Процесс Разработки Программно-Аппаратных Систем на Основе Визуального Моделирования с Использованием SysML/UML

Dmitry Ryzhov
SWD Software Ltd.
email: d.ryzhov@swd.ru

Denis Ivanov
Ай Ти Консалтинг.
email: denis.ivanov@it-konsulting.spb.ru

Abstract (English)

More and more, systems engineers are turning to the System Modeling Language (SysML) to specify and design their embedded systems. This has many advantages, including verifiability and ease of passing off information to other engineering disciplines, particularly software and hardware. This paper describes a SysML-based process that systems engineers can use to capture requirements and specify architecture. The process is function-driven and is based heavily on the identification and elaboration of operational contracts, a message-based interface communication concept. The process described is developed by Telelogic and actively being used in real projects for embedded systems development based on Model-Driven System Engineering (MDSE) approach.

***Keywords**: System Modeling Language (SysML), embedded systems, system&software integrated development process, functional decomposition, model verification and validation*

Дмитрий Рыжов
SWD Software Ltd.
email: d.ryzhov@swd.ru

Денис Иванов
Ай Ти Консалтинг.
email: denis.ivanov@it-konsulting.spb.ru

Abstract (Russian)

Все большее число системных инженеров начинают использовать язык моделирования SysML для специфицирования и проектирования программно-аппаратных систем. Это дает им много преимуществ, в том числе возможности верификации и валидации моделей систем и облегчение передачи информации представителям других инженерных дисциплин, в частности разработчикам программного и аппаратного обеспечения. Данная статья содержит описание процесса разработки систем, основанного на SysML, который системные инженеры могут использовать для спецификации требований и определения архитектуры систем. В основе процесса лежит функциональная декомпозиция, основанная на определении и дальнейшем уточнении операционных контрактов, а также концепция взаимодействия на основе обмена сообщениями. Описываемый процесс разработан компанией Telelogic и активно используется во множестве реальных проектов по разработке программно-аппаратных систем на основе визуального моделирования (MDSE).

***Keywords**: язык моделирования SysML, программно-аппаратные системы, интегрированный процесс разработки систем и ПО, функциональная декомпозиция, верификация и валидация модели*

1. Введение

Многие годы разработчики программного обеспечения с успехом применяли язык UML для разработки программного обеспечения на основе визуального моделирования. Было предпринято несколько попыток применить UML и лежаший в его основе объектно-ориентированный подход к разработке систем в целом с целью унификации всего процесса разработки. Однако многие системные инженеры продолжают использовать классические методы структурного анализа и соответствующие артефакты для определения требований к системам и дальнейшего ее проектирования. Одна из возможных причин такого положения дел состоит в том, что разработка систем определяется в большей степени функциональными требованиями. Формулировки в терминах функциональности системы при ее проектировании все еще рассматриваются как наиболее естественными большинством специалистов инженерных дисциплин, вовлекаемых на ранних этапах разработки систем (схемотехников, конструкторов, маркетологов и, конечно, заказчиков). Учитывая вышесказанное, единственный путь унификации всего процесса разработки заключается в расширении UML для получения возможности разработки систем на основе функциональной декомпозиции и определении процесса, позволяющего осуществлять плавную передачу результирующих артефактов разработчикам программного обеспечения, использующим UML. Для достижения этой цели OMG (Object Management Group) сформировала консорциум для разработки языка SysML (Systems Modeling Language).

На Рис. 1 изображены диаграммы языка SysML. Язык SysML является диалектом языка UML. Часть диаграмм языка UML не вошли в язык SysML, но появились две новых. Другие диаграммы UML были дополнены и частично изменены. Базовым элементом модели стал блок, а не класс. Диаграмма классов и диаграмма структуры UML получили новые названия.

В статье описывается процесс, основанный на языке моделирования SysML, который системные инженеры могут использовать для спецификации требований и определения архитектуры систем. Рассматриваемый процесс основан на возможности среды разработки Telelogic Rhapsody по исполнению моделей, как средстве для верификации и валидации требований к системе.

