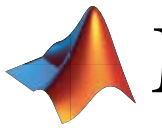
2010-01-0935

Практические стратегии для перехода на модельно-ориентированное проектирование встроенных приложений

Эрик Диллабер, Ларри Кендрик,
Венси Джин и Винод Редди



MathWorks®

MATLAB® & SIMULINK®

Аннотация

При переходе на модельно-ориентированное проектирование для разработки встроенных систем необходимо рассмотреть комплексный план, включающий подбор персонала, процессы разработки и инструменты. Очевидно, что в начале деятельности по улучшению любого процесса следует определить решаемую проблему и затем разработать план реализации решения. При переходе на модельно-ориентированное проектирование наиболее эффективным является итеративный подход — сделать, научиться, настроить и повторить. Конечная цель — это процесс разработки, где модель является частью плана, верификация осуществляется на протяжении процесса разработки с помощью моделирования, а реализация всего приложения на целевом аппаратном обеспечении высоко автоматизирована. Из-за сложностей проектирования и организации, требований к качеству и ограничений по времени и стоимости, переход можно сравнить со сменой пробитой покрышки при движении по автотрассе. Правильный выбор первых шагов является ключевым аспектом успешного перехода. Эта статья представляет набор практических стратегий для определения первых шагов при внедрении модельно-ориентированного проектирования и генерации кода в процесс разработки продукции.

Введение

Все больше организаций, разрабатывающих встроенные системы, желают перейти на модельно-ориентированное проектирование, чтобы увеличить возможности по решению комплексных задач, уменьшить время выхода на рынок, снизить затраты и повысить качество. Многие предприятия сделали первый шаг, построив модели для анализа начальной стадии проектирования или быстрого прототипирования алгоритмов управления. Следующая задача для этих организаций — определить, как эффективно включить модельно-ориентированное проектирование в процессы разработки продукции. Эффективный план перехода требует учета влияния модельно-ориентированного проектирования на предприятие, процессы, инструменты, тестирование и проектирование.

В этой статье представлен набор идей и стратегий для перехода на модельно-ориентированное проектирование для разработки встроенного программного обеспечения. Ключевым компонентом приведенных здесь стратегий является поэтапное внедрение: от первого применения продукции до оптимизированной интеграции персонала, процессов и инструментов для реализации концепции модельно-ориентированного проектирования. Также в статье представлены связанные с таким внедрением общие трудности и лучшие современные отраслевые технологии для их решения.

Эта статья содержит три основных раздела:

1. Организационные трудности.
2. Стратегия по внесению изменений.
3. План перехода для производственных процессов и инструментов.

В разделе **«Организационные трудности»** приводятся ответы на вопросы:

- Что модельно-ориентированное проектирование значит для сотрудников организации?
- Предполагается ли обучение?
- Будет ли необходимо осуществление новых видов работ?
- Следует ли реорганизовывать персонал, чтобы получить все преимущества модельно-ориентированного проектирования?
- Как входные и выходные параметры организации, разрабатывающей встроенные системы, изменятся при переходе на модельно-ориентированное проектирование?

В разделе **«Стратегия по внесению изменений»** рассматривается оптимальный метод перехода на модельно-ориентированное проектирование. Он является поэтапным и включает определение структурированной системы критериев для выбора первого проекта. Для этого необходимо получить ответы на следующие вопросы:

- Какую проблему необходимо решить?
- С какого проекта следует начать?
- Какой возврат инвестиций (ROI) необходим, чтобы подтвердить эффективность модельно-ориентированного проектирования?
- Как снизить риски при переходе?
- Как эффективно осуществить перенос большой существующей базы кодов?
- Каков следующий шаг после первоначального проекта?

В разделе **«План перехода процессов и инструментов»** рассматриваются многие важные детали, на которые необходимо обратить внимание при переходе. Подробно анализируется каждая стадия процесса разработки, включая составление требований, проектирование, реализацию и верификацию. Даются конкретные рекомендации для первоначального проекта и оптимизированного внедрения модельно-ориентированного проектирования.

Организационные трудности

При переходе на модельно-ориентированное проектирование становится очевидной необходимость внедрения новых инструментов и обучения инженерного персонала. Потребность развития организации для получения наилучшего результата при использовании модельно-ориентированного проектирования менее очевидна, но также критически важна. Необходимость вызвана следующими тремя факторами:

1. Новые инженерные задачи.
2. Акцент на составление требований и проектирование.
3. Интеграция процессов и инструментов.

Новые инженерные задачи. Переход на модельно-ориентированное проектирование часто привносит новые задачи, которые ранее решались на физических прототипах или не решались вовсе. Так, например, до перехода автомобильной промышленности на модельно-ориентированные методы калибровки, прототипы двигателей на протяжении сотен часов подвергались динамометрическим испытаниям для определения поведения двигателя по характеристикам нагрева, выбросов, мощности и эффективности. Если в ходе проектирования обнаруживались проблемы, начиналась новая итерация разработки.

