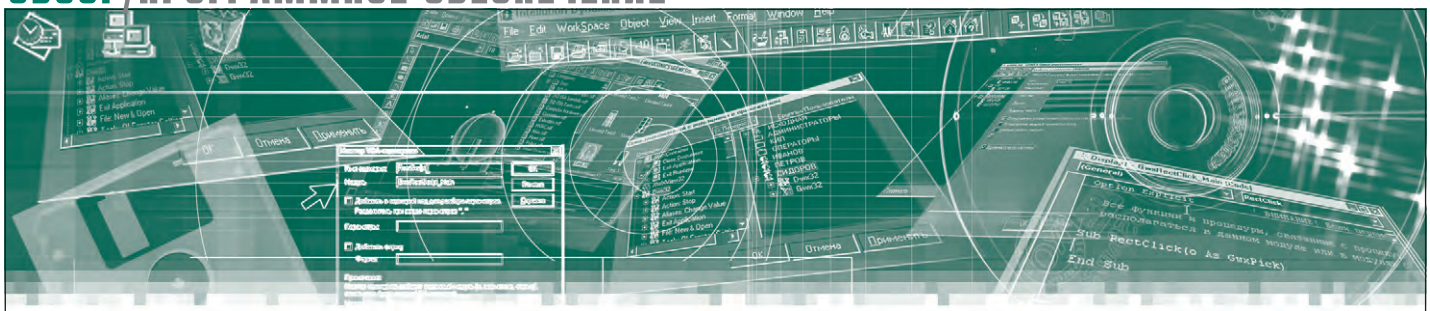


Николай Горбунов

# Безопасность и сертификация программного обеспечения

## Часть 3. Стоимость сертификации и подходы к её сокращению

**В статье приводится обзор современной терминологической и нормативно-технической базы функциональной и информационной безопасности ПО, затрагивается ряд основополагающих вопросов качества ПО и их привязка к нормативной базе. Рассматриваются примеры программных продуктов, соответствующих современным требованиям сертификации, и практические подходы к подтверждению соответствия. В третьей части речь идёт о возможных подходах к снижению стоимости сертификации.**

Если даже бегло оценить объём работ по сертификации авиационного ПО на соответствие DO-178B, то волосы встают дыбом. По данным совместного исследования компании HighRely [1] и промышленной группы DO-178 (DO-178 Industry Group) [2], проведённого в 2010 году, соблюдение требований DO-178B способно увеличить стоимость разработки на 25–100% (эти цифры делятся на два диапазона: 25–40% для успешных проектов и 60–100% для типичных), и это уже при поставленных процессах и опытной команде. Публикуемая на настоящий момент в открытых источниках статистика сертификации авиационного ПО на соответствие DO-178B даёт оценку стоимости сертификационных работ *до сотен долларов США за строку кода*. Учитывая, какими темпами растёт объём встраиваемого ПО в последние десятилетия (объём кода авионики истребителя Lockheed Martin F-35 “Lightning II” составляет *6,8 млн строк* – это *в 50 раз больше*, чем у General Dynamics F-16 “Fighting Falcon”, выпущенного всего 30 лет назад, – рис. 1, [3]), стоимость сертификации начинает становиться серьёзной проблемой, и разработчики встраиваемого ПО во всём мире сейчас активно ищут способы её сокращения.

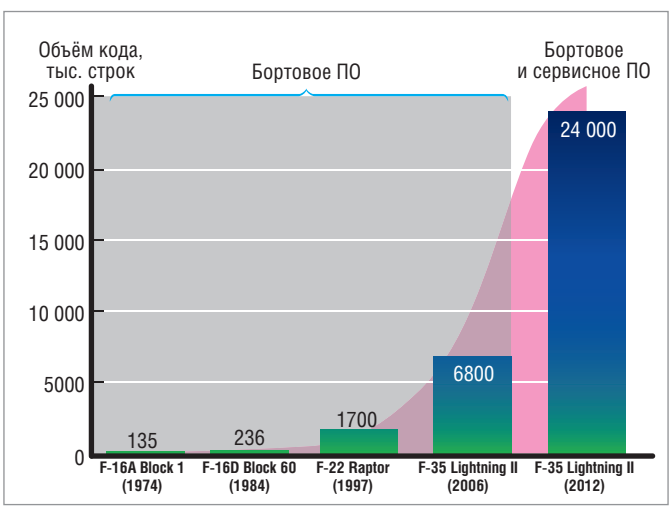



**Рис. 1. Рост объёмов кода ПО серийного истребителя за последние десятилетия**

С точки зрения разработчика, навскидку напрашиваются три способа сокращения трудозатрат на сертификацию:

- упрощение процесса создания и сопровождения сертификационной документации;
- декомпозиция ПО по разным уровням безопасности и сертификация их по отдельности;
- упрощение процесса взаимодействия с аудитором (оценщиком); этот вопрос выходит за рамки настоящего материала и поэтому более подробно не рассматривается.

Создание и сопровождение сертификационной документации объединены в один пункт не случайно. Любой программный проект – живая субстанция в динамичном мире. Невозможно разработать и задокументировать продукт один раз и навсегда, поскольку мир вокруг постоянно меняется, а значит, и в требования, код и тестовые сценарии тоже постоянно вносятся изменения. Поскольку между всеми проектными документами существует взаимосвязь, любое (даже точечное) изменение порождает множество связанных правок, и упустить что-то означает нарушить целостность. Систематическое наруше-

ние целостности, в свою очередь, приводит к тому, что требования, код и тестовые сценарии «разъезжаются» настолько, что ни самому понять, ни аудитору продемонстрировать. Грамотно же поставленный и до возможной степени автоматизированный процесс управления изменениями обеспечит минимум усилий по формированию непротиворечивого пакета сертификационных документов на каждой итерации. Одно это, по статистике компании LDRA [4], уже способно сократить трудозатраты в 10–15 раз.