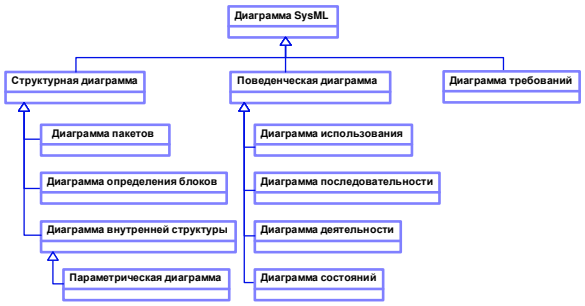

Рис. 1 Диаграммы языка SysML

2. Обзор процесса

На Рис. 2 с использованием классической “V” диаграммы показан интегрированный процесс, совмещающий в себе разработку систем и программного обеспечения. Левая наклонная часть буквы “V” отображает нисходящий процесс проектирования, в то время как правая наклонная часть отображает восходящие этапы интеграции, начиная с элементарных тестов и заканчивая приемкой конечной системы. Используя нотацию состояний, итеративная природа процесса отображается в виде “высокоуровневого перехода”, возникающего при появлении изменений. Стадия разработки систем характеризуется последовательным нисходящим потоком работ, включающим анализ требований, анализ системы и проектирование ее архитектуры. Поток работ для стадии разработки программного обеспечения характеризуются итеративными инкрементными циклами, содержащими этапы анализа, проектирования, реализации, а также этапы интеграции и тестирования различного уровня.

