Программный комплекс проектирования и анализа программно-аппаратных систем на основе архитектурных моделей[1]

Петренко А.К.[1, 2 , 4]

д.ф.-м.н., профессор, petrenko@ispras.ru

Хорошилов А.В. [1, 2, 3, 4]

к.ф.-м.н., khoroshilov@ispras.ru

[1] Институт системного программирования им. В.П.Иванникова РАН,
109004, Россия, г. Москва, ул. А. Солженицына, д. 25.
[2] Московский государственный университет имени М.В.Ломоносова,
119991, Российская Федерация, Москва, Ленинские горы, д. 1.
[3] Московский физико-технический институт,
141700, Московская область, г. Долгопрудный, Институтский пер., 9.
[4] Научно-исследовательский университет Высшая школа экономики,
101000, г. Москва, ул. Мясницкая, д. 20.

Abstract. The paper examines the functional requirements and the basic architecture of the tools supporting model-oriented techniques for design of safety critical software/hardware systems. The best practices of development such systems are described in international standards like ARP-4754, ARP-4761, and DO-178. The proposed approach is built around architecture models expressed in AADL language and its extensions.

Аннотация. В статье рассматриваются вопросы функциональных требований к инструментальным средствам поддержки моделирования систем и вопрос выбора базовой архитектуры набора инструментов для поддержки моделе-ориентированных техник разработки программно-аппаратных систем ответственного назначения. Совокупность задач, которые должны решаться при помощи этих методов и инструментов, уже зафиксированы в ряде международных стандартов, таких как ARP-4754, ARP-4761 и DO-178. В качестве языка описания архитектурных моделей систем используется язык AADL.

Ключевые слова: модели программ, встроенные системы, системы ответственного назначения, AADL, стандартизация, сертификация.

---

1 Поддержано грантом РФФИ, Договор № 17-01-00504\17.

1 Введение

Создание программно-аппаратных систем ответственного назначения, примерами которых являются, например, системы управления летательными аппаратами или энергетическими установками, требует развитых методов и инструментальных средств, поддерживающих все основные фазы процесса проектирования, разработки и эксплуатации таких систем. Общие подходы и отдельные средства поддержки разработки и анализа систем уже достигли высокого уровня зрелости. Совокупность задач, которые должны решаться при помощи этих методов и инструментов, уже зафиксированы в ряде международных стандартов, таких как ARP-4754 [1], ARP-4761 [2] и DO-178 [3].

По мере роста сложности ответственности разрабатываемых систем всё яснее осознаётся необходимость применения средств всесторонней автоматизации их разработки и анализа с использованием их моделей, в том числе формальных моделей и формальных спецификаций интерфейсов.

Однако практика использования таких методов и инструментов существенно отстает от возможностей, которые потенциально открываются перед пользователями этих методов, особенно это заметно в практике отечественных компаний, ведущих разработки в области создания программно-аппаратных систем ответственного назначения. Одной из причин, сдерживающих освоение новых техник разработки, является серьезное отставание средств поддержки проектирования, моделирования и анализа программно-аппаратных систем и их моделей, особенно, если сравнивать их возможности с возможностями современных средств поддержки программирования систем обычного назначения (без требований к надежности и безопасности повышенного уровня).

В рамках данной статьи авторы останавливаются на вопросах выбора правильного позиционирования методов и средств проектирования и на вопросах выбора базовой архитектуры инструментального комплекса.

В целом рассматриваемый подход следует отнести к широкому набору методов и средств разработки программно-аппаратных систем на основе моделей. В литературе известно несколько методологий/лозунгов, которые, в принципе, достаточно близки по духу, это MDA (Model Driven Architecture), MDD (Model Driven Design/Development), MDE (Model Driven Engineering) и другие. Эти методологии активно развивались и продвигались рядом крупных игроков на ИТ-рынке, начиная с 90-х годов XX века. Однако сейчас, по прошествии более 20 лет, можно констатировать, что реального широкого использования они не нашли.

Объяснением "холодного" отношения программной индустрии к моделеориентированным техникам разработки может служить то, что они не дают заметного и быстрого экономического выигрыша, и при этом поднимают ряд непростых проблем, связанных со сложностью внедрения, масштабируемостью, с поддержкой процесса разработки. Вместе с тем, есть области производства программ и программно-аппаратных комплексов, где востребованность моделеориентированных техник есть – это разработка комплексов управления ответ-

ственными системами, где требуется сертификация, регламенты которой могут быть выполнены с использованием моделе-ориентированных техник.

