LDRA Software Technology

экспонента
ЦЕНТР ИНЖЕНЕРНЫХ ТЕХНОЛОГИЙ И МОДЕЛИРОВАНИЯ

Внедрение стандарта IEC 61508: 2010 с набором инструментов LDRA®

ldra.exponenta.ru

Введение

Благодаря новейшим достижениям в области автоматизации программное обеспечение перестало быть небольшой частью электромеханических систем, теперь это - основополагающая технология, обеспечивающая функциональную безопасность для продуктов во многих сегментах рынка.

Требование функциональной безопасности программного обеспечения стало важной темой в области промышленной автоматизации, транспорта, производства ядерной энергии и других рынков. IEC 61508 широко признается в качестве базового стандарта, на котором строятся отраслевые стандарты.

Стандарт IEC 61508 представляет собой подход, основанный на оценке риска для определения УПБ (уровень полноты безопасности программного обеспечения) функций безопасности. Если необходима эффективная и безопасная эксплуатация компьютерных систем, то необходимо, чтобы имеющиеся рекомендации по этим аспектам, связанным с безопасностью, были достаточными, чтобы можно было принять правильные решения.

Признано, что существует множество применений с использованием электронных/программируемых электронных (Э/Э/ПЭ) систем, связанных с безопасностью, в различных секторах применений и охватывающих широкий спектр сложностей, рисков. Необходимые меры безопасности для каждого конкретного приложения будут зависеть от многих факторов, характерных для него. Общий характер IEC 61508 делает его идеальным «пустым холстом» для бесшовной интеграции этих зависимых от приложения факторов и, следовательно, для определения отраслевых стандартов.

В большинстве ситуаций безопасность достигается рядом систем, которые опираются на многие технологии (например, механические, гидравлические, пневматические, электрические, электронные, программируемые электронные). Поэтому любая стратегия безопасности должна учитывать не только все элементы в отдельной системе (например, датчики, управляющие устройства и исполнительные механизмы), но также и все связанные с безопасностью подсистемы, которые вносят вклад в систему безопасности в целом. Поэтому, хотя настоящий международный стандарт касается систем безопасности, связанных с Э/Э/ПЭ, он также служит основой, в рамках которой могут быть рассмотрены системы, связанные с безопасностью, основанные на других технологиях.

В этом документе описываются основные требования к разработке и проверке программного обеспечения в стандарте **IEC 61508** и как пакет инструментов LDRA может помочь в их удовлетворении.

IEC 61508 — Часть 3 Цели процесса:

- 7.2 Спецификация требований к программному обеспечению системы безопасности
- 7.3 Планирование подтверждения соответствия безопасности системы для аспектов программного обеспечения
- 7.4 Проектирование и разработка программного обеспечения
  - 7.4.3 Требования к проектированию архитектуры программного обеспечения
  - 7.4.4 Требования к инструментальным средствам поддержки, включая языки программирования
  - 7.4.5 Требования к детальному проектированию и разработке - проектирование системы программного обеспечения
  - 7.4.6 Требования к реализации исходных текстов программ
  - 7.4.7 Требования к тестированию программных модулей
  - 7.4.8 Требования к тестированию интеграции программного обеспечения
- 7.5 Интеграция программируемой электроники (аппаратных средств и программного обеспечения)
- 7.6 Процедуры эксплуатации и модификации программного обеспечения
- 7.7 Подтверждение соответствия аспектов программного обеспечения безопасности системы
- 7.8 Модификация программного обеспечения
- 7.9 Верификация программного обеспечения
- 8 Оценка функциональной безопасности

Существует 7 приложений, определенных в IEC 61508-3:

- Приложение A (обязательное). Руководство по выбору методов и средств
- Приложение B (справочное)[1] Подробные таблицы
- Приложение C (справочное). Свойства стойкости к систематическим отказам программного обеспечения
- Приложение D (обязательное). Руководство по безопасности для применяемых изделий. Дополнительные требования к элементам программного обеспечения
- Приложение E (справочное). Связь между IEC 61508-2 и настоящим стандартом
- Приложение F (справочное). Методы, не допускающие взаимодействия между элементами программного обеспечения на одном компьютере
- Приложение G (справочное). Руководство по адаптации жизненного цикла систем, управляемых данными

В последующем обсуждении рассматриваются требования и рекомендуемый уровень безопасности 1-4 уровня из каждого приложения и устанавливается связь с соответствующими функциями набора инструментов LDRA.

[1] В ГОСТ Р МЭК 61508-3:2012 — обязательное

Таблицы приложения А:

	Метод/средство	Ссылка	УПБ			
			A	B	C	D
1a	Полуформальные методы	Таблица В.7	+✓	+✓	++✓	++✓
1b	Формальные методы	В.2.2, С.2.4	o✓	+✓	+✓	++✓
2	Прямая трассируемость[2] между требованиями к системе безопасности и требованиями к программному обеспечению системы безопасности	С.2.11	+✓	+✓	++✓	++✓
3	Обратная трассируемость между требованиями к системе безопасности и предполагаемыми потребностями безопасности	С.2.11	+✓	+✓	++✓	++✓
4	Компьютерные средства разработки спецификаций для поддержки перечисленных выше подходящих методов/средств	В.2.4	+✓	+✓	++✓	++✓

"++" Метод строго рекомендуется для этого УПБ.
"+" Метод рекомендуется для этого УПБ.
"o" Для данного метода нет как рекомендаций, так и возражений против его использования.
✓ Удовлетворяется при помощи набора инструментов LDRA.

Таблица А.1 — Спецификация требований к безопасности программного обеспечения

Таблица (А.1) относится к разделу «Спецификация требований к программному обеспечению системы безопасности» (7.2) и соответствующему разделу В.2 и С.2 стандарта IEC 61508 (часть 7).

В разделе 7.2 показаны цели, связанные с определением требований безопасности программного обеспечения. К ним относятся выработка требований к функциям безопасности программного обеспечения, систематическому обеспечению программного обеспечения и выполнению необходимых функций безопасности.

Разделы С.2 и В.2 IEC 61508 (часть 7) рекомендуют использовать схемы потоков данных и таблицы решений/правды для представления дизайна системы. Представления графа вызовов и графа потоков выполнения из набора инструментов LDRA предоставляют соответствующую информацию о сложности системы. Таблицы истины, сформированные LCSAJ (Линейные последовательности кода и переходы, англ. Linear Code Sequence and Jumps), а также планировщик тестов MC/DC представляют собой пути решений, доступных при выполнении системы.

Формальные методы (раздел В.2.2) применяют принципы математического обоснования к спецификации и реализации технических систем. Функция статического анализа набора инструментов LDRA анализирует исходный код в отношении различных моделей программирования (например, MISRA C), в то время как механизм анализатора, доступный в наборе инструментов LDRA, математически анализирует структуру и предоставляет аналитические отчеты. Эта функция анализа набора инструментов LDRA может быть связана с TBmanager (инструментом отслеживания требований LDRA), чтобы обеспечить сквозную трассируемость требований безопасности системы и требований безопасности программного обеспечения (прямая и обратная трассируемость).