Подход с декомпозицией по уровням безопасности стал набирать вес в последнее время в свете растущей тенденции к объединению разнородных вычислительных подсистем на единой аппаратной платформе (с целью сокращения массы, габаритов и потребляемой мощности). Поскольку к различным подсистемам при этом могут предъявляться различные требования по безопасности (как функциональной, так и информационной), в зарубежной терминологии для таких систем даже появился специальный термин – «системы смешанной критичности» (mixed criticality systems). Казалось бы, решая одну аппаратную проблему, с точки зрения ПО такой подход создаёт другую: в общем случае объединение на одной аппаратной платформе нескольких программных приложений различного уровня безопасности потребовало бы сертификации по самому высокому из требуемых уровней. Однако на настоящий момент уже существуют технологии, позволяющие разбить программную систему на независимые разделы и сертифицировать ПО этих разделов по отдельности (мы к этому ещё вернёмся). Как было показано ранее, с ростом требований к безопасности ПО объём необходимой сертификационной документации ощутимо растёт, но и критичного кода в системе обычно значительно меньше, чем некритичного, так что грамотное разбиение системы на разделы может дать существенную экономию трудозатрат.

Рассмотрим этот процесс подробнее, в частности, из чего состоит подлежащее сертификации ПО и где и как можно сэкономить.

## Состав программного стека, и что из него следует

Программный стек в общем случае всегда выглядит одинаково (рис. 2). На самом верху располагается прикладное

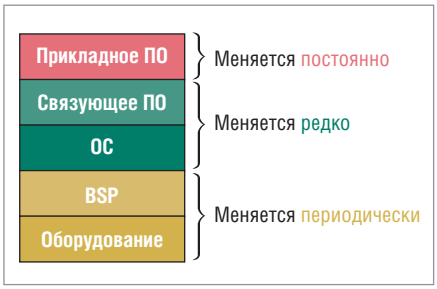



**Рис. 2. Структура программного стека в контексте сертификации**

ПО, реализующее проектно-специфичную логику – это самая важная и самая изменчивая часть; всё остальное нужно, чтобы обеспечить её работу. Ниже прикладного ПО располагаются компоненты, обеспечивающие ему необходимые сервисы – аппаратно-независимая часть ОС (или того, что играет роль ОС) и так называемое связующее ПО (СУБД, специфичные стеки протоколов и т.п.), реализующее сервисы, средствами ОС не предусмотренные. Связующее ПО использует сервисы ОС (распределение памяти, средства межзадачного взаимодействия и синхронизации, таймеры и т.п.) наравне с прикладным. В самом низу располагается аппаратно-зависимая часть ОС (так называемый *пакет поддержки оборудования*, board support package – BSP), которая отвечает за взаимодействие ОС с аппаратными средствами (забегая вперёд, можно сказать, что между ОС и BSP в этой картине может располагаться ещё и *гипервизор*, но об этом чуть позже.)

С точки зрения сертификации, самое главное в этой картинке – какие компоненты подвержены изменениям и с какой частотой, так как сертификационную документацию для относительно неизменных компонентов можно разработать один раз (а значит, не уделяя особого внимания автоматизации), в то время как для изменчивых компонентов она будет постоянно корректироваться – именно здесь и будет потрачено максимальное количество человеко-часов. Применительно к нашей модели:

- прикладное ПО всегда специфично для конкретного проекта (хотя и может содержать унаследованный код), а также подвержено текучке требований, и поэтому меняется постоянно;
- пакет BSP всегда привязан к аппаратуре, и поэтому может быть специфичен для группы проектов, использующих одно и то же оборудование, то есть меняется редко;
- связующее ПО и аппаратно-независимая часть ОС могут применяться в различных проектах в коробочном виде, и поэтому в идеале не подвержены изменениям вообще (по крайней мере, до выхода следующей версии, на которую имеет смысл переходить).

Теперь посмотрим, какие есть варианты.

## «Ручной» подход

Самым распространённым подходом к разработке сертификационной документации (по крайней мере, в отечественной практике), к сожалению, является разработка и сопровождение сертификационной документации для всего программного стека вручную. Этот подход способен дать значительный выигрыш на первой итерации (особенно притягательными для высшего менеджмента обычно кажутся нулевые подготовительные затраты), но, увы, только на первой, так как, что уже обсуждалось, программный проект никогда не ограничивается одной итерацией. Каждое вносимое в проект изменение (особенно это касается уровня прикладного ПО) будет требовать множества связанных правок в пакете сертификационных документов. Это чревато не только катастрофическим разрастанием трудозатрат по мере взросления проекта, но и нарушением целостности сертификационного пакета из-за накопления человеческих ошибок, что неизбежно ведёт к проблемам в процессе подтверждения соответствия и в конечном итоге к переделыванию всей работы заново.

Единственная часть программного стека, где определённая доля ручной работы не только оправданна, но и необходима, – это BSP. В частности, структурное тестирование BSP очень трудно поддаётся автоматизации, так как выполнение аппаратно-зависимого кода жёстко привязано к временны́м характеристикам оборудования, и инструментирование такого кода в автоматическом режиме может сделать его неработоспособным.

## Купить нельзя построить: применение сертифицированных и сертифицируемых компонентов

Одним из способов сократить трудозатраты по созданию и сопровождению сертификационной документации является применение коммерческих (Com-

mercial Off-The-Shelf – COTS) сертифицированных (или сертифицируемых – о разнице между этими понятиями уже говорилось ранее) программных компонентов.