Например, в области гражданской авионики базовыми стандартами в этой области являются:

- ARP4754 “Условия сертификации для высоко интегрированных или сложных авиационных систем”, регламентирующий процессы проектирования и разработки комплексов бортового оборудования (КБО) и его программно-аппаратных подсистем;
- ARP4761 “Правила и методы выполнения процесса оценки безопасности бортовых гражданских систем и оборудования”, регламентирующий цели и задачи проведения анализа КБО с точки зрения безопасности;
- DO-178 и его последняя версия DO-178C “Требования к программному обеспечению бортовой аппаратуры и систем при сертификации авиационной техники”, регламентирующий процессы разработки как функционального, так и системного программного обеспечения КБО.

Производными по отношению к DO-178 являются стандарты для других отраслей техники, например железные дороги (IEC 62279), ядерная энергетика (IEC 61513), автомобилестроение (ISO 26262), а также региональные стандарты, например европейские (ED-12B/C) или российские (КТ-178B/C).

Заметим, что требования стандартов адресованы в первую очередь не к программному или программно-аппаратному изделию, к его качеству, а к процессу жизненного цикла создания и эксплуатации изделия. При этом важно понимать, что при такой постановке вопроса предъявлять свидетельства (evidences), подтверждающие квалификационные требования исполнителя нужно, не в конце проекта, а начале и в ходе проекта, что непривычно для большинства компаний-разработчиков ПО. Как следствие, у компаний, имеющих опыт в разработке ответственных систем заданного вида, возникает конкурентное преимущество – они еще на старте проекта или даже до старта в состоянии предъявить свидетельства, объективно подтверждающие их готовность и состоятельность в роли исполнителя. Хотя разработка и поддержка документации, определяемой упомянутыми выше стандартами, требует существенных затрат, развертывание соответствующих процессов вносит существенный вклад в качество процессов, тем самым снижает риски, которые могут реализоваться в ходе работ и, в конечном счете, растет надежность финального продукта. В частности, наличие подробной, четко структурированной документации по требованиям и по проекту ПО упрощает введение в проект новых разработчиков и упрощает общение специалистов разных областей, так или иначе вовлеченных в проект.

У российской промышленности опыта освоения процессов сертификации, которые опираются на перечисленные стандарты пока мало. По этой причине и случаев реального внедрения моделе-ориентированных методов и поддерживающих их инструментов буквально единицы. Из зарубежных инструментов, которые поддерживают MDD, в России, вероятно, самым известным является SCADE компании ANSYS. В дополнение к известному на рынке инструменту разработки управляющего программного обеспечения компания эта компания

стартовала разработку моделе-ориентированного комплекса автоматизации проектирования и анализа КБО SCADE SYSTEM. В ИСП РАН при поддержке ГосНИИАС уже несколько лет идет разработка комплекса средств для создания архитектурных моделей на основе языка AADL [4, 5], который получил название MASIW (Modular Avionics System Integrator Workplace) [6-9]. Данная статья базируется на опыте разработки этого инструмента.

Использование архитектурных моделей при разработке программно-аппаратных комплексов позволяет решать различные задачи, но в основном это задачи поддержки работы архитекторов и интеграторов авионики. Инструмент MASIW предназначен именно для этих целей, он является рабочим местом архитектора и интегратора КБО. В число задач этих специалистов входит уточнение и согласование требований с разработчиками программного и аппаратного обеспечения и собственно проектирование программно-аппаратной платформы исходя из потребностей функциональных приложений в аппаратных ресурсах, в том числе:

- распределение функциональных приложений по вычислительными модулям с учетом потребностей приложений (количество процессорного времени, распределение процессорного времени между строго периодическими приложениями, объем памяти ОЗУ/ПЗУ, пропускная способность сетевых интерфейсов и т. п.);
- определение состава сетевых компонентов (топологии сети) с учетом требований надежности, согласованности интерфейсов, времени доставки сообщений от отправителя к получателю и т. п.
- проверка разрабатываемого комплекса бортового оборудования (КБО) на соответствие требованиям, изложенных в проектной документации к самолету, КБО и его отдельным компонентам;
  - анализ структуры программно-аппаратного комплекса – достаточности аппаратных ресурсов, согласованности интерфейсов и т. п.;
  - анализ характеристик передачи данных в сети AFDX – времени доставки сообщений от отправителя к получателю, глубины очередей передающих портов и т. п.;
  - симуляцию модели программно-аппаратного комплекса с генерацией пользовательских отчётов по результатам работы симулятора, в т.ч. совместную симуляцию работы прикладных разделов под управлением ОС РВ в эмуляторе Qemu и универсального симулятора AADL моделей;
- построение дерева неисправностей и численный анализ дерева неисправностей для определения вероятности отказного события верхнего уровня;
- анализ видов и последствий отказов на основе архитектурной модели комплекса бортового оборудования, включая построение таблицы видов и последствий отказов;
- подготовка конфигурационных таблиц для компонентов программно-аппаратной платформы.