После перехода на модельно-ориентированное проектирование для решения аналогичных задач производители двигателей проводят моделирование поведения объектов. Также моделирование применяют для выполнения тестов, которые невозможно провести на реальном объекте из-за различных ограничений, в том числе связанных с безопасностью. Чтобы получить все преимущества ранней верификации в модельно-ориентированном проектировании, организация-разработчик должна решать новые инженерные задачи, такие как создание модели объекта. В некоторых случаях это может означать необходимость добавления в организацию новых подразделений. Учитывая приведенный выше пример, моделей поведения, подходящих для верификации разработанных систем управления и встроенного программного обеспечения, может и не быть, если отдел использовал для разработки двигателя традиционные методы автоматизированного проектирования (CAD). Опыт создания физических моделей доказывает свою эффективность для процесса разработки в целом, благодаря устранению зависимостей от внешних систем управления. Полученные модели могут быть направлены непосредственно на разработку систем управления, что ведет к быстрому моделированию и соответствующему уровню детализации в областях, важных для разработки систем управления или диагностики.

Методы взаимодействия с другими организациями, ведущими разработки в области электричества или механики, также могут измениться при введении практики создания моделей. В некоторых компаниях используется единая модель для организации совместной работы разработчиков аппаратуры и встроенных систем. Одна и та же модель используется разработчиками аппаратного обеспечения и систем управления. Необходимо проявлять осторожность при многозадачной работе с подобной моделью, так как уровень требуемой детализации меняется в зависимости от пользователей. Разработчикам аппаратного обеспечения необходимы сложные модели, которые долго исполняются, в то время как для проведения исчерпывающих тестов разработчикам встроенных систем может потребоваться возможность работы в реальном времени. Часто эти два ограничения приводят к тому, что модель оказывается или слишком сложной, или пригодной только для разработки встроенных систем. Модели объектов, которые скорее прогнозируют тенденции, чем дают точные оценки, более удобны как для анализа, так и для валидации на основе моделирования. Таким образом, если модель объекта создается не разработчиком систем управления, а кем-либо другим, то необходимо поддерживать тесное взаимодействие, гарантирующее, что модель будет отвечать нуждам модельно-ориентированного проектирования.

Новые навыки будут необходимы независимо от того, кто создает модели. Тесное взаимодействие между разработчиками моделей объектов и разработчиками встроенных систем управления и диагностики — подтвержденная эффективная стратегия при переходе на модельно-ориентированное проектирование. Чтобы создать исходную модель, следует обучить собственные кадры, либо, для первоначального проекта, задействовать сторонних специалистов, в то время как собственные сотрудники компании обучаются в процессе работы.

Акцент на составление требований и проектирование. Организации необходимо сконцентрировать внимание и ресурсы на этапах составления требований и разработки проекта с целью получить выгоду от верификации на ранних стадиях. Известно, что инженеры-разработчики встроенных систем тратят много времени на исправление дефектов, обнаруженных на поздних стадиях разработки, но допущенных на более ранних этапах.

Помимо автоматической генерации кода, ранняя и непрерывная верификация и валидация разработок — краеугольный камень модельно-ориентированного проектирования. Фактически, полная рентабельность инвестиций не может быть достигнута без увеличения затрат на создание моделей за счет введения валидации и верификации. Как результат, инженеры, ранее занимавшиеся решением проблем, связанных с реакциями объекта на воздействия, теперь будут создавать надежные средства тестирования,

объединяя свои разработки и тестовую инфраструктуру с требованиями и проводя тщательную валидацию разработок с помощью моделирования. Проведение итоговой верификации реализованной разработки с использованием той же тестовой инфраструктуры окончательно доказывает эффективность затрат на создание моделей, подтверждая верность подхода с применением предварительных работ перед созданием первого аппаратного продукта, и позволяет инженерам проводить итерации в безопасной среде до тех пор, пока не будут выполнены все требования.

Интеграция процесса и инструментов. Организация должна интегрировать новые инструменты с новыми процессами для максимального увеличения эффективности, достигаемого с помощью модельно-ориентированного проектирования. Автоматизация ключевых этапов процесса обычно осуществляется как часть реализации. Наиболее успешные организации уходят от текущих методов работы, чтобы исследовать совершенно новые подходы. Например, большинство организаций, для которых автоматическая генерация кода является чем-то новым, некоторое время продолжают по инерции проверять код традиционными методами.

В конечном счете, процесс ставится под сомнение, и в результате внимание переключается с проверки кода на проверку моделей. Так как разработчики очень редко проверяют код перед компилированием, ручная проверка кода C или HDL со временем останется в прошлом. Модели превращаются в единый источник, по которому можно судить о верности разработки. Следуя дальше по этому пути, полная интеграция процесса и инструментов устанавливает механизмы проверки моделей, которые автоматизируют большую часть рутинных аспектов ручной проверки. Этот подход позволяет избежать чрезмерных затрат на верификацию и позволяет инженеру решать дополнительные задачи, такие как формальная верификация или оптимизация разработки.