Смысл этого подхода в том, что некоторые части программного стека (например, ОС и связующее ПО) остаются неизменными достаточно долгое время, и вместо того чтобы разрабатывать и сопровождать их (а также сопутствующую сертификационную документацию) самостоятельно, может оказаться гораздо дешевле (а главное, быстрее) приобрести готовый коммерческий продукт, уже имеющий необходимый сертификат или снабжённый сертификационным пакетом. Это не отменит необходимости в подготовке сертификационной документации для остальных компонентов (например, BSP и прикладного ПО), но существенная часть работы будет сделана – по статистике компании Wind River, применение сертифицируемой ОС в сочетании с коммерческим сертификационным пакетом способно сократить трудозатраты *на 2 человеко-года и более*.

У этого подхода, правда, есть три тонкости.

**Во-первых**, поскольку требования безопасности налагают ограничения на архитектуру ПО, а стоимость сертификации напрямую зависит от объёма подлежащего сертификации кода, далеко не любой программный продукт является в принципе пригодным для сертификации. Чтобы пройти сертификацию, продукт должен быть разработан определённым образом, иметь определённое внутреннее устройство (в соответствии с требованиями нормативной базы, о чём говорилось ранее) и быть по возможности более компактным (чтобы стоимость сертификации не получилась неоправданно высокой). Этим, кстати, объясняется чрезвычайно низкий уровень присутствия ОС Linux в системах с высокими требованиями к функциональной безопасности – Linux без существенных доработок не проходит ни по требованиям стандартов, ни по стоимости сертификационных работ.

**Во-вторых**, не всякий пригодный к сертификации программный продукт имеет сертификаты (или сертификационные пакеты) по всем возможным стандартам функциональной и информационной безопасности – это было бы слишком дорого. Производители продуктов обычно сами решают, на каких рынках фокусироваться, и затем уже обеспечивают своим продуктам необходимую базу для отраслевой сертификации. Если продолжать говорить об ОС, то абсолютный чемпион в этом смысле из представленных в России – ОС VxWorks компании Wind River, имеющая сертификационные пакеты DO-178B/C до уровня A включительно и МЭК 15408 до уровня EAL 6+ (согласно [5] версии 1.3), а также сертификат МЭК 61508 до уровня SIL 3 включительно. На базе сертификационного пакета МЭК 61508 для VxWorks в последние годы компанией Wind River был также выполнен ряд железнодорожных проектов с сертификацией по EN 50128 вплоть до уровня SIL 4. Почти аналогичный набор (за исключением DO-178B/C) есть у широко распространённой в России ОС QNX Neutrino, кроме того, её защищённая версия (ЗОСРВ КПДА 10964-01 «Нейтрино») также имеет отечественные сертификаты информационной безопасности в системах сертификации ФСТЭК и МО. Сертифицированные ОС на базе Linux (например, российская Astra Linux Special Edition) представляют собой противоположный экстремум – сертификаты информационной безопасности у них есть, а функциональной – нет.

**В-третьих**, применение коммерческих сертифицированных и сертифицируемых программных компонентов всегда сопряжено с оговорками, так как сертификация «сферического компонента в вакууме» – это одно, а интеграция его в сложную многокомпонентную систему – совсем другое. Также важен вопрос взаимного доверия оценщиков: стандарты функциональной и информационной безопасности говорят об этом разными словами, но практический смысл очень схож: наличие сертификата у коммерческого программного компонента ещё не гарантирует положительный вердикт оценщика, и может потребоваться повторная оценка. Хорошая новость, правда, заключается в том, что если есть сертификат, то есть и сертификационная документация, то есть повторную оценку безопасности не придётся делать с нуля, а значит, она обойдётся дешевле.

Ну и, естественно, данный подход работает только для компонентов программного стека, не подверженных частым изменениям (читай – ОС и связующего ПО). Для остальных компонентов приходится искать другие способы сокращения сертификационных трудозатрат.

## Автоматизируй это: инструментарий верификации и валидации

Большинство методик обеспечения качества ПО, описанных в статье, являются в той или иной степени автоматизируемыми. То есть понятно, что нельзя автоматизировать процесс, скажем, написания тестов, но процесс генерации «обёртки» (test harness) и входных данных для тестирования автоматизировать можно. Также можно автоматизировать процесс статического анализа, а его выходные результаты, в свою очередь, автоматически подать на вход процесса анализа структурного покрытия, и так далее. Потом по результатам выполнения всех необходимых задач можно автоматически сгенерировать отчёт, оформленный в соответствии с требованиями нужного стандарта, и подшить его к сертификационной документации. Основная ценность такого подхода в том, что он позволяет, однажды настроив процесс генерации документов, повторять его по необходимости в автоматическом режиме. Это обеспечивает непрерывную целостность сертификационного пакета и предохраняет от человеческих ошибок.

В литературе, посвящённой качеству ПО, постоянно фигурируют два термина – «*верификация*» (verification) и «*валидация*» (validation). Их часто путают друг с другом, в основном благодаря тому, что их классические определения безобразно неоднозначны, и поэтому разные источники относят к ним различные конкретные действия (например, в терминах DO-178B почти все описанные в настоящей статье методики относятся к верификации). Впрочем, хорошая новость заключается в том, что эти термины почти всегда употребляют вместе (видимо, чтобы не попасть впросак), поэтому достаточно сказать «верификация и валидация» (V&V) – и точно не ошибёшься.

