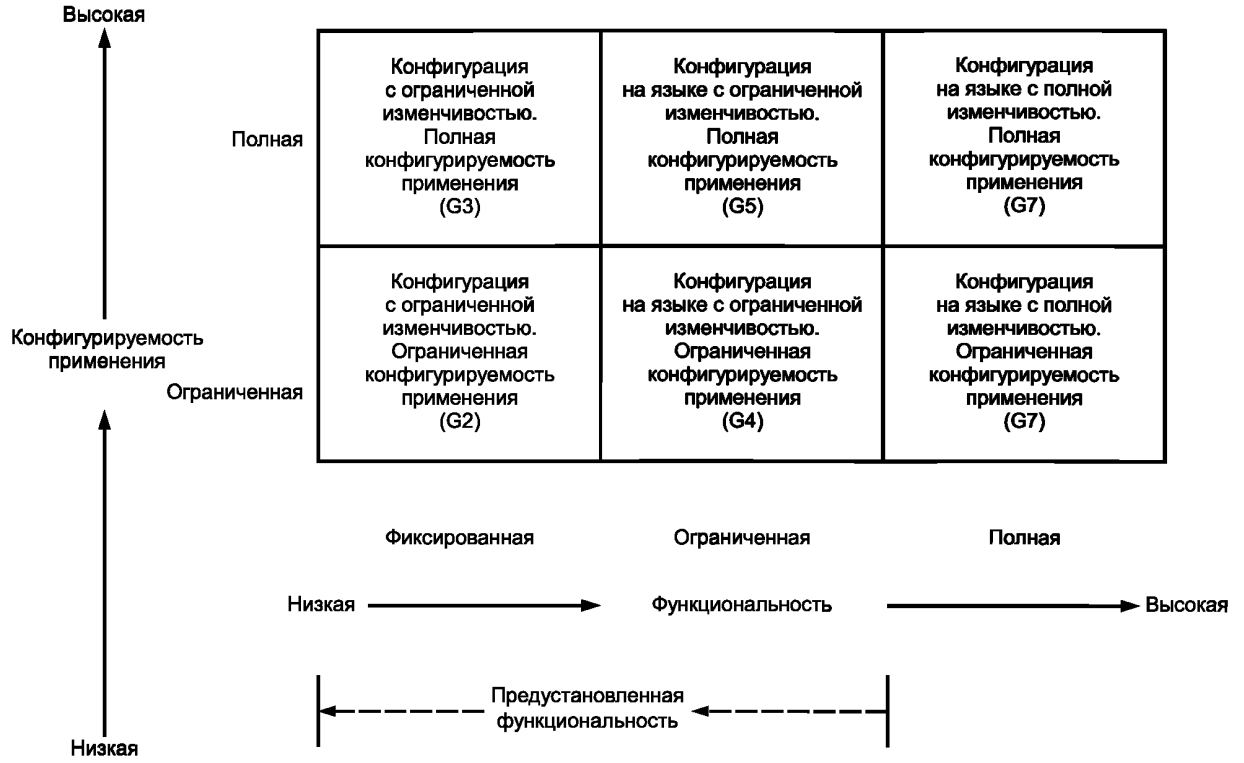

Рисунок G.1 — Изменчивость языка и сложность систем, управляемых данными

По осям на рисунке G 1откладываются классы сложности для:

а) изменчивости языка:

- фиксированная программа;
- ограниченная изменчивость (в некоторых отраслях прикладная программа рассматривается как «данные», которые интерпретируются системной частью);

- полная изменчивость (обычно не рассматриваемый как управляемый данными такой тип систем может также использоваться для разработки применений и включен в настоящее приложение для полноты картины) и

b) возможной конфигурируемости применения:

- ограниченная конфигурируемость;
- полная конфигурируемость.

В действительности конкретная система может включать в себя разные уровни сложности и конфигурируемости. Кроме того, сложность можно представить как плавную шкалу с непрерывно изменяющимися значениями по осям на рисунке G 1. Если необходимо адаптировать жизненный цикл программного обеспечения, то должен быть идентифицирован соответствующий уровень сложности и должна быть обоснована степень адаптации.

Ниже приведено описание различных типов систем для каждого уровня сложности. Указания по предполагаемым методам для реализации каждого типа системы приведены в [5].

Типичные системы для каждого класса сложности описаны в G.2.