Если группа разработчиков уже имеется, поддержка среды модельно-ориентированного проектирования может стать естественным расширением ее обязанностей. Если такой группы не существует, должна быть назначена специальная группа по решению комплексных задач для представления ей функций управления программами, инженерной поддержки систем и средств управления, а также решения проблем, связанных с программным обеспечением. Сфера деятельности группы должна включать следующие аспекты:

- Определение и осуществление необходимых изменений или интеграций в процессе и инструментах, которые включают построение автоматических интерфейсов связи с другими инструментами, такими как системы управления конфигурацией или системы управления текстовыми требованиями, построение систем отслеживания дефектов и систем управления проектами.
- Создание среды и процессов, используемых разработчиками приложений.
- Определение и соблюдение стандартов, связанных с модельно-ориентированным проектированием. Эти стандарты критически важны для ведения совместной разработки, характерной для крупных проектов. Совместная разработка может быть проведена различными отделениями одной организации, удаленно расположенными подразделениями организации или поставщиками. Например, стандарты стиля описания архитектуры и моделей необходимы для общего понимания и возможности производства полученных разработок, а также интегрирования компонентов, поступающих из разных источников.
- Конфигурирование и подготовка инструментов.
- Определение процедур и стандартов взаимодействия с поставщиками.

Модельно-ориентированное проектирование не заменяет инженерного опыта в разработке систем управления и создания архитектур программного обеспечения. При использовании модельно-ориентированного проектирования роль инженеров по системам управления расширяется от составления требований до их реализации, благодаря инструментам генерации финального кода и верификации. Инженеры по программному обеспечению, с другой стороны, продолжают играть ключевую роль, несмотря на то, что их деятельность сместится в сторону проектирования архитектуры программного обеспечения, настройки инструментов по генерации кода, создания платформы на базе программного обеспечения и системной интеграции.

Таким образом, внедрение модельно-ориентированного проектирования требует новых знаний и усиливает разделение между разработкой приложений и платформ. Проведение этих изменений требует обучения и организационных нововведений, чтобы с помощью модельно-ориентированного проектирования добиться максимального прироста качества и эффективности.

Стратегия по внесению изменений

План изменений

Введение модельно-ориентированного проектирования должно сопровождаться необходимыми изменениями процесса, которые могут включать в себя принятие таких стандартов, как AUTOSAR, ISO 26262 и DO-178C, разработку передовых технологий (например, гибридных электрических транспортных средств и ветроэнергетических систем) или инициативу непрерывного улучшения.

Определение решаемой задачи

Перед переходом на модельно-ориентированное проектирование необходимо иметь полное понимание неэффективности и убыточности текущего процесса и организации. Необходимо ввести показатели, чтобы поставить цели, количественно рассчитать неэффективность и определить ключевые области, специфические стратегии и инструменты.

Часто число детализированных показателей текущего процесса и организации ограничено. В таких случаях промышленные показатели служат хорошей отправной точкой для определения ключевых областей, на которых следует сконцентрировать улучшение процесса. Известны публикации, в которых приводится описание типичных промышленных показателей [2 – 4]. Одно из исследований показывает, что основной источник затруднений и связанных с этим задержек во всех отраслях промышленности — исправление ошибок в спецификации и верификация разработки. Согласно другому исследованию, 60 % дефектов вносятся в спецификацию, причем 55 % этих дефектов не обнаруживаются до стадии тестирования, что приводит к неэффективному проектированию и отладке требований инженерами по тестированию и настройке. Относительная стоимость исправления ошибок на поздних стадиях процесса разработки представлена в [2]. Зачастую имеются другие показатели неэффективности процесса, такие как время, которое тратят инженеры по настройке на отладку программного обеспечения контроллеров.

В итоге, сосредоточение мероприятий по улучшению процесса на устранении дефектов и убытков приводит к большему возврату инвестиций.

Выбор проекта с подходящим уровнем сложности и технологичности

Какие проекты являются лучшими кандидатами для начального внедрения модельно-ориентированного проектирования? Проекты, требующие минимальных изменений, не очень хороши из-за того, что в них мало возможностей получения экономии. Контроллер двигателя, перенесенный на модель следующего года с минимальными изменениями в настройках, не сможет мотивировать к переходу на создание моделей, в отличие от проекта с большим числом новых особенностей, такого как новый контроллер гибридной силовой передачи. Если проект критически важен для успеха компании, иногда ведется резервное проектирование традиционными методами. Этот подход призван снизить риски и предоставить приемлемые показатели качества для процесса и инструментов. Большинство организаций, однако, выбирают или немного менее рискованный проект, или тот, в котором модельно-ориентированное проектирование является единственным способом уложиться в проектный график.

Типичная встроенная система затрагивает несколько инженерных дисциплин. Важно, что компоненты выбранного проекта могут быть представлены в разных средах создания моделей, доступных при модельно-ориентированном проектировании. Например, алгоритмы управления, описанные с помощью блок-схемы, лучше всего реализуются в таких инструментах создания моделей, как Simulink. Алгоритмы, включающие сложную логику, диспетчерское управление, блок-схемы для управления отказами и конечные автоматы, лучше всего могут быть описаны с помощью таких инструментов, как Stateflow. Компоненты с сильной привязкой к аппаратному обеспечению, например, операционные системы или драйвера устройств, лучше подходят для низкоуровневых языков, таких как C. Там, где компонент не вписывается в рамки одного инструмента, часто можно провести параллельное моделирование, чтобы учесть моделирование полной системы при верификации и валидации.