[2] В ГОСТ Р МЭК 61508-3:2012 используется термин «трассируемость», в данном документе используется эквивалентный термин «трассируемость»

	Метод/средство	Ссылка	УПБ			
			A	B	C	D
1	Обнаружение ошибок	C.3.1	o	+✓	++✓	++✓
2	**Коды обнаружения ошибок**	C.3.2	+✓	+✓	+✓	++✓
3a	**Программирование с проверкой ошибок**	C.3.3	+✓	+✓	+✓	++✓
3b	Методы контроля (при реализации процесса контроля и контролируемой функции на одном компьютере обеспечивается их независимость)	C.3.4	o	+	+	o
3c	Методы контроля (реализация процесса контроля и контролируемой функции на разных компьютерах)	C.3.4	o	+	+	++
3d	Многовариантное программирование, реализующее одну спецификацию	C.3.5	o	o	o	+
3e	Функционально многовариантное программирование, реализующее различные спецификации требований к программному обеспечению	C.3.5	o	o	+	++
3f	Восстановление предыдущего состояния	C.3.6	+	+	o	o
3g	Проектирование программного обеспечения, не сохраняющего состояние (или проектирование ПО, сохраняющего ограниченное описание состояния)	C.2.12	o	o	+	++
4a	Механизмы повторных попыток парирования сбоя	C.3.7	+	+	o	o
4b	Постепенное отключение функций	C.3.8	+	+	++	++
5	Исправление ошибок методами искусственного интеллекта	C.3.9	o	o	o	o
6	Динамическая реконфигурация	C.3.10	o	o	o	o
7	Модульный подход	Table B.9	++✓	++✓	++✓	++✓
8	Использование доверительных/проверенных элементов программного обеспечения (при наличии)	C.2.10	+✓	++✓	++✓	++✓
9	Прямая трассируемость между спецификацией требований к программному обеспечению системы безопасности и архитектурой программного обеспечения	C.2.11	+✓	+✓	++✓	++✓
10	Обратная трассируемость между спецификацией требований к программному обеспечению системы безопасности и архитектурой программного обеспечения	B.2.11	+✓	+✓	++✓	++✓
11a	Методы структурных диаграмм**	C.2.1	++✓	++✓	++✓	++✓
11b	Полуформальные методы**	Table B.7	+✓	+✓	++✓	++✓
11c	Формальные методы проектирования и усовершенствования **	B.2.2, C.2.4	o	+✓	++✓	++✓
11d	Автоматическая генерация программного обеспечения	C.4.6	+	+	+	+
12	Автоматизированные средства разработки спецификаций и проектирования	B.2.4	+	+	++	++
13a	Циклическое поведение с гарантированным максимальным временем цикла	C.3.11	+	++	++	++
13b	Архитектура с временным распределением	C.3.11	+	++	++	++
13c	Управление событиями с гарантированным максимальным временем реакции	C.3.11	+	++	++	o

14	Статическое выделение ресурсов	C.2.6.3	o	+	++	++
15	Статическая синхронизация доступа к разделяемым ресурсам	C.3.5	o	o	+	++

"++" Метод строго рекомендуется для этого УПБ.
"+" Метод рекомендуется для этого УПБ.
"o" Для данного метода нет как рекомендаций, так и возражений против его использования.
✓ Удовлетворяется при помощи набора инструментов LDRA.
** Из группы 1 "Структурированные методы" следует применять метод 1a, только если метод 1b не подходит для предметной области с УПБ 3+4.

Таблица A.2 — Проектирование и разработка программного обеспечения: проектирование архитектуры программного обеспечения

Таблица (A.2) относится к разделу «Требования к проектированию архитектуры программного обеспечения» (7.4.3) и соответствующим разделам C.2, C.3 и B.2 стандарта IEC 61508 (часть 7). Они определяют, какие методы обнаружения неисправностей должны быть реализованы в отношении проектирования архитектуры программного обеспечения.

В Разделе C.3 IEC 61508 (часть 7) обсуждаются методы архитектурного проектирования, такие как обнаружение ошибок, обнаружение ошибок и программирование утверждения отказа, которые помогают в обнаружении сбоев в системе. Такие методы предназначены для выявления сбоев, что создает основу для контрмер, чтобы свести к минимуму их последствия. Функция проверки структурированного программирования тестовой среды LDRA может использоваться для идентификации неструктурированного кода, что может привести к ошибочному поведению приложения.

В разделе C.2.11 документа IEC 61508 (часть 7) упоминается о двунаправленной трассируемости между этапами жизненного цикла.

Что касается пунктов 9 и 10, TBManager создает матрицу трассируемости для обеспечения прямой и обратной трассируемости между спецификацией требований безопасности программного обеспечения и архитектурой программного обеспечения.

В разделе C.2.12 документа IEC 61508 (часть 7) упоминается архитектура программного обеспечения без учета состояния (или с ограниченными состояниями), которая требует наблюдения за входами и выходами, предоставляемыми системе.

TBrun (инструмент модульного тестирования LDRA) относится к пункту 3g тем, что он предоставляет средства для подтверждения правильного поведения системы в ответ на переданные ему параметры.

Метрики сложности, такие как Цикломатическая Сложность и метрики Холстеда, помогают определить размер программного модуля, сложность ПО и информацию о потоке данных. LDRA Testbed генерирует данные анализа потока данных и результаты проверки качества ПО, которые обеспечивают меру качества ПО. Использование динамических переменных и динамических объектов (раздел C.2.6.3 IEC 61508 (часть 7)) может быть идентифицировано во время статического анализа.

	Метод/средство	Ссылка	УПБ			
			A	B	C	D
1	Выбор соответствующего языка программирования	C.4.5	++	++	++	++
2	Строго типизированные языки программирования	C.4.1	++✓	++✓	++✓	++✓
3	Подмножество языка	C.4.2	o	o	++✓	++✓
4a	Сертифицированные средства и сертифицированные трансляторы	C.4.3	++✓	++✓	++✓	++✓
4b	Инструментальные средства, заслуживающие доверия на основании опыта использования	C.4.4	++✓	++✓	++✓	++✓

"++" Метод строго рекомендуется для этого УПБ.
"+" Метод рекомендуется для этого УПБ.
"o" Для данного метода нет как рекомендаций, так и возражений против его использования.
✓ Удовлетворяется при помощи набора инструментов LDRA.

Таблица A.3 — Проектирование и разработка программного обеспечения: инструментальные средства поддержки и языки программирования

Таблица (A.3) относится к разделам «Требования к инструментальным средствам поддержки, включая языки программирования» (7.4.4) и «C.4» IEC 61508 (часть 7), в которых указываются защитное программирование, сильная типизация и методы структурированного программирования.

В разделе 7.4.4 упоминаются требования к инструментам поддержки, включая языки программирования. Это относится к использованию инструментов верификации и валидации (раздел 7.4.4.2), статическим анализаторам кода, средствам сора покрытия и средствам управления конфигурацией.