Создание, редактирование и управление моделями, а также конфигурационных данных реализованы с использованием широко распространенных расширений среды Eclipse, такими как Eclipse Modeling Framework, Graphical Editing Framework, Eclipse Team Providing, SVN Team Provider, GIT Team Provider.

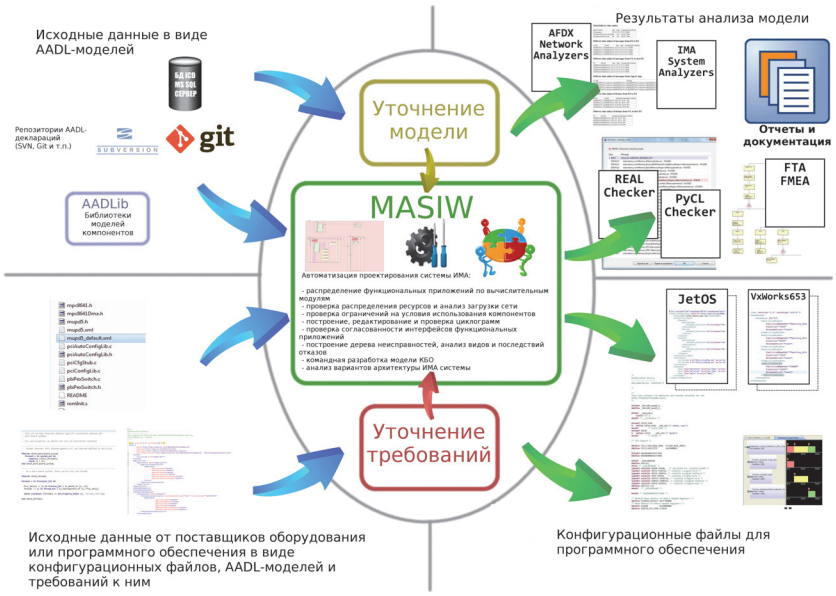

Рис. 1. Состав инструментального комплекса MASIW

Набор инструментов MASIW построен на основе модульной архитектуры и в связи с этим функционал MASIW может быть расширен – сторонние разработчики путём создания собственных модулей могут расширить функционал инструмента в соответствии с потребностями.

Подход, использованный при создании системы MASIW основывается на использовании модели разрабатываемого КБО как основы, вокруг которой строится функциональность разнообразных дополнений, отвечающих за решение задач редактирования, визуализации, синтеза, анализа, трансформации и т. д. Эти дополнения либо вносят изменения в модель КБО, развитие которой при этом контролируется средствами управления версиями и конфигурациями, либо используют модель как источник данных и генерируют внешние артефакты такие как отчеты о результатах анализов или конфигурационные таблицы пригодные для загрузки в аппаратные компоненты КБО или их эмуляторы.

В целом архитектуру MASIW можно рассматривать как комплекс, интегрирующий разнообразные артефакты и структуры данных. В первую очередь в перечень этих структур входят основные:

- тестовое и графическое (внутреннее) представление моделей программно-аппаратных систем, включая статические атрибуты компонентов этих моделей;
- текстовое (гипертекстовое) представление требований (с возможностью построения иерархических структур требований);

и дополнительные:

- текстовое представление статических артефакторов верификации и тестирования (спецификации данных, операций, инвариантов, утверждений (assertions)), задач тестирования/верификации, тестовых сценариев и планов тестирования;
- текстовое и табличное представление динамических артефактов: трасс тестирования/верификации, отчетов о результатах тестирования/верификации, отчетов о покрытии, отчетов о статусе выполнения задач, сценариев/планов тестирования/верификации.