Поэтапный переход для снижения рисков

С целью минимизации рисков важно сначала выяснить, каким образом можно согласовать между собой работу инструментов для модельно-ориентированного проектирования, чтобы соответствовать поставленным целям. Производственные организации редко имеют возможность приостанавливать разработку на время проведения крупных изменений в процессе. Часто возникают вопросы о том, как это повлияет на время проектирования, сможет ли технология соответствовать требованиям приложения и будет ли итоговый код достаточно эффективным. Эти вопросы помогает разрешить поэтапный переход на модельно-ориентированное проектирование, который предусматривает разделение развертывания модельно-ориентированного проектирования на этапы и оптимизацию процесса с учетом опыта, полученного после прохождения очередного этапа. Этот подход увеличивает защищенность и минимизирует риски. Необходимо проявлять осторожность при расчете времени обучения при планировании и получении выводов о потенциальном увеличении эффективности начального проекта.

Поэтапный подход, показанный на рис. 1, начинается с разработки начального плана перехода, который используется для последующего масштабированного внедрения сначала отдельных компонентов, а затем всего приложения. Полное развертывание приложения сопровождается этапом оптимизации процесса и расширенного внедрения, после которого следует обслуживание и обновление как часть непрерывного улучшения. Каждый этап должен иметь четко определенные цели, показатели и контрольные точки, и включать последующую оценку и уточнение плана перехода. Действия на каждом этапе охватывают и создание моделей, и использование приложений, улучшающих процесс. Улучшение инструментов и процессов для модельно-ориентированного проекти-

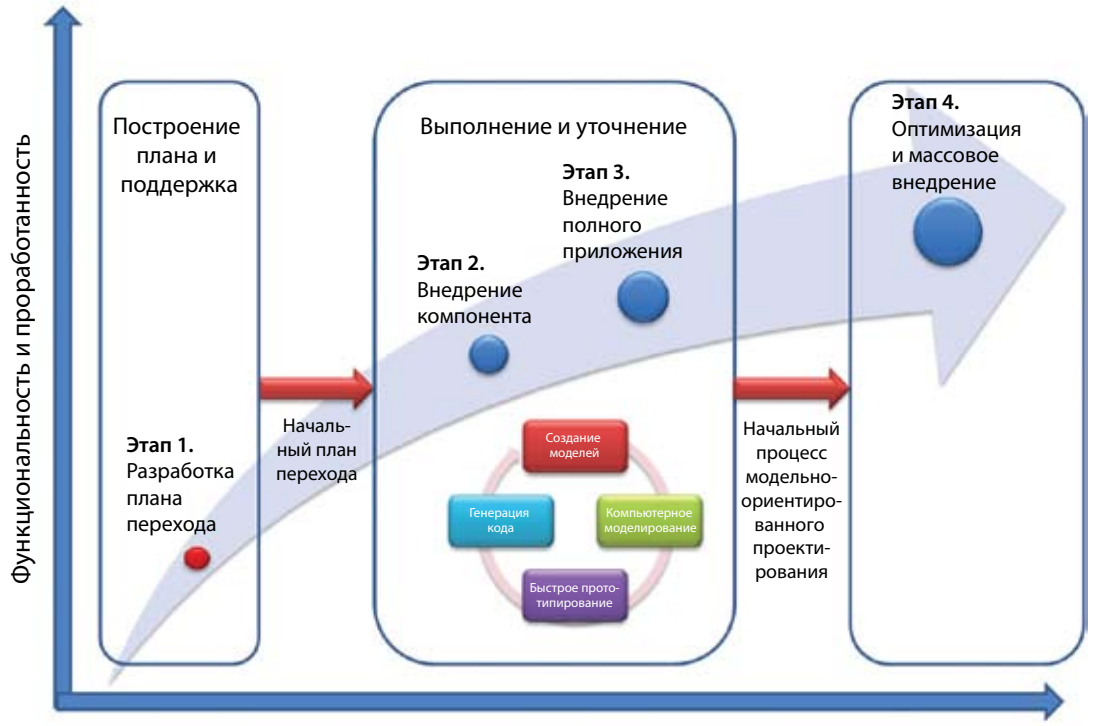


Рис. 1. Поэтапный подход к развертыванию

рования является постоянным. Введение новых функций, выявление дополнительных возможностей для снижения количества ручных действий и анализ стандартов и методов проектирования будут непрерывно продолжаться для повышения эффективности и уровня качества.

Этап 1. Разработка начального плана перехода

Цель этого этапа состоит в определении инструментов и функциональных возможностей, необходимых для выполнения требований приложения, улучшения процесса и понимания того, как инструменты интегрируются со связанными элементами для разработки. Эта стадия обычно начинается с формального обучения работе с инструментами для понимания их возможностей. Следующий шаг состоит в привязке требований по приложениям и процессам к специфическим инструментам и функциональным возможностям, необходимым для их выполнения. Последний шаг — применить полученные знания на практике, осуществив процесс разработки с эталонной моделью. Данный этап обычно осуществляется посредством разработки простой модели, генерации кода из модели, компиляции кода и загрузки кода на целевое устройство. Всегда доступная поддержка производителей инструментов сильно снижает время, необходимое на обучение. Эта стадия также демонстрирует возможность использования инструментов поддержки построения моделей для любого последующего масштабного внедрения. Результатом этапа является начальный план перехода, используемый для последующего масштабированного внедрения сначала одного компонента, а затем и всего приложения.