Средства статического анализа набора инструментов LDRA предоставляют возможности для проверки соответствия стандартам программирования, таким как MISRA, CERT C и т. д., которые обнаруживают уязвимости или скрытые ошибки в исходном коде.

Возможности структурного охвата набора инструментов LDRA обеспечивают широкий диапазон показателей покрытия, включая покрытие операторов, ветвей, решений, MC/DC и LCSAJ (последовательность линейных кодов и переходов).

В разделе C.4.5 документа IEC 61508 (часть 7) рекомендуется, чтобы «выбранный язык программирования должен приводить к легко проверяемому коду с минимальными усилиями и способствовать разработке, проверке и обслуживанию программ».

Особенности, которые затрудняют верификацию и поэтому нежелательны:

- Безусловные переходы, исключая вызовы подпрограмм
- Рекурсия
- Указатели, кучи или любые типы динамических переменных или объектов
- Обработка прерываний на уровне исходного кода
- Несколько входов или выходов из циклов, блоков или подпрограмм
- Неявная инициализация переменных или их объявление
- Варианты записей и эквивалентности
- Процедурные параметры

Инструменты статического анализа LDRA проверяют исходный код (C, C ++, Ada83, Ada95 и Java), чтобы выделить такие плохие методы кодирования, которые могут привести к сбоям при реализации.

Для пакета инструментов LDRA доступен пакет поддержки квалификации инструмента, который требуется для сертификации инструмента.

	Метод/средство	Ссылка	УПБ			
			A	B	C	D
1a	Методы структурных диаграмм **	C.2.1	++✓	++✓	++✓	++✓
1b	Полуформальные методы **	Table B.7	+✓	++✓	++✓	++✓
1c	Формальные методы проектирования и усовершенствования **	C.2.4, C.2.4	o	+✓	+✓	++✓
2	Средства автоматизированного проектирования	B.3.5	+	+	++	++
3	Программирование с защитой	C.2.5	o	+✓	++✓	++✓
4	Модульный подход	Table B.9	++✓	++✓	++✓	++✓
5	Стандарты для проектирования и кодирования	C.2.6, Table B.1	+✓	++✓	++✓	++✓
6	Структурное программирование	C.2.7	++✓	++✓	++✓	++✓
7	Использование доверительных/проверенных программных модулей и компонентов (по возможности)	C.2.10	+✓	++✓	++✓	++✓
8	Прямая трассируемость между спецификацией требований к программному обеспечению	C.2.11	+✓	+✓	++✓	++✓

"++" Метод строго рекомендуется для этого УПБ.
"+" Метод рекомендуется для этого УПБ.
"o" Для данного метода нет как рекомендаций, так и возражений против его использования.
✓ Удовлетворяется при помощи набора инструментов LDRA.
** Из группы 1 "Структурированные методы" следует применять метод 1a, только если метод 1b не подходит для предметной области с УПБ 3+4.

Таблица А.4 — Проектирование и разработка программного обеспечения: детальное проектирование

Таблица (А.4) относится к разделам «Требования к детальному проектированию и разработке - проектирование системы программного обеспечения» (7.4.5), «Требования к реализации исходных текстов программ» (7.4.6) и C.2 IEC 61508 (часть 7). Они определяют меры стандартизации, соответствующие исходному коду.

Требования к безопасности программного обеспечения ожидают рассмотрения следующего этапа проектирования и разработки:

- Полнота требований к требованиям безопасности программного обеспечения
- Правильность в отношении требований к безопасности программного обеспечения
- Отсутствие ошибок, связанных с внутренним дизайном
- Простота и понятность
- Предсказуемость поведения
- Подтверждаемый и проверяемый дизайн
- Отказоустойчивость/обнаружение неисправностей
- Отсутствие общих отказов

TBManager помогает генерировать информацию о трассируемости для проверки правильности дизайна, синхронизируя спецификацию требований с набором инструментов LDRA. Здесь дизайн кода приложения можно проверить на соответствие отраслевым стандартам, которые могут быть использованы для определения сложности разработанного программного обеспечения, используемого в критически важных для безопасности приложениях.

Набор инструментов LDRA поддерживает широкий спектр стандартов промышленного кодирования, включая MISRA-C: 1998, MISRA-C: 2004, CERT C, DERA C, JSF ++ AV и т.д.

	Метод/средство	Ссылка	УПБ			
			A	B	C	D
1	Вероятностное тестирование	C.5.1	o	+✓	+✓	++✓
2	Динамический анализ и тестирование	B.6.5, Table B.2	+✓	++✓	++✓	++✓
3	Регистрация и анализ данных	C.5.2	++✓	++✓	++✓	++✓
4	Функциональное тестирование и тестирование методом "черного ящика"	B.5.1, B.5.2, Table B.3	++✓	++✓	++✓	++✓
5	Тестирование рабочих характеристик	Table B.6	+✓	+✓	++✓	++✓
6	Тестирование, основанное на модели	C.5.27	+	+	++	++
7	Тестирование интерфейса	C.5.3	+✓	+✓	++✓	++✓
8	Управление тестированием и средства автоматизации	C.4.7	+✓	++✓	++✓	++✓
9	Прямая трассируемость между спецификацией проекта программного обеспечения и спецификациями тестирования модуля и интеграции	C.2.11	+✓	+✓	++✓	++✓
10	Формальная верификация	C.5.12	o	o	+✓	+✓

"++" Метод строго рекомендуется для этого УПБ.
"+" Метод рекомендуется для этого УПБ.
"o" Для данного метода нет как рекомендаций, так и возражений против его использования.
✓ Удовлетворяется при помощи набора инструментов LDRA.

Таблица А.5 — Проектирование и разработка программного обеспечения: тестирование и интеграция программных модулей.

Таблица (А.5) относится к разделам «Требования к тестированию программных модулей» (7.4.7), «Требования к тестированию интеграции программного обеспечения» (7.4.8) и С.5 IEC 61508 (часть 7). Они определяют такие методы, как тестирование компонентов ПО и тестирование интеграции программного обеспечения, что способствует достижению безопасности программного обеспечения.

Обеспечение правильного соответствия компонента его тестовой спецификации является мероприятием верификации. Комбинация рассмотрения кода и тестирования компонента ПО верифицирует, соответствие компонента ПО его спецификации. Следующие свойства тестирования компонента ПО относятся к этой таблице:

- Полнота тестирования в отношении спецификации программного обеспечения
- Правильность тестирования в отношении спецификации программного обеспечения (успешное завершение)
- Воспроизводимость
- Точно определенная конфигурация тестирования

Tbrun автоматически генерирует тестовые драйверы и обвязки (код оболочки) без необходимости дополнительного кодирования или сценариев, что позволяет легко и эффективно проводить тесты на блоках кода. Эти тесты могут быть автоматически регрессированы с прозрачной регистрацией и хранением тестовых векторов и результатов тестирования. Процесс обслуживания тестовых векторов упрощается путем автоматического обнаружения изменений исходного кода и документации необходимых изменений. После того, как тест был запущен, пользователь может добавить информацию от трассируемости, и может сохранить тестовые вектора для переиспользования в автоматическом регрессионном тестировании.