G.2 Конфигурация с ограниченной изменчивостью, конфигурируемость применения ограничена

Для систем, соответствующих требованиям IEC 61508, используется патентованный язык конфигурации с предварительно установленной поставляемой функциональностью.

Такой язык конфигурации не позволяет программисту изменять функцию системы. А конфигурирование ограничено настройкой нескольких параметров, позволяющей системе соответствовать ее применению. Например, интеллектуальные датчики и приводы, сетевые контроллеры, контроллеры последовательности, небольшие системы регистрации данных и интеллектуальные инструменты настраиваются введением в них конкретных значений параметров.

Обоснование адаптации жизненного цикла системы безопасности должно включать в себя (но не быть ограничено):

a) спецификацию входных параметров для данного применения;

b) верификацию правильности реализации параметров в действующей системе;

c) подтверждение соответствия всех комбинаций входных параметров;

d) рассмотрение специальных и конкретных режимов работы в процессе конфигурирования;

e) человеческий фактор/эргономику;

f) блокировки, например, обеспечение гарантии того, что блокировки выполнения прошли подтверждение соответствия во время процесса конфигурации;

g) непреднамеренное реконфигурирование, например, доступа к ключу переключения, устройств защиты.

G.3 Конфигурация на языке с низкой изменчивостью, полная конфигурируемость применения

Для систем, соответствующих требованиям IEC 61508, используется патентованный язык конфигурации с предварительно установленной поставляемой функциональностью.

Такой язык конфигурации не позволяет программисту изменять функцию системы. А конфигурирование ограничено созданием обширных данных статических параметров, позволяющих системе соответствовать ее применению. Примером может служить система авиадиспетчерской службы, состоящая из данных с большим числом сущностей данных, каждая из которых имеет один или более атрибутов. Существенной характеристикой этих данных является то, что они не содержат явных последовательностей, упорядочиваний или ветвлений и не содержат представления комбинаторных состояний применения.

Обоснование адаптации жизненного цикла системы безопасности, кроме рассмотренного в G.2, должно включать в себя (но не быть ограничено):

a) средства автоматизации для создания данных;

b) проверку непротиворечивости, например, на самосовместимость данных;

c) проверку правил, например, для гарантирования генерации данных, удовлетворяющих определенным ограничениям;

d) подтверждение соответствия интерфейсов системам подготовки данных.

G.4 Конфигурация на языке с ограниченной изменчивостью, конфигурируемость применения ограничена

Для систем, соответствующих требованиям IEC 61508, используется проблемно-ориентированный язык с ограниченной предварительно установленной поставляемой функциональностью, в котором операторы языка содержат или напоминают терминологию пользователя применения.

Эти языки позволяют пользователям с ограниченной гибкостью настраивать функции системы в соответствии с собственными конкретными требованиями, используя ряд аппаратных и программных элементов.

Существенной характеристикой языка с ограниченной изменчивостью является то, что данные могут содержать явные последовательности, упорядочивания или ветвления и вызывать комбинаторные состояния применения, например, программирование на основе функциональных блоков, многозвенные логические схемы, системы, основанные на крупноформатной таблице, и графические системы.

Обоснование адаптации жизненного цикла системы безопасности, кроме рассмотренного в G.3, должно включать в себя (но не быть ограничено):

a) спецификацию основных требований применения;
b) допускаемые подмножества языка для этого применения;
c) методы разработки для объединения подмножеств языка;
d) критерии охвата проверкой решений для комбинаций возможных системных состояний.

G.5 Конфигурация на языке с ограниченной изменчивостью, полная конфигурируемость применения

Для систем, соответствующих требованиям IEC 61508, используется проблемно-ориентированный язык с ограниченной предварительно установленной поставляемой функциональностью, в котором операторы языка содержат или напоминают терминологию пользователя применения.

Существенным отличием применения языка с ограниченной изменчивостью при полной конфигурируемости применения от применения языка с ограниченной изменчивостью при ограниченной конфигурируемости применения является сложность конфигурации применения. Примерами таких применений могут служить графические системы и системы управления серийного производства на основе SCADA-систем.

Обоснование адаптации жизненного цикла системы безопасности, кроме рассмотренного в G.4, должно включать в себя (но не быть ограничено):

a) проект архитектуры применения;
b) обеспечение шаблонов;
c) верификацию отдельных шаблонов;
d) верификацию и подтверждение соответствия применения.