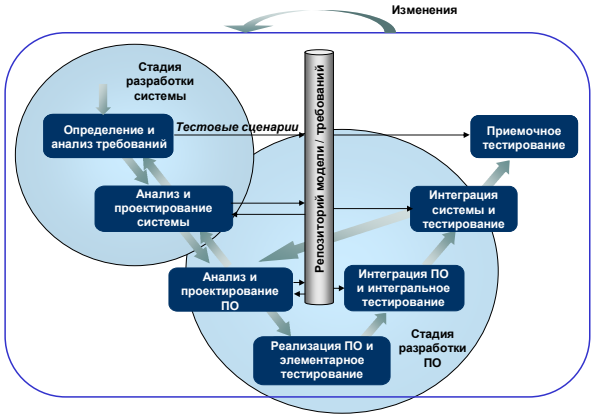

Рис. 2 Интегрированный процесс разработки систем и программного обеспечения

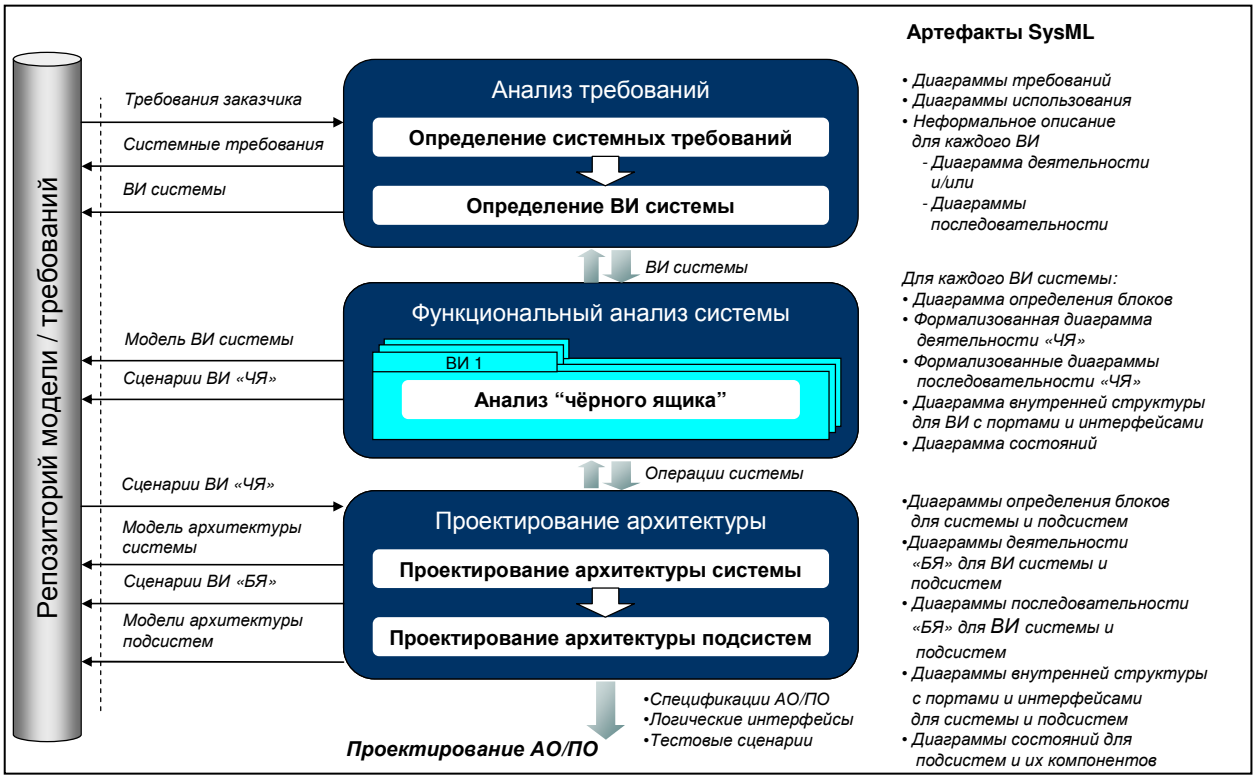

Рис. 3 Процесс разработки систем, основанный на SysML

На Рис. 2 также отображен факт создания и повторного использования на всех этапах проектирования тестовых сценариев, определяемых на основе исходных требований. Эти сценарии используются на этапах интеграции и тестирования, а также для регрессионного тестирования при внесении изменений в систему.

Перечислим ключевые цели процесса разработки систем:

• Определение требуемой функциональности системы и источников происхождения требований

• Определение состояний и режимов работы системы

• Определение подсистем и их компонентов и привязка к ним требований и необходимой функциональности

Для процесса моделирования эти цели подразумевают движение сверху вниз, начиная с абстракций самого высокого уровня. На Рис. 3 приведена схема процесса разработки систем, основанного на SysML. Для каждой фазы процесса показаны основные этапы, а также входные данные и создаваемые артефакты. В следующих разделах приводится их описание, а также определение последовательности шагов для каждого из этапов.

3. Фаза анализа требований

3.1. Этап определения требований

Фаза анализа требований начинается с анализа имеющейся информации. На основе требований заказчика создаются системные требования, которые определяют, что система должна делать (функциональные требования) и как хорошо она должна это делать (требования к качеству).

3.2. Этап определения вариантов использования

После того как требования к системе становятся понятны они группируются по вариантам использования. Один вариант использования описывает конкретный операционный аспект системы (поток операций). Он специфицирует поведение, которое видно пользователю, и определяет поток сообщений между пользователем и системой. При этом внутренняя структура системы не специфицируется никоим образом (представление "черного ящика"). Варианты использования могут быть структурированы иерархически. Для

отображения множества вариантов использования применяется *диаграмма использования*.

Полученные варианты использования связываются с требованиями к системе, после чего производится проверка полноты покрытия требований элементами модели.

4. Фаза функционального анализа

На фазе функционального анализа для каждого определенного варианта использования создается отдельная модель системы “черного ящика” и относящиеся к ним требования верифицируются и валидируются путем исполнения модели. При этом используется подход, основанный на обмене сообщениями:

- Структура системы описывается с помощью структурных диаграмм SysML, в качестве базовых элементов используются блоки, для которых определяются порты и предоставляемые / требуемые интерфейсы.
- Взаимодействие между блоками осуществляется путем обмена сообщениями (запроса сервисов).
- Функциональность блоков определяется с использованием операционных контрактов (сервисов), например, *операция1(),.., операция4()*
- Функциональная декомпозиция осуществляется путём декомпозиции операционных контрактов.

На Рис. 4 показано как связаны между собой артефакты SysML, создаваемые для каждого варианта использования в процессе функционального анализа.

Анализ варианта использования "черного ящика" начинается с определения окружения для варианта использования. Для этого используется такой артефакт SysML как *структурная диаграмма*. Элементами этой диаграммы являются блоки, моделирующие действующих лиц и саму систему.

На следующем шаге анализа варианта использования определяются сценарии "черного ящика". Один сценарий описывает определенную последовательность взаимодействия в рамках варианта использования. Он детализирует поток сообщений между действующими лицами и системой в рассматриваемом варианте использования, а также результирующее поведение (операционные контракты) на сторонах получателей сообщений. С использованием SysML сценарии определяются графически на *диаграммах последовательности*. Линии жизни на диаграмме последовательности "черного ящика" относятся к действующим лицам и системе.