Анализ рабочих характеристик может быть выполнен с использованием TBrun. Отчеты о динамических и регрессионных тестах могут использоваться для проверки производительности тестируемого программного обеспечения.

	Метод/средство	Ссылка	УПБ			
			A	B	C	D
1	Функциональное и черное полевое тестирование	B.5.1 B.5.2 Table B.3	++✓	++✓	++✓	++✓
2	Тестирование производительности	B.2.2, C.2.4	+✓	+✓	++✓	++✓
3	Перспективная прослеживаемость между требованиями к программному обеспечению и программному обеспечению для интеграции аппаратного и программного обеспечения и спецификаций теста интеграции оборудования / программного обеспечения	C.2.11	+✓	+✓	++✓	++✓

"++" Метод строго рекомендуется для этого УПБ.
"+" Метод рекомендуется для этого УПБ.
"o" Для данного метода нет как рекомендаций, так и возражений против его использования.
✓ Удовлетворяется при помощи набора инструментов LDRA.

Таблица A.6 — Интеграция программируемых электронных устройств (программное обеспечение и аппаратные средства)

Таблица (A.6) относится к разделам Интеграция программируемой электроники (7.5) и C.2 IEC 61508 (часть 7). Для обеспечения совместимости и соответствия требованиям предполагаемого уровня УПБ необходим подход к интеграции программного обеспечения и целевого оборудования. Это требует выполнения следующих тестов:

- Функциональные тесты
- Тесты методом "Черного ящика" (Black box), чтобы проверить динамическое поведение в реальных функциональных условиях, которые показывают сбои в выполнении функциональных спецификаций

Сюда входят данные тестирования:

- Допустимые диапазоны
- Недопустимые диапазоны
- Ограничения диапазона
- Экстремальные значения
- Комбинации вышеуказанных классов

Покрытие функциональности и кода обеспечивается как средствами модульного, так и системного тестирования, из набора инструментов LDRA. Они работают вместе: так, например, динамический системный тест может генерировать покрытие большей части исходного кода. Эти данные могут быть дополнены покрытием, генерируемым во время модульных тестов, предназначенных для исполнения конструкций кода, которые недоступны при нормальной работе, например, защитного кода. Граничные значения могут быть предоставлены для проверки допустимых и недопустимых диапазонов, а продолжительность испытаний может быть подтверждена с использованием функции анализа времени TBrun. Набор инструментов LDRA предоставляет платформу для демонстрации того, что требования к структурному покрытию были выполнены.

	Метод/средство	Ссылка	УПБ			
			A	B	C	D
1	Вероятностное тестирование	C.5.1	o	+✓	+✓	++✓
2	Моделирование процесса	C.5.18	+	+	++	++
3	Моделирование	Table B.5	+	+	++	++
4	Функциональное тестирование и тестирование методом "черного ящика"	B.5.1, B.5.2, Table B.3	++✓	++✓	++✓	++✓
5	Прямая трассируемость между спецификацией требований к программному обеспечению системы безопасности и планом подтверждения соответствия программного обеспечения системы безопасности	C.2.11	+✓	+✓	++✓	++✓
6	Обратная трассируемость между планом подтверждения соответствия программного обеспечения системы безопасности и спецификацией требований к программному обеспечению системы безопасности	C.2.11	+✓	+✓	++✓	++✓

"++" Метод строго рекомендуется для этого УПБ.
"+" Метод рекомендуется для этого УПБ.
"o" Для данного метода нет как рекомендаций, так и возражений против его использования.
✓ Удовлетворяется при помощи набора инструментов LDRA.

Таблица A.7 — Подтверждение соответствия аспектов программного обеспечения безопасности системы

Таблица (A.7) относится к разделам «Подтверждение соответствия аспектов программного обеспечения безопасности системы» (7.7) и C.2 IEC 61508 (часть 7), в которых указаны аспекты программного обеспечения для проверки безопасности системы. Это гарантирует, что интегрированная система соответствует спецификации требований безопасности программного обеспечения на требуемом уровне целостности безопасности.

Tbrun выполняет функциональное и тестирование методом "черного ящика", чтобы проверить, действуют ли функции системы или программы в соответствии с требованиями спецификации при выполнении в заданной среде в соответствии с установленными критериями.

Файл тестовых векторов (TCF), созданный TBrun, может быть сохранен и использован для автоматического анализа регрессии, чтобы подтвердить постоянное соблюдение указанных требований. TBManager дополняет этот функционал, обеспечивая трассируемость между спецификацией требований безопасности программного обеспечения и планом проверки безопасности программного обеспечения.

	Метод/средство	Ссылка	УПБ			
			A	B	C	D
1	Анализ влияния	C.5.23	++✓	++✓	++✓	++✓
2	Повторная верификация измененных программных модулей	C.5.23	++✓	++✓	++✓	++✓
3	Повторная верификация программных модулей, на которые оказывают влияние изменения в других модулях	C.5.23	+✓	++✓	++✓	++✓
4a	Повторное подтверждение соответствия системы в целом	Table A.7	o	+✓	++✓	++✓
4b	Регрессионное подтверждение соответствия	C.5.25	+✓	++✓	++✓	++✓
5	Управление конфигурацией программного обеспечения	C.5.24	++	++	++	++
6	Регистрация и анализ данных	C.5.2	++✓	++✓	++✓	++✓
7	Прямая трассируемость между спецификацией требований к программному обеспечению системы безопасности и планом модификации программного обеспечения (включая повторные верификацию и подтверждение соответствия)	C.2.11	+✓	+✓	++✓	++✓
8	Обратная трассируемость между планом модификации программного обеспечения (включая повторную верификацию и подтверждение соответствия) и спецификацией требований к программному обеспечению системы безопасности	C.2.11	+✓	+✓	++✓	++✓

"++" Метод строго рекомендуется для этого УПБ.
"+" Метод рекомендуется для этого УПБ.
"o" Для данного метода нет как рекомендаций, так и возражений против его использования.
✓ Удовлетворяется при помощи набора инструментов LDRA.

Таблица A.8 — Модификация

Таблица (A.8) относится к разделам «Модификация программного обеспечения» (7.8) и C.5 IEC 61508 (часть 7). Они определяют шаги, которые необходимо выполнить во время модификации программного обеспечения. В этих разделах содержится руководство по внедрению исправлений, усовершенствований и адаптаций проверенного программного обеспечения, обеспечивающих сохранение функций требуемого программного обеспечения.