Близким примером интеграции разнородных инструментов является архитектура конфигурируемых анализаторов программ CPAchecker [10], которая позволяет соединять в единый комплекс инструменты верификации программного кода от различных поставщиков и использовать их совместно или даже в параллельно-конкурентном стиле, с тем, чтобы остановиться на лучшем из полученных результатов или для того, чтобы построить решение задачи из комбинации различных решений, которые получили различные программы-верификаторы. Этот опыт весьма поучителен, хотя пока он не был опробован для интеграции существенно разных по своим задачам компонентов инструментальных систем.

2 Состояние работ в предметной области

Тема создания базовой архитектуры комплекса инструментов, включающего поддержку работы с моделями, обсуждается достаточно широко как в научной печати, так и на научных конференциях. Основными площадками таких обсуждений являются международные конференции MODELSWARD, MODELS, ECMFA; научный журнал - SoSyM - Journal on Software & System Modeling.

В качестве основы для интеграции инструментов используются либо платформы типа Eclipse, либо XML технологии. Общим недостатком имеющихся базовых архитектур является сложность интеграции разнородных инструментов и структур данных, а также невозможность поддержки моделей и трасс моделирования/мониторинга большого размера.

Среди исследовательских и коммерческих инструментов схожего назначения необходимо выделить средства проектирования и интеграции модульной авионики крупных авиастроительных корпораций, таких как Airbus и Boeing, а также средства моделирования программно-аппаратных систем общего назначения IBM Rational Rhapsody, Sparx Systems Enterprise Architect, SCADE System, TOPCASED, OSATE [11], Thales Capella. Актуальную информацию об AADL-

инструментах можно найти на страницах Open AADL [12]. Из российских научно-технических центров, которые активно занимаются данной тематикой, в первую очередь нужно назвать ГосНИИАС. Более подробный обзор можно найти в [9].

3 Заключение

В статье рассмотрены основные задачи, которые должны решаться при разработке ответственных программно-аппаратных систем с использованием архитектурных моделей. Описываются проектные решения, положенные в основу комплекса средств поддержки архитектурных моделей MASIW, нацеленные на простоту развития комплекса и интеграции его с новыми инструментами конструирования и анализа моделей и реализация КБО.

Литература

1. ARP4754 “Условия сертификации для высоко интегрированных или сложных авиационных систем”.
2. ARP4761 “Правила и методы выполнения процесса оценки безопасности бортовых гражданских систем и оборудования”.
3. RTCA/DO-178C "Software Considerations in Airborne Systems and Equipment Certification", RTCA/EUROCAE, 2012.
4. SAE International. “Architecture Analysis & Design Language (AADL)”, SAE International Standards document AS5506B, Nov 2004, Revised Mar 2012.
5. Julien Delange. AADL tutorial at MODELS'15 // http://www.openaadl.org/post/2015/09/28/models/ (проверено 20.10.2017).
6. Alexey Khoroshilov, Eugene Kornykhin. PyCL – Python-based AADL Constraint Language. Proceedings of the Second International Workshop on Architecture Centric Virtual Integration - ACVI 2015, Madrid, Spain, June 26, 2015 http://www.aadl.info/aadl/acvi/acvi2015 /papers/ACVI15_submission_5.pdf.
7. D.Buzdalov, A.Khoroshilov. A discrete-event simulator for early validation of avionics systems. Proceedings of the First International Workshop on Architecture Centric Virtual Integration - ACVI 2014, Valencia, Spain, September 29, 2014. CEUR WS Vol-1233, pp.28-38, http://ceur-ws.org/Vol-1233/acvi14_submission_3.pdf.
8. Khoroshilov, A., Albitskiy, D., Koverninskiy, I., Olshanskiy, M., Petrenko, A., Ugnenko, A., «AADL-Based Toolset for IMA System Design and Integration,» SAE Int. J. Aerosp. 5(2):2012, doi:10.4271/ 2012-01-2146.
9. Д.В. Буздалов, С.В. Зеленов, Е.В. Корныхин, А.К. Петренко, А.В. Страх, А.А. Угненко, А.В. Хорошилов. Инструментальные средства проектирования систем интегрированной модульной авионики // Труды Института системного программирования РАН, том 26, вып. 1, 2014, стр. 201-230.
10. И.С. Захаров, М.У. Мандрыкин, В.С. Мутилин, Е.М. Новиков, А.К. Петренко, А.В. Хорошилов. Конфигурируемая система статической верификации модулей ядра операционных систем. Программирование, №1, 2015, с. 44-67.
11. https://wiki.sei.cmu.edu/aadl/index.php/Osate_2 (проверено 20.10.2017).
12. http://www.openaadl.org/ (проверено 20.10.2017).