Этап 2. Внедрение компонента

Цель этапа внедрения компонента состоит в тестировании и уточнении новых возможностей за счет разработки одного компонента с помощью модельно-ориентированного проектирования при ограничении воздействия. Сгенерированный код рассматривается как написанный вручную и вручную интегрируется со стандартным рабочим приложением, созданным для этого этапа. Уточнения, определенные на этом этапе, вносятся в план перехода как способы улучшения процесса.

Этап 3. Внедрение всего приложения

Цель этапа внедрения всего приложения состоит в разработке полного приложения с помощью модельно-ориентированного проектирования, генерации кода и автоматического генерации и загрузки исполняемых файлов на целевое устройство. Процесс интеграции с базовым программным обеспечением (операционная система, драйверы устройств и коммуникационные стеки) полностью автоматизирован, равно как и процессы генерации и загрузки.

Этап 4. Оптимизация и расширение внедрения

Цель этапа расширенного внедрения состоит в оптимизации и развертывании процесса на более широком уровне. Рекомендации, полученные на предыдущих этапах, учитываются при подготовке к развертыванию. Действия на данном этапе сконцентрированы на улучшениях процесса с целью соответствия более широким нуждам организации и дальнейшего улучшения функциональности и проработанности процесса в таких областях, как учет требований, автоматическая генерация документа-

ции и более широкое использование моделей объектов для непрерывной верификации.

Выбор подходящих имеющихся инструментов для осуществления перехода

Многие производственные организации имеют большое количество собственного кода, написанного для предыдущих разработок. Нужно ли весь собственный код переводить в модели? Если так, нужно ли привлекать для этого специалистов или можно обойтись собственными силами? Сколько сил будет на это затрачено и каковы будут выгоды?

Как было замечено в статье «Best Practices for Establishing a Model-Based Design Culture» [1], переход предоставляет огромные возможности для обучения. Во встроенных системах управления, разрабатываемых годами разными авторами, вполне обычны сегменты кода, которые не до конца понятны текущим разработчикам. Процесс создания моделей существующих алгоритмов позволяет разработчику глубже понять эти области, повышая при этом ясность, качество и удобство сопровождения.

При наличии неоспоримых преимуществ перевода собственного кода в модели, очевидными недостатками являются затрачиваемые усилия и риск. К счастью, нет необходимости осуществлять переход всего приложения для получения преимуществ использования модельно-ориентированного проектирования. Использование модельно-ориентированного проектирования предоставляет средства для интеграции с собственным кодом, которые поддерживают одновременно имитационное моделирование и генерацию кода. Модель архитектуры системы может, таким образом, содержать как созданные модели компонентов, так и существующие компоненты. В этом смешанном подходе учитывается поэтапный переход существующих компонентов, и при этом поддерживается возможность моделирования и верификации системы и генерации кода. На рис. 2 показаны компоненты типичной архитектуры встроенного программного обеспечения. Затемненные компоненты на рис. 3 показывают стратегический переход программных компонентов в область создания моделей во времени. Незатемненные компоненты остаются в виде собственного кода.

Следует осуществить аккуратную проверку и анализ существующей базы кода, чтобы определить приоритетность перехода каждого компонента. Приоритет перехода компонентов должен определяться по следующим признакам:

- частое изменение или необходимость изменения в будущем;
- проблемы с качеством;
- высокая сложность, трудоемкость сопровождения;
- возможность представления в виде модели.

Компоненты, не удовлетворяющие этим критериям, не дают больших преимуществ при их переводе в модели.