В отношении модификации программного обеспечения следует учитывать следующие методы и меры:

- Полнота модификации в отношении ее требований
- Правильность изменения в отношении его требований
- Отсутствие введения внутренних ошибок конструкции
- Избегание нежелательного поведения
- Подтверждаемый и проверяемый дизайн
- Регрессионное тестирование и верификация

Анализ воздействия направлен на определение того, повлияло ли изменение или совершенствование системы программного обеспечения на существующую систему. После того, как анализ был завершен требуется принятие решения относительно проверки системы. На решение будет влиять количество затронутых программных модулей, критичность затронутых программных модулей и характер изменений. **Возможные решения:**

- Перепроверяется только измененный программный модуль
- Все затронутые программные модули проверяются или
- Полная система проверяется заново

TBManager создает отчеты по анализу воздействия, которые указывают, где программное обеспечение было изменено с более ранней версии.

Функция регрессионного анализа TBrun может использоваться для проверки того, повлияли ли только что введенные модули только на нужную функциональность существующей системы.

Полная система может быть перепроверена с использованием мощных методов статического анализа и динамического анализа из набора инструментов LDRA.

	Метод/средство	Ссылка	УПБ			
			A	B	C	D
1	Формальное доказательство	C.5.12	o	+✓	+✓	++✓
2	Анимация спецификации и тестирования	C.5.26	+	+	+	+
3	Статический анализ	B.6.4, Table B.8	+✓	++✓	++✓	++✓
4	Динамический анализ и тестирование	B.6.5, Table B.2	+✓	++✓	++✓	++✓
5	Прямая трассируемость между спецификацией проекта программного обеспечения и планом верификации (включая верификацию данных) программного обеспечения	C.2.11	+✓	+✓	++✓	++✓
6	Обратная трассируемость между планом верификации (включая верификацию данных) программного обеспечения и спецификацией проекта программного обеспечения	C.2.11	+✓	+✓	++✓	++✓
7	Численный анализ в автономном режиме	C.2.13	+✓	+✓	++✓	++✓
Тестирование и интеграция программного модуля		См. таблицу A.5				
Проверка интеграции программируемых электронных устройств		См. таблицу A.6				
Тестирование программной системы (подтверждение соответствия)		См. таблицу A.7				

"++" Метод строго рекомендуется для этого УПБ.
"+" Метод рекомендуется для этого УПБ.
"o" Для данного метода нет как рекомендаций, так и возражений против его использования.
✓ Удовлетворяется при помощи набора инструментов LDRA.

Таблица A.9 — Верификация программного обеспечения

Таблица (A.9) относится к разделу «Верфикация программного обеспечения» (7.9) и соответствующему разделу C.2 IEC 61508 (часть 7).

В этом подразделе рассматриваются общие аспекты верификации, которые являются общими для нескольких фаз жизненного цикла безопасности.

Формальное доказательство правильности программы связывает ее абстрактную модель с теоретическими и математическими моделями и правилами. LDRA Testbed генерирует планировщики тестов на основе математических вычислений для предоставления информации, необходимой для структурного покрытия. Также поддерживается широкий спектр стандартов программирования, которые могут использоваться для проверки правильности исходного кода.

Динамический анализ и тестирование обнаруживают сбои спецификации, проверяя динамическое поведение прототипа в продвинутом состоянии завершения. Динамический анализ системы, связанной с безопасностью, осуществляется путем проведения тестов почти рабочего прототипа системы, связанной с безопасностью, при которых вводятся данные, характерные для предполагаемой рабочей среды. Анализ является удовлетворительным, если наблюдаемое поведение системы, связанной с безопасностью, соответствует требуемому поведению.

Набор инструментов LDRA предоставляет информацию о покрытии кода, которая может быть трассируема в отношении требований.

В разделе C.2.13 документа IEC 61508 (часть 7) упоминается автономный численный анализ. При вычислении математической функции может возникать числовая погрешность вследствие использования конечных представлений идеальных функций и чисел. Тестирование методом «черного ящика» в TBrun можно использовать для проверки функциональности блоков кода, чтобы гарантировать, что входные параметры выдают ожидаемые выходы.

	Метод/средство	Ссылка	УПБ			
			A	B	C	D
1	Таблица контрольных проверок	B.2.5	+✓	+✓	+✓	+✓
2	Таблицы решений (таблицы истинности)	C.6.1	+✓	+✓	+✓	+✓
3	Анализ отказов	Table B.4	+	+	++	++
4	Анализ отказов по общей причине различного программного обеспечения (если различное программное обеспечение используется)	Table C.6.3	o	+	++	++
5	Структурные схемы надежности	C.6.4	+	+	+	+
6	Прямая трассируемость между требованиями раздела 8 и планом оценки функциональной безопасности программного обеспечения	C.2.11	+✓	+✓	++✓	++✓

"++" Метод строго рекомендуется для этого УПБ.
"+" Метод рекомендуется для этого УПБ.
"o" Для данного метода нет как рекомендаций, так и возражений против его использования.
✓ Удовлетворяется при помощи набора инструментов LDRA.

Таблица A.10 — Оценка функциональной безопасности

Таблица (A.10) относится к разделу «Оценка функциональной безопасности» (8) и соответствующему разделу C.6 IEC 61508 (часть 7).

Чтобы обеспечить проверку широких вариаций (B.2.5), большинство контрольных списков содержат вопросы, которые применимы ко многим типам систем. В результате в контрольном списке могут быть вопросы, которые не имеют отношения к рассматриваемой системе и которые следует игнорировать. В равной степени в отдельно взятой системе может возникнуть необходимость дополнить стандартный контрольный список вопросами,

относящимися к рассматриваемой системе. Матрица трассируемости требований, созданная TBManager, может использоваться для достижения прямой и обратной трассируемости как к необходимым контрольным спискам, так и к результатам верификации.

Таблицы решений и истинности (MC/DC и LCSAJ) генерируются как отчет планировщика LDRA Testbed, и требуются для процесса подтверждения соответствия программного обеспечения.

Таблицы приложения B:

	Метод/средство	Ссылка	УПБ			
			A	B	C	D
1	Использование стандартов кодирования для сокращения вероятности ошибок	C.2.6.2	++✓	++✓	++✓	++✓
2	Не использовать динамические объекты	C.2.6.3	+✓	++✓	++✓	++✓
3a	Не использовать динамические переменные	C.2.6.3	o	+✓	++✓	++✓
3b	Проверка создания динамических переменных в неавтономном режиме	C.2.6.4	o	+	++	++
4	Ограниченное использование прерываний	C.2.6.5	+✓	+✓	++✓	++✓
5	Ограниченное использование указателей	C.2.6.6	o	+✓	++✓	++✓
6	Ограниченное использование рекурсий	C.2.6.7	o	+✓	++✓	++✓
7	Не использовать неструктурированное управление в программах, написанных на языках высокого уровня	C.2.6.2	+✓	++✓	++✓	++✓
8	Не использовать автоматическое преобразование типов	C.2.6.2	+✓	++✓	++✓	++✓

"++" Метод строго рекомендуется для этого УПБ.
"+" Метод рекомендуется для этого УПБ.
"o" Для данного метода нет как рекомендаций, так и возражений против его использования.
✓ Удовлетворяется при помощи набора инструментов LDRA.