Так вот, существует целый класс программных продуктов, автоматизирующих задачи V&V, их так и называют – инструменты верификации и валидации (verification and validation tools). К таким инструментам относятся средства статического анализа, анализа покрытия (ряд производителей встраиваемых ОС даже включает их в дистрибутив комплекта разработчика), автоматизированного тестирования и т.п. Количество таких инструментов на рынке исчисляется десятками; однако каждый

ADVANTIX intellect

# ОТКАЗОУСТОЙЧИВОЕ РЕШЕНИЕ ДЛЯ КРИТИЧЕСКИ ВАЖНЫХ ПРИЛОЖЕНИЙ

intel Technology Provider Platinum 2015

## КЛЮЧЕВЫЕ ПРЕИМУЩЕСТВА

- «Нулевое» время простоя — обеспечение непрерывности работы приложений без потери данных и транзакций
- «Нулевое» администрирование — решение является простым в эксплуатации и не требует высоких затрат на обслуживание
- Предотвращает простои, а не просто выполняет восстановление после сбоев
- Уровень доступности 99,999%, что соответствует 5,25 минуты простоя в год

## AdvantiX Intellect FT BOX

Microsoft Linux SCADA

WWW.ADVANTIX-PC.RU

ОФИЦИАЛЬНЫЙ ДИСТРИБЬЮТОР ПРОДУКЦИИ ADVANTIX

ProSoft®

**МОСКВА** Тел.: (495) 234-0636 • Факс: (495) 234-0640 • info@prosoft.ru • www.prosoft.ru
**С.-ПЕТЕРБУРГ** Тел.: (812) 448-0444 • Факс: (812) 448-0339 • info@spb.prosoft.ru • www.prosoft.ru
**ЕКАТЕРИНБУРГ** Тел.: (343) 376-2820 • Факс: (343) 310-0106 • info@prosoftsystems.ru • www.prosoftsystems.ru

Реклама

из них решает только свою часть задачи, поэтому основная цель – поддержание целостности сертификационной документации – в случае использования разрозненных инструментов не достигается. К тому же разрозненные инструменты зачастую не привязаны к конкретным стандартам, а значит, следить за тем, чтобы созданные сертификационные документы содержали подтверждение выполнения всех нормативных требований, в этом случае тоже нужно вручную.

Гораздо более привлекательным вариантом, с точки зрения снижения стоимости разработки и сопровождения сертификационной документации, являются интегрированные программные пакеты, изначально ориентированные на решение задач подтверждения соответствия. Хороший пример такого пакета – LDRA Tool Suite компании LDRA, содержащий готовые решения по подготовке сертификационной документации согласно DO-178B/C, МЭК 61508, EN 50128, МЭК 26262 и МЭК 62304 (а в ближайшей перспективе ещё и МЭК 60880). В состав пакета входят:

- **шаблоны и примеры проектов** подтверждения соответствия по всем поддерживаемым стандартам. Иными словами, любой проект в LDRA Tool Suite всегда начинается с выбора целевого стандарта, соответствие которому нужно продемонстрировать, из этого сразу следует актуальный список конкретных задач;
- средства **интеграции с системами управления требованиями и трассировки требований** (рис. 3). Задача управления требованиями выходит за рамки компетенции LDRA Tool Suite, для этого существует отдельный класс программных продуктов (например, популярная в авиации система IBM Rational DOORS). Однако, поскольку для подтверждения соответствия нужно демонстрировать привязку кода и тестов к требованиям, степень покрытия требова-

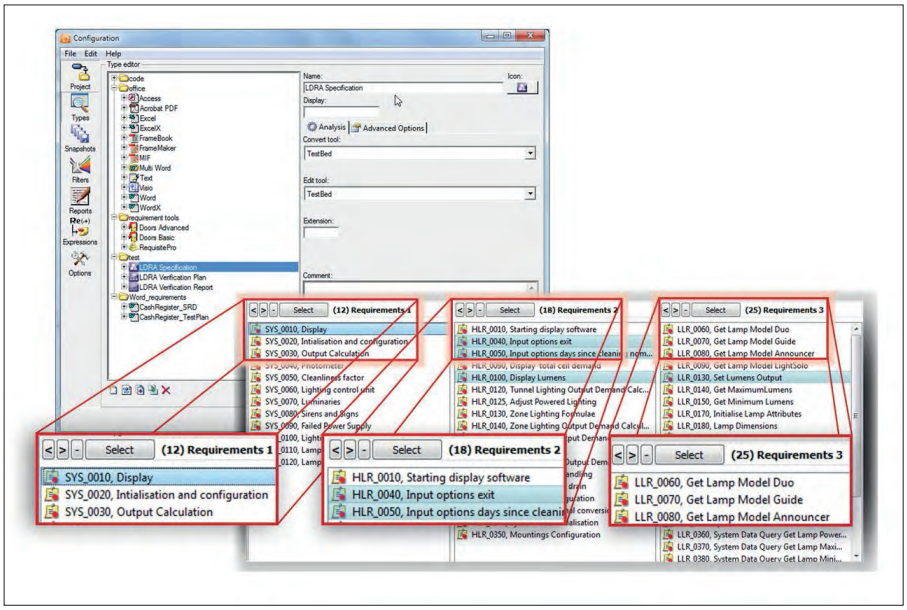



**Рис. 3. Трассировка требований в LDRA Tool Suite**

ний тестами и т.п., LDRA Tool Suite умеет импортировать требования из внешних источников (они, к слову, могут быть любыми, хоть документами Microsoft Office) и настраивать и поддерживать необходимые информационные связи между требованиями разного уровня и остальными проектными документами. Когда затем что-то в проекте меняется (требования, код, тестовые процедуры и т.п.), LDRA Tool Suite уведомляет пользователя, какие из связанных документов затрагиваются внесёнными изменениями и требуют обновления. Кроме поддержания целостности, эта функция играет ещё одну важную роль – она позволяет оценить стоимость изменений, так как наглядно демонстрирует, какой объём работы придётся реально выполнить, чтобы внести, с первого взгляда, незначительную точечную правку;