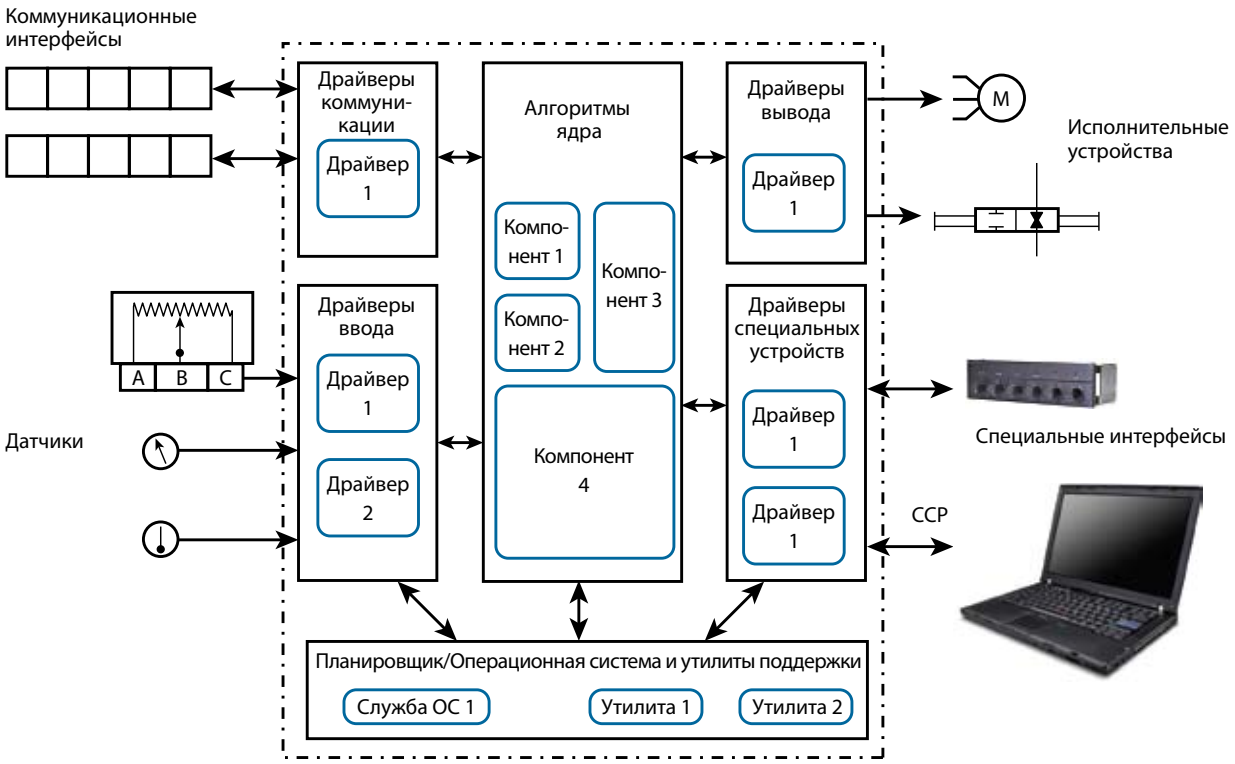


Рис. 2. Общая архитектура существующего приложения

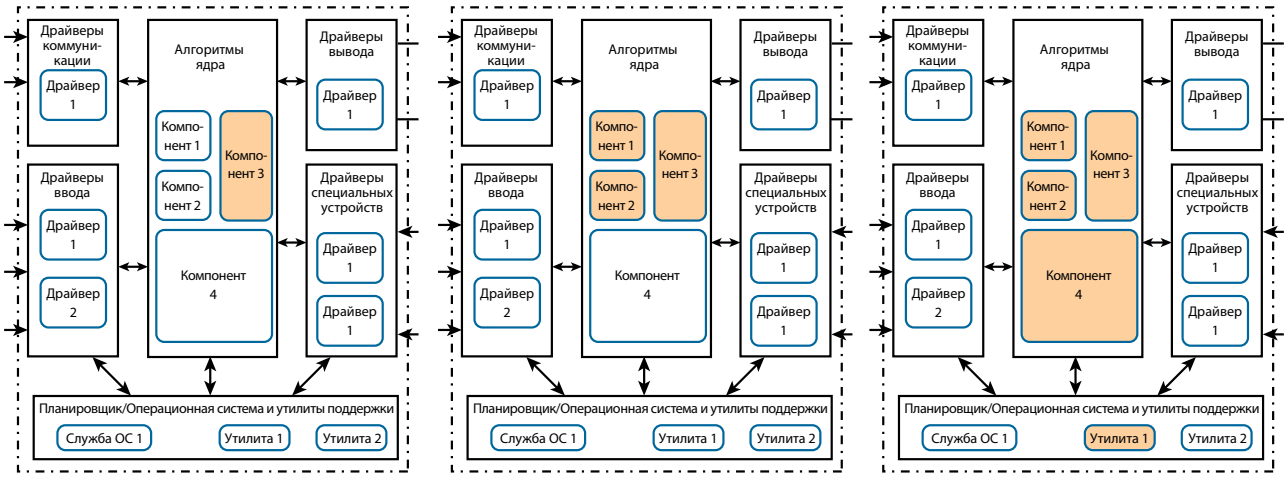


Рис. 3. Этапы с 1 по 3 перехода существующих компонентов

План перехода для производственных процессов и инструментов

Модельно-ориентированное проектирование предоставляет большой диапазон инструментов и методов для облегчения каждого этапа процесса разработки встроенного программного обеспечения. Разработка начального плана перехода для процесса и инструментов требует понимания того, как действия процесса модельно-ориентированного проектирования отличаются от традиционных операций. В этом разделе представлен обзор действий в модельно-ориентированном проектировании и набор практических стратегий для каждого этапа процесса. Стратегии разрабатываются для того, чтобы помочь организации понять, какие подходы следует внедрить в первую очередь и чего следует избегать. Эти рекомендации основаны на накопленном опыте в деле помощи организациям по осуществлению перехода.

На рис. 4 представлен типичный процесс разработки встроенного программного обеспечения, включающий в себя составление требований, проектирование, реализацию, интеграцию и этапы тестирования. Непрерывная верификация, управление конфигурацией и изменениями распространяются на каждый этап разработки.

Этап составления требований

Составление спецификации по требованиям — это процесс анализа и документирования требований и ограничений, которым должна удовлетворять разрабатываемая система. Возможность использовать модель для создания выполняемой спецификации и моделирования позволяет анализировать и проводить валидацию требований на ранних этапах и на ранее не достижимом глубоком уровне.