Таблица B.1 — Стандарты для проектирования и кодирования

Таблица (B.1) относится к подробным методам и средствам, указанным в таблице A.4.

Чтобы уменьшить вероятность ошибок в коде, связанном с безопасностью, и облегчить его проверку, использование стандарта кодирования является рекомендуемой практикой в IEC 61508. Набор инструментов LDRA поддерживает многие стандарты промышленного программирования, включая MISRA-C:1998, MISRA-C: 2004, JSF ++ AV, CERT C и т. д.

Код приложения может быть проверен в отношении стандартов программирования в наборе инструментов LDRA для разработки эффективной и надежной системы.

В разделе C.2.6.2 IEC 61508 (часть 7) рекомендуется использовать модульный подход, в котором могут быть определены предельные размеры программного модуля и показатели сложности программного обеспечения. Также рекомендуется внедрить метрики «понятности» кода. Это может быть реализовано с использованием функции анализа качества, доступной в LDRA Testbed, для оценки ясности, сопровождаемости и проверяемости кода.

В LDRA Testbed можно проверить использование прерываний, указателей, рекурсии и неструктурированного потока управления и проверки времени выполнения. Проверка структурированного программирования позволяет определить неструктурированные части кода приложения, которые могут привести к сбоям.

	Метод/средство	Ссылка	УПБ			
			A	B	C	D
1	Выполнение тестового вектора, связанного с анализом граничных значений[3]	C.5.4	+✓	++✓	++✓	++✓
2	Выполнение тестового вектора, связанного с предполагаемой ошибкой	C.5.5	+✓	+✓	+✓	+✓
3	Выполнение тестового вектора, связанного с введением ошибки	C.5.6	o	+✓	+✓	+✓
4	Выполнение тестового вектора, сгенерированного на основе модели	C.5.27	+✓	+✓	++✓	++✓
5	Моделирование реализации	C.5.20	+✓	+✓	+✓	++✓
6	Разделение входных данных на классы эквивалентности	C.5.7	+✓	+✓	+✓	++✓
7	Структурный тест со 100%-ным покрытием (точки входа) **	C.5.8	++✓	++✓	++✓	++✓
7b	Структурный тест со 100%-ным покрытием (операторы)**	C.5.8	+✓	++✓	++✓	++✓
7c	Структурный тест со 100%-ным покрытием (условные переходы) **	C.5.8	+✓	+✓	++✓	++✓
7d	Структурный тест со 100%-ным покрытием (составные условия, MC/DC) **	C.5.8	+✓	+✓	+✓	++✓

"++" Метод строго рекомендуется для этого УПБ.
"+" Метод рекомендуется для этого УПБ.
"o" Для данного метода нет как рекомендаций, так и возражений против его использования.
✓ Удовлетворяется при помощи набора инструментов LDRA.
** Если 100%-ное покрытие не может быть достигнут (например, покрытие оператора кода защиты), то должно быть дано соответствующее объяснение

Таблица B.2 — Динамический анализ и тестирование

Таблица (B.2) относится к подробным методам/средствам, указанным в таблицах A.5 и A.9.

В разделе C.5.4 IEC 61508 (часть 7) обсуждается использование анализа граничных значений, который используется для обнаружения ошибок программного обеспечения, возникающих при ограничении или границах параметров. В TBrun может быть задан диапазон входных значений, а также переменные граничные значения и проверены на ожидаемые результаты. Например, вероятность ошибки деления на ноль может быть задокументирована на этапе динамического тестирования.

[3] Термин по ГОСТ Р МЭК 61508, «тестовый пример» здесь и далее заменен эквивалентным термином «тестовый вектор»

В разделе C.5.4 IEC 61508 (часть 7) рассматривается погрешность ошибок. В этом методе могут быть предусмотрены специальные значения или комбинации значений для проверки того, является ли поведение кода подверженным ошибкам. Тестирование методом «черного ящика» с использованием TBrun помогает сравнить фактические результаты с ожидаемыми результатами.

В разделе C.5.7 IEC 61508 (часть 7) рассматриваются классы эквивалентности и тестирование входных разделов, описывающее, как правильно тестировать программное обеспечение с использованием минимальных тестовых данных. Анализ профиля в LDRA Testbed помогает идентифицировать любые избыточные тестовые данные, а анализ набора данных в LDRA Testbed помогает определить, какие тестовые данные влияют на данные покрытия.

В разделе C.5.8 IEC 61508 (часть 7) рассматривается структурное тестирование, то есть «На основе анализа программы выбирается набор входных данных так, чтобы мог быть проанализирован достаточно большой (часто с заранее заданным значением) процент программных кодов. Меры покрытия[4] программы, в зависимости от степени требуемой строгости, могут быть различными. В любом случае целью должны быть 100% выбранной метрики покрытия; если невозможно достигнуть 100%-ного покрытия, то причины, почему 100%-ный покрытие не может быть достигнуто, должны быть документально оформлены в отчете о тестировании (например если возникает аппаратная проблема, то может быть введен только код защиты).»

В соответствии с уровнями УПБ Структурное покрытие может быть классифицировано как:

- **УПБ 4**
  - Структурное покрытие (точки входа) = 100%
  - Структурное покрытие тестами (операторы) = 100%
  - Структурное покрытие тестами (ветви) = 100%
  - Структурное покрытие (условия, MC/DC) = 100%
- **УПБ 3**
  - Структурное покрытие (точки входа) = 100%
  - Структурное покрытие тестами (операторы) = 100%
  - Структурное покрытие тестами (ветви) = 100%
- **УПБ 2**
  - Структурное покрытие (точки входа) = 100%
  - Структурное покрытие тестами (операторы) = 100%
- **УПБ 1**
  - Структурное покрытие (точки входа) = 100%

Анализ динамического покрытия LDRA помогает достичь необходимого уровня покрытия для соответствия IEC 61508. Также доступен пакет поддержки квалификации для сертификации набора инструментов LDRA для текущего проекта до требуемых уровней УПБ.

[4] Термин по ГОСТ Р МЭК 61508, «охват» здесь и далее заменен эквивалентным термином «покрытие»

	Метод/средство	Ссылка	УПБ			
			A	B	C	D
1	Выполнение тестового вектора, на основе причинно-следственных диаграмм	B.6.6.2	o	o	+	+
2	Выполнение тестового вектора, сгенерированного на основе модели	C.5.27	+	+	++	++
3	Прототипирование[5]/анимация	C.5.17	o	o	+	+
4	Разделение входных данных на классы эквивалентности, включая анализ граничных значений	C.5.7, C.5.4	+✓	++✓	++✓	++✓
5	Моделирование процесса		+	+	+	+

"++" Метод строго рекомендуется для этого УПБ.
"+" Метод рекомендуется для этого УПБ.
"o" Для данного метода нет как рекомендаций, так и возражений против его использования.
✓ Удовлетворяется при помощи набора инструментов LDRA.