Как только набор существенных сценариев определен, полученные сценарии объединяются в единое описание последовательности действий для варианта использования. Для этой цели используется такой артефакт SysML как *диаграмма деятельности*. Каждое действие на диаграмме деятельности соответствует операционному контракту на диаграмме последовательности. Диаграмма деятельности "черного ящика" играет важную роль на этапе проектирования архитектуры системы.

Основываясь на информации, определенной на диаграммах последовательности и диаграмме деятельности "черного ящика", для блоков структурной диаграммы определяются порты и связанные с ними интерфейсы.

Следующим шагом в анализе варианта использования "черного ящика" является описание поведения системы на основе состояний. Для этого используется такой артефакт SysML как *диаграмма состояний*. Диаграмма состояний отображает состояния и режимы работы системы, а также логику их изменения при возникновении внешних воздействий. Создание диаграммы состояний для варианта использования производится на основе определенных на предыдущих шагах сценариев.

На следующем шаге производится верификация и валидация модели варианта использования и связанных с ним требований. Верификация и валидация осуществляются посредством исполнения модели с использованием определенных сценариев "черного ящика" как основы для генерации входных воздействий для исполняемой модели. Следуя определенным выше ключевым целям процесса разработки систем, необходимо отметить, что основной фокус при этом направлен больше на анализ получаемых последовательностей вызовов, чем на реализуемую на их основе функциональность.

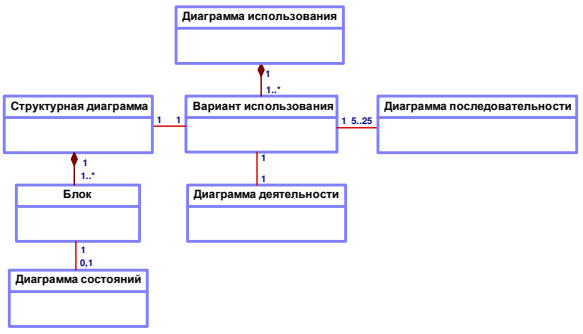

Рис. 4 Артефакты SysML, создаваемые в процессе функционального анализа

После того как созданы модели для всех вариантов использования определенные в них операционные контракты для системы объединяются в один системный блок.

В конце фазы функционального анализа, системный блок содержит верифицированные и валидированные операционные контракты, соответствующие функциональным требованиям к системе.

5. Фаза проектирования архитектуры

5.1. Этап проектирования архитектуры системы

Целью этапа проектирования архитектуры системы является привязка верифицированных и валидированных операционных контрактов к элементам физической архитектуры системы. Привязка является итеративным процессом к которому привлекаются эксперты из различных инженерных дисциплин. Они могут проанализировать различные архитектурные концепции, с учетом требований к производительности, безопасности и стоимости, определенные на этапе анализа требований.

Проектирование архитектуры системы начинается с определения физических подсистем. Для этого используется такой артефакт SysML как *структурная диаграмма*. Элементами модели являются блоки действующих лиц и системный блок. Частями системного блока являются физические подсистемы, определенные на основе выбранной архитектуры.

Следующие несколько шагов выполняются итеративно для каждого варианта использования.

Определенные ранее операционные контракты для системы в рассматриваемом варианте использования привязываются к физическим подсистемам с использованием диаграммы деятельности "белого ящика". Эта диаграмма является практически копией диаграммы деятельности "черного ящика". Единственное различие состоит в том, что данная диаграмма содержит разбиение на разделы, каждый из которых соответствует физической подсистеме. На основе выбранной архитектуры, операционные контракты системы размещаются на диаграмме в разделах соответствующих подсистем. Важным постусловием для данной привязки является то, что изначальные связи (последовательность действий) между операционными контрактами сохраняются.

Помимо диаграммы деятельности "белого ящика" для рассматриваемого варианта использования создаются диаграммы последовательности "белого ящика". Они представляет собой декомпозицию ранее определенных диаграмм последовательности "черного ящика" и используется для определения интерфейсов физических подсистем. На диаграммах последовательности "белого ящика" линия жизни системы расщепляется на множество линий жизни соответствующих подсистем. В соответствии с привязкой сделанной на диаграмме деятельности “белого ящика”, операционные контракты подсистем перемещаются на соответствующие линии жизни. Для сохранения изначально определенной последовательности действий может возникнуть необходимость определить дополнительные запросы сервисов между физическими подсистемами. Их определение дополняет интерфейсы между подсистемами.