Разработка выполняемой спецификации как возможность установить формальные требования

Отраслевые исследования показывают, что ошибки в требованиях являются наиболее распространенным источником ошибок при разработке встроенных систем и обходятся наиболее дорого, так как обычно их не обнаруживают до проведения интеграции системы и тестирования в конце процесса. Один из наиболее значимых источников снижения времени проектирования и экономии на расходах в модельно-ориентированном проектировании — это раннее обнаружение и исправление ошибок в требованиях. Однако если формальные требования в

Рис. 4. Типичный процесс разработки встроенных систем

проекте отсутствуют, разработка выполняемой спецификации как возможность создания формальных требований не обеспечивает мгновенной экономии, но приносит выгоду на дальнейших итерациях проектирования. В некоторых случаях требования для проектирования, на которых основан новый проект, необходимо синтезировать из реализующего его кода. После того как формальные требования установлены, выполняемую спецификацию следует связать с требованиями и полностью интегрировать с процессом управления изменениями.

Проведение валидации требований перед переходом к рабочему проекту

Перед началом рабочего проекта большое количество времени следует потратить на получение, анализ и верификацию требований. Верификация часто предусматривает не только анализ требований. В случае новых или комплексных приложений она будет включать в себя моделирование и быстрое прототипирование для проверки правильности и полноты требований. В случае неудачного исхода необходимо повторить процесс проектирования. Если верификация невозможна без полностью продуманного проекта, можно использовать модель для документирования ожидаемого выхода для каждого из возможных входных воздействий и завершить проектирование упрощенной реализацией.

Взаимодействие с внутренними и внешними клиентами и поставщиками с помощью модели

Спецификации разрабатываются для того, чтобы облегчить взаимодействие между изготовителем комплектного оборудования (OEM) и его поставщиком или между двумя подразделениями одной компании. Модель представляет собой недвусмысленно выполняемую спецификацию, которая может служить идеальным инструментом взаимодействия. Инженеры могут использовать выполняемую спецификацию, чтобы избавиться от одного из наиболее общих источников ошибок — неправильного толкования или неправильного перевода спецификации. Когда выполняемая спецификация используется совместно изготовителем комплектного оборудования и его поставщиком, поставщик должен в качестве контрольных точек проекта проводить проверку выполняемой спецификации вместе с изготовителем комплектного оборудования, показывая, как следует исполнять каждое требование, и подтверждая соответствие системы требованиям.

Модель как источник документации

Модель выполняет функцию ядра спецификации, но также ее можно использовать для производства другой документации, такой как обзорный тестовый отчет и, с использованием точной полученной информации, пользовательское руководство по настройке. Существуют инструменты для автоматизации процессов извлечения, перекомпоновки и представления информации, содержащейся в модели. Автоматизация помогает избежать необходимости подготовки многих не связанных с моделью проектных документов, которые могут плохо отражать текущее состояние проекта.

Этап проектирования

Проектирование — это процесс определения архитектуры и интерфейсов программного обеспечения и разработки детализированных функций и операций, удовлетворяющих требованиям. В модельно-ориентированном проектировании для описания проекта вместо проектных документов используются модели. Элементы, разработанные на этапе составления требований, такие как выполняемая спецификация, непосредственно являются отправной точкой для начального проектирования, тем самым снижая ошибки трансляции и ускоряя разработку. Проектные документы генерируются из модели. Компьютерное моделирование и быстрое прототипирование используются для быстрого повторения процедур разработки и подтверждения того, что проектирование идет согласно плану, и разработка будет удовлетворять требованиям как при моделировании, так и в реальных условиях.

Заблаговременный выбор архитектуры и технологии компонентов

Архитектура модели и технология, используемая для инкапсуляции компонентов в проекте, влияют на ряд областей, включая производительность моделирования, скорость генерации кода, эффективность кода, процесс проверки конфигурации и изменений и удобство эксплуатации. Особенности компонентов, таких как ссылки на модель (model reference), элементарные подсистемы (atomic subsystem) и библиотеки, следует учитывать сразу в начале процесса проектирования для получения масштабируемой архитектуры модели, отвечающей текущим нуждам. Например, ссылку на модель следует использовать для компонентов, которые необходимо независимо верифицировать вне модели, которой они принадлежат, или когда есть необходимость генерировать повторно используемую функцию в программном обеспечении.

Внедрение и закрепление стандартов проектирования (например, MAAB)

Среды для построения моделей обеспечивают высокую гибкость и одновременно с этим предоставляют внешние библиотеки шаблонов проектирования для решения общих задач. При неаккуратном обращении модели могут быстро стать нечитаемыми и неэффективными, снижая эффективность моделирования, генерации кода и передачи информации между участниками проекта. Введение стандартов проектирования позволяет предотвратить создание неэффективных моделей. Они гарантируют, что модель будет пригодна к производству и понятна, и дают возможность избежать общих ошибок проектирования перед переходом к следующему этапу. Начать

можно с принятия промышленных стандартов, таких как «Руководство по оформлению автомобильного консультативного комитета MathWorks» (MAAB), которое можно внедрить с помощью ряда автоматизированных проверок. Если требуется обеспечить критичный по безопасности рабочий процесс, следует также рассмотреть правила создания моделей, соответствующих стандартам IEC 61508 или DO-178B. Автоматическая проверка позволяет убедиться в гибкости разработки и эффективности последующей реализации. Стандарты и автоматические проверки основываются на накопленном отраслевом опыте, и на них стоит обратить повышенное внимание.