Таблица B.3 — Функциональное тестирование и проверка методом "черного ящика»

Таблица (B.3) относится к подробным методам/средствам, указанным в таблицах A.5, A.6 и A.7.

В этом разделе разъясняется, что анализ тестовых векторов находится на уровне программной системы и основывается только на спецификации.

В разделе C.5.27 IEC 61508 (часть 7) рассматривается автоматическое создание тестовых векторов из системных моделей и создание высоко повторяемых наборов тестов. Набор инструментов LDRA интегрирован с инструментами моделирования, и тестовые вектора могут быть реализованы непосредственно в TBrun, с возможностью установления трассируемости с использованием TBManager. TBeXtreme обеспечивает автоматическую генерацию тестовых векторов из исходного кода. Записанные тестовые вектора могут быть непосредственно связаны как с документом требований, так и с кодом приложения для создания матрицы трассируемости.

	Метод/средство	Ссылка	УПБ			
			A	B	C	D
1a	Причинно-следственные диаграммы	B.6.6.2	+	+	+	+
1b	Анализ методом дерева событий	B.6.6.3	+	+	+	+
2	Анализ методом дерева отказов	B.6.6.5	+	+	+	+
3	Анализ функциональных отказов программного обеспечения	B.6.6.4	+	+	+	+

"++" Метод строго рекомендуется для этого УПБ.
"+" Метод рекомендуется для этого УПБ.
"o" Для данного метода нет как рекомендаций, так и возражений против его использования.
✓ Удовлетворяется при помощи набора инструментов LDRA.

Таблица B.4 — Анализ отказов

[5] Термин по ГОСТ Р МЭК 61508, «макетирование» здесь и далее заменен эквивалентным термином «прототипирование»

Таблица (B.4) относится к подробным методам/мерам, указанным в таблице A.10. Этот раздел может быть рассмотрен с использованием инструментов анализа надежности.

TBManager может установить информацию о трассируемости, сгенерированную этими инструментами в формате MS Word, чтобы обеспечить трассируемость между требованиями и анализом сбоев.

	Метод/средство	Ссылка	УПБ			
			A	B	C	D
1	Диаграммы потоков данных	C.2.2	+✓	+✓	+✓	+✓
2a	Метод конечных автоматов	B.2.3.2	o	+	++	++
2b	Формальные методы	B.2.2, C.2.4	o	+✓	+✓	++✓
2c	Моделирование во времени сетями Петри	B.2.3.3	o	+	++	++
3	Моделирование реализации	C.5.20	+	++	++	++
4	Прототипирование/анимация	C.5.17	+	+	+	+
5	Структурные диаграммы	C.2.3	+✓	+✓	+✓	++✓

"++" Метод строго рекомендуется для этого УПБ.
"+" Метод рекомендуется для этого УПБ.
"o" Для данного метода нет как рекомендаций, так и возражений против его использования.
✓ Удовлетворяется при помощи набора инструментов LDRA.

Таблица B.5 — Моделирование

Таблица (B.5) относится к подробным методам/мерам, указанным в таблице A.7.

В разделе C.2.2 IEC 61508 (часть 7) описывается, как ввод данных преобразуется в вывод, причем каждый этап диаграммы представляет собой явное преобразование. Графы вызовов и потоков выполнения, сгенерированные LDRA Testbed, предоставляют информацию о потоке данных.

Формальные методы были реализованы в наборе инструментов LDRA. Они используются для генерации отчетов об анализах процедур и интерфейсов, обеспечивая информацию о зависимостях между компонентами.

Структурные диаграммы аналогичны диаграммам потоков данных, и представляют системные зависимости.

	Метод/средство	Ссылка	УПБ			
			A	B	C	D
1	Проверка на критические нагрузки и стресс-тестирование	C.5.21	+✓	+✓	++✓	++✓
2	Ограничения на время ответа и объем памяти	C.5.22	++✓	++✓	++✓	++✓
3	Требования к реализации	C.5.19	++✓	++✓	++✓	++✓

"++" Метод строго рекомендуется для этого УПБ.
"+" Метод рекомендуется для этого УПБ.
"o" Для данного метода нет как рекомендаций, так и возражений против его использования.
✓ Удовлетворяется при помощи набора инструментов LDRA.

Таблица B.6 — Тестирование рабочих характеристик

Таблица (B.6) относится к подробным методам/мерам, указанным в таблицах A.5 и A.6.

В разделе C.5.21 IEC 61508 (часть 7) рассматриваются проверки на критические нагрузки и стресс-тестирование. Стресс-тестирование относится к испытаниям, которые определяют надежность программного обеспечения путем тестирования за пределами нормальной работы. Стресс-тестирование особенно важно для критически важного программного обеспечения и сродни покрытию ветвей, что требует не только обычных положительных тестовых векторов, доказывающих ожидаемую функциональность, но и отрицательных тестовых векторов, когда программное обеспечение должно каким-то образом отказать.

Примером отрицательного вектора будет вызов функции с неправильными параметрами. TBrun можно использовать для проверки функций с граничными входными значениями или путем передачи неправильных параметров для проверки надежности программного обеспечения.

В разделе C.5.22 IEC 61508 (часть 7) указывается использование временного анализа. TBrun можно использовать для проверки времени выполнения определенного модуля кода или компонента.

	Метод/средство	Ссылка	УПБ			
			A	B	C	D
1	Логические/функциональные блок-схемы	IEC 61131-3	+✓	+✓	++✓	++✓
2	Диаграммы последовательности действий	IEC 61131-3	+	+	++	++
3	Диаграммы потоков данных	C.2.2	+✓	+✓	+✓	+✓
4a	Конечные автоматы/диаграммы переходов	B.2.3.2	+	+	++	++
4b	Моделирование во времени сетями Петри	B.2.3.3	+	+	++	++
5	Модели данных сущность-связь-атрибут	B.2.4.4	+✓	+✓	+✓	+✓
6	Диаграммы последовательности сообщений	C.2.14	+	+	+	+
7	Таблицы решений и таблицы истинности	C.6.1	+✓	+✓	+✓	+✓
8	UML	C.3.12	+	+	+	+

"++" Метод строго рекомендуется для этого УПБ.
"+" Метод рекомендуется для этого УПБ.
"o" Для данного метода нет как рекомендаций, так и возражений против его использования.
✓ Удовлетворяется при помощи набора инструментов LDRA.

Таблица B.7 — Полуформальные методы

Таблица (B.7) относится к подробным методам/мерам, указанным в таблицах A.1, A.2 и A.4.

Графы вызовов и потоков выполнения, сгенерированные LDRA Testbed, графически представляют программные функциональные блоки, потоки управления и информацию о потоке данных. В диаграммах потоков данных описывается, как ввод данных преобразуется в вывод, на каждом этапе диаграммы, представляющей собой явное преобразование. Аннотированный исходный код предоставляет информацию о том, как данные передаются между каждым модулем.