- средства **интеграции с системами управления конфигурацией** (контроля версий). Поскольку стандартами безопасности ПО предписывается необходимость контроля версий проектных документов, документы обычно хранятся в централизованном репозитории, организованном с помощью одной из автоматизированных систем контроля версий (version control), – CVS, SVN, Perforce, SourceSafe и т.п. Пакет LDRA Tool Suite поддерживает интеграцию с этими системами, что упрощает его подключение к существующей ИТ-инфраструктуре; в свою очередь, использование автоматизированной системы контроля версий позволяет быть уверенным в корректности входных данных и облегчает трассировку;
- средства **статического анализа**, в том числе подсчёта метрик, построения графов вызовов и подтверждения соответствия стандартам кодирования (рис. 4). В процессе статического анализа, кроме проведения необходимых проверок исходного текста и расчёта его количественных характеристик, формируется граф внутренней структуры ПО и взаимосвязи модулей, ко-

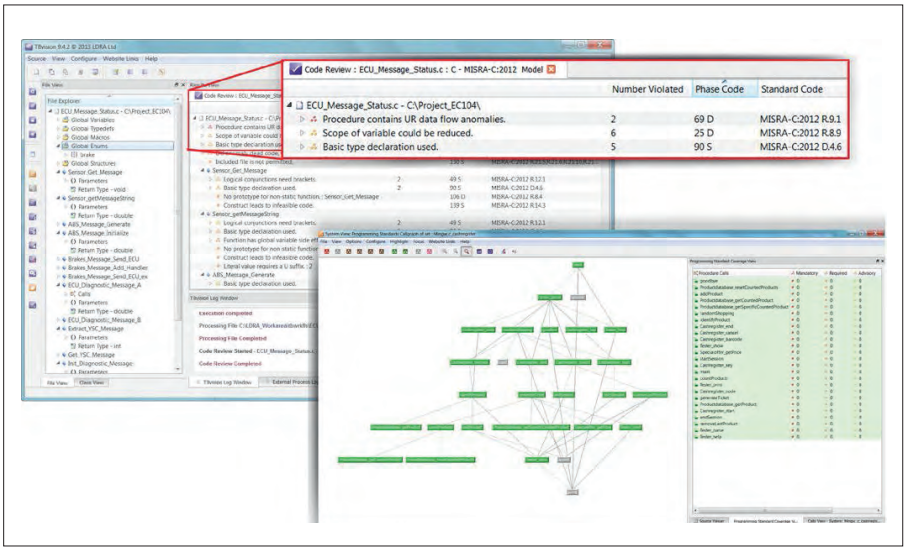



**Рис. 4. Статический анализ в LDRA Tool Suite**

ПРОИЗВОДСТВО ЭЛЕКТРОНИКИ ЛЮБОЙ СТЕПЕНИ СЛОЖНОСТИ

Доломант

ЗАО «НАУЧНО-ПРОИЗВОДСТВЕННАЯ ФИРМА «ДОЛОМАНТ»

КОНТРАКТНОЕ ПРОИЗВОДСТВО

Контрактная сборка электронного оборудования

- ОКР, технологические консультации
- Макеты, установочные партии
- Полное комплектование производства, поддержание складов
- Серийное плановое производство
- Гарантийный и постгарантийный сервис

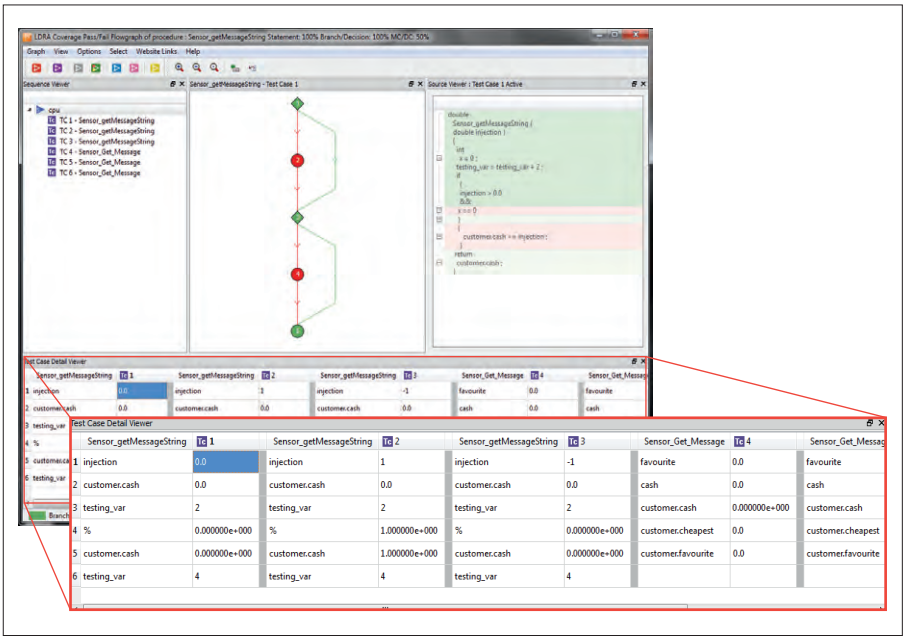
ЗАКАЗНЫЕ РАЗРАБОТКИ

Разработка электронного оборудования по ТЗ заказчика в кратчайшие сроки

- Модификация КД существующего изделия
- Разработка спецвычислителя на базе COM-модуля
- Конфигурирование модульного корпусированного изделия
- Сборка магистрально-модульной системы по спецификации заказчика
- Разработка изделия с нуля