При реализации и тестировании модулей самого низкого уровня проблема жизненного цикла, рассматриваемая в настоящем стандарте, вероятно, будет неактуальна (в зависимости от используемого языка).

G.6 Конфигурация на языке с полной изменчивостью, конфигурируемость применения ограничена

См. G.7.

G.7 Конфигурация на языке с полной изменчивостью, полная конфигурируемость применения

Для этих систем применяются требования настоящего стандарта на всем жизненном цикле.

Части систем с полной изменчивостью разработаны на универсальных языках программирования или на универсальных языках базы данных, или с использованием универсальных научных пакетов и пакетов моделирования. Обычно эти части выполняются в компьютерной системе под управлением операционной системы, которая управляет выделением системных ресурсов и средой мультипрограммирования реального времени. Например, системы, написанные на языках с полной изменчивостью, могут включать в себя: специализированные системы управления оборудованием, специально разработанные системы управления полетом, или веб-сервисы для управления сервисами, связанными с безопасностью.

Приложение ДА
(справочное)

Сведения о соответствии ссылочных международных стандартов межгосударственным стандартам

Таблица ДА.1

Обозначение ссылочного международного стандарта	Степень соответствия	Обозначение и наименование соответствующего межгосударственного стандарта
ISO/IEC Guide 51:1990	—	*
IEC Guide 104:1997	—	*
IEC 61508-1:2010	—	*
IEC 61508-2:2010	—	*
IEC 61508-4:2010	—	*
* Соответствующий национальный стандарт отсутствует.		

Библиография

[1] IEC 61511 (all parts) Functional safety — Safety instrumented systems for the process industry sector (Безопасность функциональная. Системы безопасности приборные для промышленных процессов)

[2] IEC 62061 Safety of machinery — Functional safety of safety-related electrical, electronic and programmable electronic control systems (Безопасность оборудования. Функциональная безопасность систем управления электрических, электронных и программируемых электронных, связанных с безопасностью)

[3] IEC 61800-5-2 Adjustable speed electrical power drive systems — Part 5-2: Safety requirements — Functional (Системы силовых электроприводов с регулируемой скоростью. Часть 5-2. Требования функциональной безопасности)

[4] IEC 60601 (all parts) Medical electrical equipment (Изделия медицинские электрические (все части))

[5] IEC 61508-7: 2010 Functional safety of electrical/electronic/programmable electronic safety-related systems — Part 7: Overview of techniques and measures (Функциональная безопасность систем электрических, электронных, программируемых электронных, связанных с безопасностью. Часть 7. Методы и средства)

[6] IEC 61508-6: 2010 Functional safety of electrical/electronic/programmable electronic safety-related systems — Part 6: Guidelines on the application of IEC 61508-2 and IEC 61508-3 measures (Функциональная безопасность систем электрических, электронных, программируемых электронных, связанных с безопасностью. Часть 6. Руководство по применению МЭК 61508-2 и МЭК 61508-3)

[7] IEC 61131-3 Programmable controllers — Part 3: Programming languages (Контроллеры программируемые. Часть 3. Языки программирования)

УДК 62-783:614.8:331.454:006.354 ОКС 25.040.40 Группа Т51

Ключевые слова: функциональная безопасность; жизненный цикл систем; электрические компоненты; электронные компоненты; программируемые электронные компоненты и системы; системы, связанные с безопасностью; планирование функциональной безопасности; программное обеспечение; уровень полноты безопасности

БЗ 11—2017/2

Редактор *Р.Г. Говердовская*
Технический редактор *В.Н. Прусакова*
Корректор *С.В. Смирнова*
Компьютерная верстка *Е.А. Кондрашовой*

Сдано в набор 07.09.2018. Подписано в печать 18.09.2018. Формат 60×84⅛. Гарнитура Ариал.
Усл. печ. л. 13,02. Уч.-изд. л. 11,78.

Подготовлено на основе электронной версии, предоставленной разработчиком стандарта

Создано в единичном исполнении ФГУП «СТАНДАРТИНФОРМ»
для комплектования Федерального информационного фонда стандартов,

117418 Москва, Нахимовский пр-т, д. 31, к. 2.
www.gostinfo.ru info@gostinfo.ru