Таблицы решений и истинности дают четкую и согласованную спецификацию и анализ сложных логических комбинаций и отношений. Этот метод использует двумерные таблицы для краткого описания логических отношений между булевыми переменными. Планировщик тестов MC/DC обеспечивает руководство по выбору соответствующих тестовых значений с целью достижения 100% покрытия MC/DC.

	Метод/средство	Ссылка	УПБ			
			A	B	C	D
1	Анализ граничных значений	C.5.4	+✓	+✓	++✓	++✓
2	Таблица контрольных проверок	B.2.5	+✓	+✓	+✓	+✓
3	Анализ потоков управления	C.5.9	+✓	++✓	++✓	++✓
4	Анализ потоков данных	C.5.10	+✓	++✓	++✓	++✓
5	Предположение ошибок	C.5.5	+✓	+✓	+✓	+✓
6a	Формальные проверки, включая конкретные	C.5.14	+✓	+✓	++✓	++✓
6b	Сквозной контроль (программного обеспечения)	C.5.15	+✓	+✓	+✓	+✓
7	Тестирование на символьном уровне	C.5.11	o	o	+	+
8	Анализ проекта	C.5.16	++✓	++✓	++✓	++✓
9	Статический анализ выполнения программы с ошибкой	B.2.2, C.2.4	+✓	+✓	+✓	++✓
10	Временной анализ выполнения при наихудших условиях	C.5.20	+	+	+	+

"++" Метод строго рекомендуется для этого УПБ.
"+" Метод рекомендуется для этого УПБ.
"o" Для данного метода нет как рекомендаций, так и возражений против его использования.
✓ Удовлетворяется при помощи набора инструментов LDRA.

Таблица B.8 — Статический анализ

Таблица (B.8) относится к подробным методам/мерам, указанным в таблице A.9.

В разделах B.2.2 и C.2.4 IEC 61508 (часть 7) описывается статический анализ поведения среды времени исполнения. **Набор инструментов LDRA статически проверяет наличие проблем времени исполнения, включая:**

- Деление на ноль
- Индексацию за пределы массива
- Неинициализированную локальную переменную
- Некорректное разыменование указателя
- Неинициализированный указатель
- Неинициализированную переменную
- Бесконечный цикл
- Вызов без выхода

Формальная проверка - это структурированный процесс для проверки кода программного обеспечения, которые выполняются экспертами, чтобы найти дефекты и дать возможность производителю улучшить код. Критерии входа и выхода должны определяться на основе свойств, необходимых для программного элемента. Информация о покрытии кода в наборе инструментов LDRA содержит информацию о критериях входа и выхода.

Символьное выполнение, как описано в разделе C.5.11 IEC 61508 (часть 7), можно проверить, предоставив ожидаемое значение в тестовых векторах с использованием TBrun с помощью Active Graphs.

	Метод/средство	Ссылка	УПБ			
			A	B	C	D
1	Ограничение размера программного модуля	C.2.9	++✓	++✓	++✓	++✓
2	Управление сложностью программного обеспечения	C.5.13	+✓	+✓	++✓	++✓
3	Ограничение доступа/инкапсуляция информации	C.2.8	+	++	++	++
4	Ограниченное число параметров/фиксированное число	C.2.9	+✓	+✓	+✓	+✓
5	Одна точка входа и одна точка выхода в каждой подпрограмме и функции	C.2.9	++✓	++✓	++✓	++✓
6b	Полностью определенный интерфейс	C.2.9	++✓	++✓	++✓	++✓

"++" Метод строго рекомендуется для этого УПБ.
"+" Метод рекомендуется для этого УПБ.
"o" Для данного метода нет как рекомендаций, так и возражений против его использования.
✓ Удовлетворяется при помощи набора инструментов LDRA.

Таблица B.9 — Модульный подход

Таблица (B.9) относится к подробным методам/мерам, указанным в таблице A.4.

Функционал проверки качества, доступная в тестовом LDRA Testbed, проверяет программное обеспечение на большое количество показателей сложности, которые перечислены ниже:

- Цикломатическая сложность МакКейба
- Существенная цикломатическая сложность
- Узлы
- Существенные узлы
- Уровень вложенности
- Метрики размера Холстеда
- Недостижимый код и мертвый код
- Неподходящий код
- Плотность LCSAJ
- Метрики C ++ OO
- Точки входа/выхода
- Процедуры/интерфейсы

Вывод

Из этого документа видно, что набор инструментов LDRA может внести значительный вклад в достижение сертификационного зачета по стандарту IEC 61508.Он предоставляет подходящие средства для любого УПБ, а зрелость инструмента и долговечность являются непревзойденными.

Набор инструментов LDRA широко используется во всем мире, особенно в средах, связанных с безопасностью и надежностью. Он используется для облегчения сертификации программного обеспечения по многим международным стандартам, включая IEC 61508, IEC 62304, ISO/DIS 26262 и DO-178B.

Доступность

Информацию о доступности пакета инструментов LDRA см. на веб-сайте LDRA или обратитесь в LDRA.

Проверенное соответствие отраслевым стандартам

LDRA имеет длинный послужной список по обеспечению соблюдения стандартов в области программирования, а также является ключевым соавтором стандартов MISRA C и C ++. LDRA играет активную роль в комитете MISRA C ++, и представлена в комитете его членами и председателем. LDRA также представлена в комитете MISRA C тремя членами технической команды LDRA. Неудивительно, что набор инструментов LDRA обеспечивает наиболее полное внедрение стандартов кодирования C и C ++ из всех доступных на рынке инструментов.

Набор инструментов LDRA используется компаниями по всему миру для соблюдения многочисленных стандартов, включая IEC 61508. Инструмент используется в проектах критически важных по отношению к безопасности более двадцати лет.

Гибкая поддержка инструментов

Поскольку сертифицирующие органы приобретают дополнительный опыт применения IEC 61508 для проектов, важно, чтобы поставщики инструментов имели гибкость, чтобы справляться с меняющимися требованиями. LDRA стремится помогать существующим клиентам соответствовать изменяющимся требованиям сейчас и в будущем.

Простота использования

Простота использования - это ключевая проблема при создании процедур проекта. Набор инструментов LDRA специально разработан для обеспечения простого измерения соответствия различным классам IEC 61508. Отчеты предназначены для быстрого и точного предоставления пользователям информации об IEC 61508, ускоряя процедуру тестирования. Отчеты могут быть созданы в форматах ASCII или HTML. Любой формат может быть легко включен в текстовый процессор или систему DTP. HTML имеет дополнительное преимущество в виде гиперссылок и возможности публикации отчетов в интрасети.

Набор инструментов LDRA используется следующими компаниями:

Продукты и услуги LDRA широко используются компаниями, имена которых являются синонимами промышленных устройств безопасности, включая Aerosystems Intl Ltd, Analox Cyberdyne Inc, General Dynamics, GE, Khrohne, MBDA, Mitsubishi Electronic Corporation, QinetiQ и Serco.

+7 (495) 009-65-85
info@exponenta.ru
ldra.exponenta.ru