ТЕЛ.: (495) 739-0775 / PRODUCT@DOLOMANT.RU / WWW.DOLOMANT.RU

Реклама



Test Case Detail Viewer

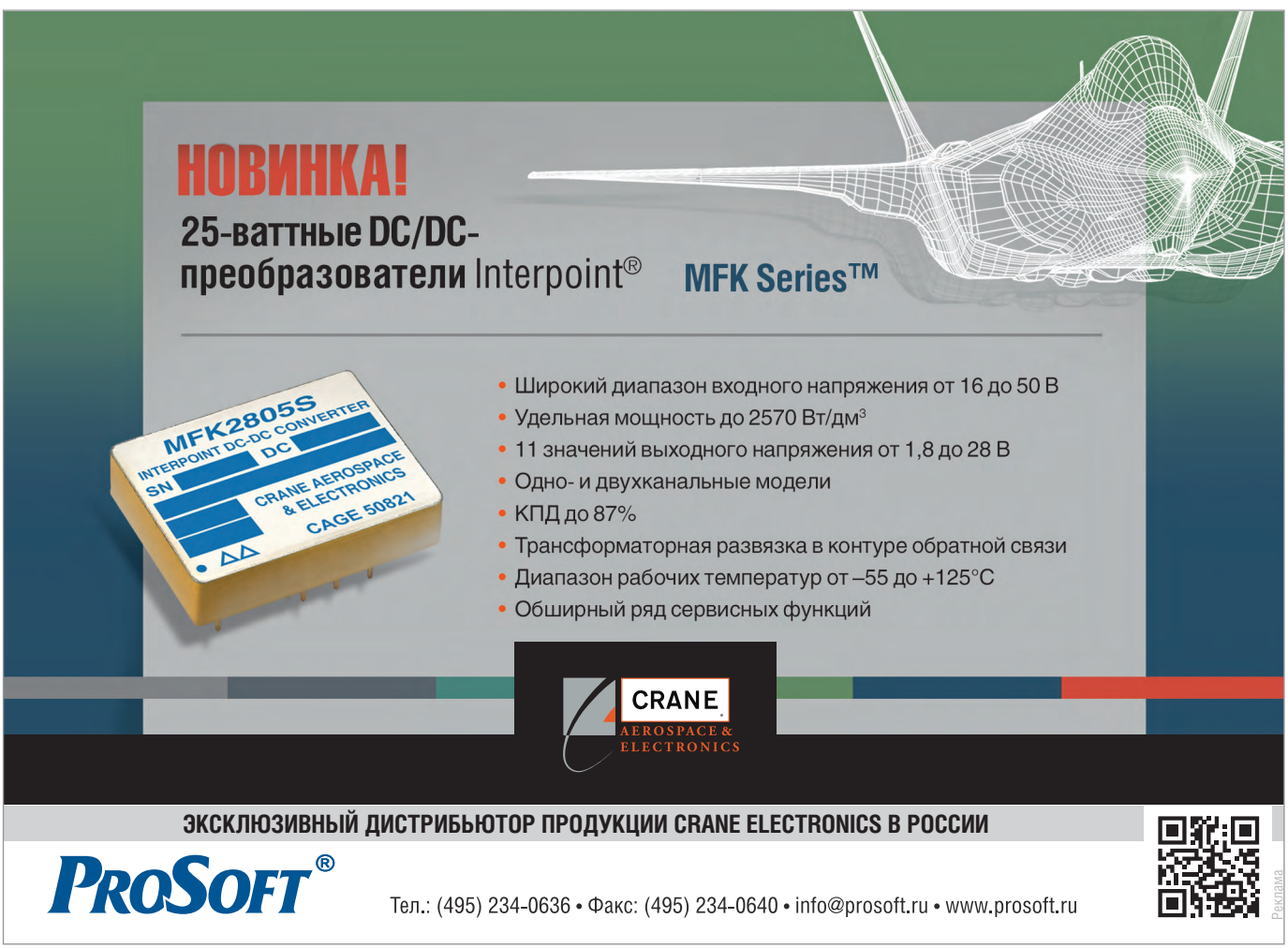
| | Sensor_getMessageString | Tc 1 | Sensor_getMessageString | Tc 2 | Sensor_getMessageString | Tc 3 | Sensor_Get_Message | Tc 4 | Sensor_Get_Messag |
|---|---|---|---|---|---|---|---|---|---|
| 1 | injection | 0.0 | injection | 1 | injection | -1 | favourite | 0.0 | favourite |
| 2 | customer.cash | 0.0 | customer.cash | 0.0 | customer.cash | 0.0 | cash | 0.0 | cash |
| 3 | testing_var | 2 | testing_var | 2 | testing_var | 2 | customer.cash | 0.000000e+000 | customer.cash |
| 4 | % | 0.000000e+000 | % | 1.000000e+000 | % | 0.000000e+000 | customer.cheapest | 0.0 | customer.cheapest |
| 5 | customer.cash | 0.000000e+000 | customer.cash | 1.000000e+000 | customer.cash | 0.000000e+000 | customer.favourite | 0.0 | customer.favourite |
| 6 | testing_var | 4 | testing_var | 4 | testing_var | 4 | | | |

**Рис. 5. Автоматизированное тестирование и анализ покрытия в LDRA Tool Suite**

торый затем используется как входная информация для тестирования и анализа покрытия;