На основе полученных сценариев "белого ящика" между физическими подсистемами определяются связи и соответствующие порты и интерфейсы.

Шаги определенные выше повторяются для всех вариантов использования системы.

После обработки всех вариантов использования для каждой физической подсистемы определяется поведение с использованием *диаграммы состояний*. Диаграммы состояний создаются на основе полученных сценариев “белого ящика”

Корректность и полнота модели архитектуры системы проверяется путем ее исполнения. После того, как функциональность модели верифицирована, может быть проведен анализ архитектуры на соответствие требованиям к производительности и безопасности с привлечением соответствующих инженерных методов.

5.2. Этап проектирования архитектуры подсистем

На данном этапе основной фокус делается на определении способа реализации привязанных к подсистемам операционных контрактов.

Проектирование архитектуры подсистем производится отдельно для каждой подсистемы. Последовательность шагов для каждой подсистемы аналогичен этапу проектирования архитектуры системы. Ниже определена последовательность

шагов, которая выполняется итеративно для каждого варианта использования подсистемы.

На первом шаге принимается решение о том, какие операционные контракты физической подсистемы следует реализовать в аппаратуре (механически или с использованием электроники), а какие с помощью программного обеспечения. Для этого используются расширенные диаграммы активности “белого ящика”. Для операционных контрактов, которые затрагивают более чем одну инженерную дисциплину, потребуется дополнительный анализ. В данном анализе могут участвовать эксперты из различных инженерных дисциплин, работающие над подсистемой.

Для каждого сценария "белого ящика" рассматриваемого варианта использования , линии жизни физической подсистемы расщепляются на линии жизни аппаратных и/или программных компонентов. В соответствии с выбранной архитектурой, операционные контракты помещаются на соответствующие линии жизни компонентов, а определенная ранее последовательность взаимодействия между подсистемами устанавливается через соответствующие запросы сервисов к компонентам.

На основе полученных расширенных сценариев “белого ящика” определяются порты и интерфейсы компонентов подсистем.

Шаги определенные выше повторяются для всех вариантов использования физической подсистемы.

После этого для каждого компонента физической подсистемы определяется поведение с использованием диаграммы состояний.

Полученная модель архитектуры подсистемы проверяется путем регрессионного тестирования.

По окончанию этапа проектирования архитектуры подсистем полученная модель содержит операционные контракты, привязанные к аппаратным и программным компонентам, и связанные с изначальными требованиями.

Так как диаграммы SysML являются подмножеством диаграмм UML, это дает возможность плавного перехода к разработке программного обеспечения с использование UML. Для перехода на последующую стадию разработки программного и аппаратного обеспечения, для каждой физической подсистемы на основе модели архитектуры подсистемы генерируются следующие документы:

- Спецификация требований для аппаратного и программного обеспечения
- Описание логических интерфейсов между компонентами
- Тестовые сценарии для подсистем и их компонентов полученные на основе сценариев вариантов использования уровня системы.

6. Заключение

В этой статье продемонстрировано, что использование языка SysML позволяет унифицировать процесс разработки систем на основе визуального моделирования с применением функциональной декомпозиции. SysML может рассматриваться в качестве “диалекта” языка UML, понятного как для системных инженеров, так и для разработчиков ПО. Основываясь на текущей спецификации SysML (версия 1.0), может быть определен интегрированный процесс разработки систем и программного обеспечения, позволяющий осуществлять плавный переход между ними.

Литература

[1] Hans-Peter Hoffman, SysML-Based Systems Engineering Using a Model-Driven Development Approach, Telelogic whitepaper, 1 July 2008, (http://modeling.swd.ru)

[2] B.P. Douglass, Real Time UML Workshop for Embedded Systems, Book, Elsevier, 2007

[3] Tim Weilkiens, Systems Engineering with SysML/UML: Modeling, Analysis, Design, Book, Elsevier, 2006

[4] B.P. Douglass, Real-Time UML Advances in the UML for Real-Time Systems, Book, Addison-Wesley, 2004

[5] B.P. Douglas, Doing Hard Time: Developing Real-Time Systems with UML, Objects, Frameworks and Patterns, Book, Addison-Wesley, 1999