- средства **автоматизированного тестирования**, включая автоматическую генерацию «обёртки» и массивов входных данных (в т.ч. граничных и случайных значений) для программных модулей, анализ структурного покрытия и тестирование на целевой системе (рис. 5); необходимые режимы и глубина тестирования определяются применимыми стандартами;
- средства **командной работы** с распределением ролей. Подтверждение соответствия включает в себя множество задач, в общем случае выполняемых разными людьми (а зачастую эти люди ещё и *обязаны* быть разными согласно нормативным требованиям). LDRA Tool Suite позволяет распределить задачи между соответствующими исполнителями и централизованно контролировать процесс;
- система **генерации отчётов** в настраиваемом формате. Выполнение предписанных выбранным стандартом процедур V&V – это ещё не всё: все полученные результаты нужно ещё представить в удобочитаемой форме для предъявления оценщику в составе сертификационной документации. Встроенный генератор отчётов LDRA Tool Suite позволяет экспортировать результаты в документы настраиваемого формата, а также автоматически генерировать сложные кросс-отчёты, например матрицу трассировки.

Часть инструментов пакета может интегрироваться непосредственно в интерфейс среды разработки, в частности, средства статического анализа из комплекта LDRA Tool Suite могут быть использованы как отдельно, так и в составе интегрированных сред Wind River Workbench (при разработке систем на базе ОС VxWorks или Wind River Linux)

или QNX Momentics (при разработке систем на базе ОС QNX Neutrino).

У этого подхода, правда, тоже есть одна тонкость: чтобы оценщик признал результаты V&V, полученные с использованием инструментального пакета, этот инструментальный пакет должен предварительно пройти так называемую *квалификацию* (qualification), чтобы подтвердить, что генерируемые им результаты корректны. Квалификация может производиться третьей стороной, поэтому важно, чтобы оценщик, работающий с сертификационной документацией, созданной с помощью инструментального пакета, признал его квалификацию. В общем случае оценщик может потребовать повторной независимой квалификации инструментального пакета (например, квалификация инструментальных средств, выполненная зарубежными оценщиками, в России может не признаваться), и чтобы сделать это возможным, производители инструментальных пакетов V&V предоставляют для своих продуктов *пакеты квалификационной документации*.

Пример комбинации коммерческих сертифицируемых ОС с коммерческим инструментарием V&V для снижения стоимости сертификации приведён на рис. 6.


**Рис. 6. Сокращение трудозатрат на подготовку к сертификации за счёт применения коммерческих компонентов и инструментария**

## Разделяй и властвуй: IMA, MILS и виртуализация

Вычислительная мощность и функциональные возможности современных процессоров позволяют им выполнять несколько приложений одновременно, и если с точки зрения сокращения массогабаритных характеристик и потребляемой мощности это однозначно плюс, то с точки зрения стоимости сертификации, на первый взгляд, всё совсем наоборот. Дело в том, что, как уже упоминалось, объём работ по подтверждению соответствия сильно зависит от уровня безопасности (как функциональной, так и информационной), по которому производится сертификация, а критичного кода в системе часто значительно меньше, чем некритичного. Объединение на одном процессоре нескольких приложений разной степени критичности требует сертификации всех этих приложений по максимальному среди них уровню безопасности, а значит, суммарные трудозатраты резко возрастают. Или нет?

Хорошим выходом из положения было бы разделить приложения по соответствующим уровням безопасности и сертифицировать их по отдельности, каждое по своему уровню, но для этого необходимо продемонстрировать, что

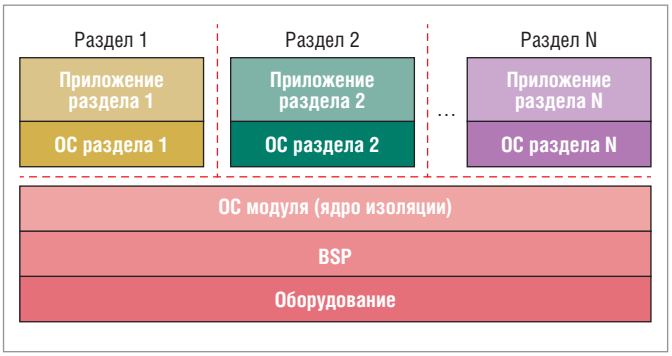



**Рис. 7. ОС с архитектурой IMA (пример на основе VxWorks 653)**

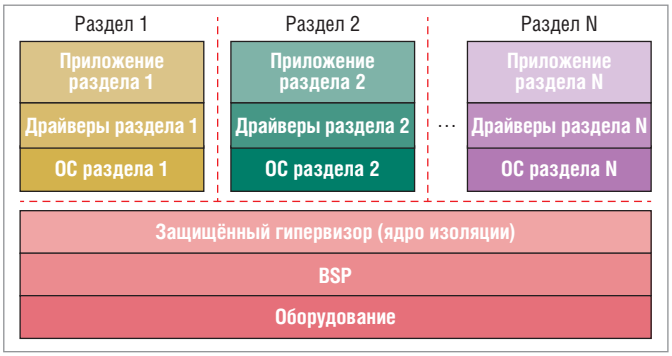



**Рис. 8. ОС с архитектурой MILS (пример на основе VxWorks MILS)**

разделение реализовано достаточно полно, и все приложения вроде и выполняются на одном и том же процессоре, но взаимодействуют только «по уставу» и нарушить работу друг друга никак не могут. Реализовать и продемонстрировать оценщику такое разделение можно только при условии наличия соответствующей поддержки со стороны процессора и ОС. В сфере функциональной безопасности такой подход впервые появился в авиаприборостроении и получил название интегрированной модульной авионики (Integrated Modular Avionics – IMA, рис. 7), в сфере информационной безопасности аналогичный подход называют множественными независимыми уровнями безопасности (Multiple Independent Levels of Security – MILS). Правда, поскольку в термины IMA и MILS заложен не только сам подход, но и определённые архитектурные детали реализации, в случае когда нужно абстрагироваться от деталей, взамен употребляют также термины «системы смешанной безопасности» (mixed safety systems), «системы смешанной защищённости» (mixed security systems) или упомянутый более общий термин «системы смешанной критичности» (mixed criticality systems), включающий в себя предыдущие два.

Поскольку речь идёт о сертификации ПО, аппаратную сторону вопроса оставим за кадром; с точки же зрения ОС, архитектуры IMA и MILS, как и можно было предположить, выглядят очень схоже, в качестве примера рассмотрим соответственно реализацию ОС VxWorks 653 и VxWorks MILS уже упомянутой компании Wind River. Обе эти ОС являются двухуровневыми и фактически представляют собой комбинацию сертифицируемого гипервизора и сертифицируемых гостевых ОС. Гипервизор (играющий роль так называемого *ядра изоляции* – separation kernel) занимается формированием и обслуживанием разделов безопасности, в то время как гостевые ОС разделов отвечают за выполнение приложений в этих разделах.

Разделы получают процессорное время по определённому графику, то есть тоже подвержены *планированию*, иными словами, планировщики ОС VxWorks 653 и VxWorks MILS также являются двухуровневыми: с одной стороны, гипервизор планирует разделы, с другой, гостевые ОС разделов планируют потоки своих приложений, когда их раздел активен.

Всё это обеспечивает *пространственную и временну́ю изоляцию разделов*, предписываемую концепциями IMA и MILS. Подробный обзор архитектур IMA и MILS приведён в 1-й части статьи [6].

Обратите внимание, что приведённые на рис. 7 и 8 примеры IMA- и MILS-архитектур отличаются расположением драйверов устройств: в случае MILS-архитектуры драйверы виртуализированы и выполняются в соответствующих разделах в пространстве пользователя, что повышает защищённость. В IMA-архитектурах, напротив, разделы полностью абстрагированы от оборудования, и драйверы выполняются в пространстве ядра в составе ОС модуля. Вследствие этого, несмотря на концептуальное сходство, ОС с IMA-архитектурой сложнее привести к соответствию требованиям информационной безопасности – подробнее об этом сказано далее.

В рамках данной концепции гипервизор подлежит сертификации по самому высокому из требуемых уровней безопасности, зато гостевые ОС и приложения разделов могут сертифицироваться по уровню *не выше требуемого*. В результате удаётся, объединив на одном процессоре несколько приложений с различными требованиями к безопасности, одновременно сэкономить на массе, габаритах и энергопотреблении, и, как минимум, не увеличить стоимость сертификации (а при грамотном распределении кода по разделам безопасности и дополнительно сократить её).

Забегая немного вперёд, дополнительно надо отметить, что в системах смешанной безопасности, не относящихся к IMA и поэтому не подпадающих под требования спецификации ARINC 653, реализация *временно́й* изоляции разделов обязательной не является – достаточно обеспечить *пространственную* изоляцию. Это упрощает создание виртуализированных систем смешанной безопасности на базе многоядерных процессоров, но об этом будет сказано позже.

В четвёртой, заключительной части статьи рассматриваются примеры решений на основе коммерческих (COTS) компонентов и возможные перспективы развития технологий безопасности ПО. ●

## Литература

1. DO-178B&DO-254 White Papers [Электронный ресурс] // Режим доступа: http://www.highrely.com/whitepapers.php.
2. DO-178 Industry Group for Engineers [Электронный ресурс] // Режим доступа: http://www.do178site.com/
3. Christian Hagen, Jeff Sorenson. Delivering military software affordably // Defense AT&L. – 2013. – March–April.
4. LDRA Certification Services [Электронный ресурс] // Режим доступа: http://www.ldra.com/en/services-support/certification-services.
5. U.S. Government Protection Profile for Separation Kernels in Environments Requiring High Robustness. – USA: Information Assurance Directorate, 2007.
6. Паркинсон Пол. Многоядерные вычислительные среды и безопасность ПО. Часть 1 // Современная электроника. – 2013. – № 8.

**Автор – сотрудник фирмы ПРОСОФТ**
**Телефон: (495) 234-0636**
**E-mail: info@prosoft.ru**

РОССИЙСКИЙ БРЕНД
ИМПОРТОЗАМЕЩЕНИЕ

PROVS®

# Обнаружить. Распознать. Предупредить.


## ПРОФЕССИОНАЛЬНЫЕ СИСТЕМЫ ВИДЕОНАБЛЮДЕНИЯ

- ПРОМЫШЛЕННЫЕ ОБЪЕКТЫ
- БИЗНЕС-ЦЕНТРЫ
- ТРАНСПОРТ
- СИСТЕМЫ «БЕЗОПАСНЫЙ ГОРОД», «УМНЫЙ ДОМ»

- Комплексные программно-аппаратные решения любой сложности
- IP-видеокамеры любых типов и исполнений
- Видеокамеры HD-SDI
- Видеорегистраторы IP, HD-SDI и гибридные
- Аналоговые видеокамеры и регистраторы
- Видеорегистраторы специализированные
- Видеорегистраторы на базе промышленных компьютеров AdvantiX, Advantech, MEN
- Радиолокационные системы охраны
- Периферийные устройства и аксессуары, коммутаторы
- Программное обеспечение

PROSOFT®

Тел.: (495) 234-0636 доб. (1574) • provs@prosoft.ru

Реклама