Внедрение процесса для упрощения повторного использования

Проектировать следует с учетом возможности повторного использования элементов проекта, но стоит помнить и о способах упрощения повторного использования за счет разделения ресурсов. Это позволит упростить разработку и увеличить ее гибкость. Хотя это лучшая практика с точки зрения здравого смысла, лишь малая группа компаний осуществляет ее, и те, кто это делает, занимают передовые позиции [5]. Ведущие компании-производители, определяемые по их успеху в достижении ряда инженерных целей, объединяют подготовку и верификацию разработок для повторного использования в своих рабочих процессах. В этих компаниях также предпочитают разделять ресурсы для ускорения разработок при повторном использовании. Для упрощения повторного использования в модельно-ориентированном проектировании могут использоваться следующие стратегии:

- Отделение характеристик целевого устройства, таких как типы данных и драйверы устройств, от проектной модели. Эту процедуру можно осуществить, применяя архитектуру с использованием раздельного словаря данных для определения специфических типов целевого устройства.
- Применение ссылки на модель (model reference) для инкапсуляции компонентов, которые можно использовать повторно, с целью облегчения верификации и усиления контекстных независимых границ.

Исключение избыточного проектирования

Модельно-ориентированное проектирование позволяет быстро проводить сложные разработки. Обычно в модели есть несколько путей описания одного и того же функционала. Следует затратить определенные силы для обзора модели. Это позволит определить шаблоны проектирования, которые могут улучшить читаемость и производительность, как в следующих примерах:

- Использование конечных автоматов только в случае, если выходом схемы состояний является функция внутренних состояний, а не только входных сигналов. Если выходной сигнал является функцией только входных сигналов, лучше использовать блок-схему.
- Использование руководства по созданию конструкций на языке C для того, чтобы эффективно представить общие конструкции языка C при создании модели.
- Открытие общего доступа к библиотеке часто используемых шаблонов проектирования для совместного использования вместе с другими членами команды разработчиков.

Разработка модели объекта с «правильным поведением» («trend-correct»)

Модели объекта критически важны для проведения ранней и непрерывной верификации на протяжении всего пути от начальной идеи до реализации продукции. Возможно построение модели с разной степенью достоверности. Как правило, высокая достоверность требует дополнительных усилий при разработке. Модель, описывающую реальный объект с определенными приближениями, будем называть моделью с «правильным поведением». Модель объекта с «правильным поведением» можно использовать для подтверждения правильной работы модели контроллера или для регрессионного тестирования при изменении контроллера. Такая модель использует эмпирические данные вместе с физическими законами.

Модель объекта ограничена по сложности и обеспечивает приблизительную связь между выходом и входом (обратную связь) контроллера, и, как правило, может быть разработана с привлечением небольших ресурсов. Часто модели с «правильным поведением» разрабатываются инженерами, проектирующими стратегии управления.

Приоритетная задача — определение целей для использования модели объекта

Организации, которые успешно внедрили модельно-ориентированное проектирование, в основном начинали с моделей с «правильным поведением» и со временем улучшали эти модели, делая их более прогнозируемыми. Такой подход обеспечивает моментальную выгоду за счет возможности проверить правильность поведения стратегий управления и непрерывного улучшения модели объекта для поддержки разработки контроллера.

Повышенная прогнозирующая способность требуется в случаях, когда модель объекта нужно использовать для валидации стратегии управления. Прогнозирующая модель высокой точности потенциально может снизить или исключить дорогостоящие циклы тестирования с использованием прототипов транспортных средств. Однако следует тщательно спланировать требуемое время проектирования и проведения технической экспертизы и итоговую скорость моделирования. Когда модель описывает реальный объект более точно, и модель, и процедура ее

валидации становятся более сложными. Часто необходимо привлекать к работе экспертов с более детальными знаниями моделируемого объекта. Зачастую таких экспертов можно найти в подразделении организации, ответственном за проектирование физического механизма. Экспертам следует плотно работать с конечными пользователями модели, чтобы гарантировать подходящий уровень детализации модели.

Этап реализации

Реализация — это процесс перевода разработки во встроенное программное обеспечение, которое можно запускать на целевом аппаратном обеспечении. В модельно-ориентированном проектировании перевод из разработки во встроенное программное обеспечение автоматизирован за счет генерации кода, которая значительно снижает количество ошибок [12] и экономит время на написание кода. Основное направление разработки может сместиться в сторону создания интеллектуальной собственности, такой как модели разработки, которая является основой конкурентного преимущества и прибыльности компании.

Изменение модели вместо редактирования кода

К модели стоит относиться как к единственному эталону разработки. Изменять сгенерированный код напрямую не рекомендуется. Вместо этого следует вносить изменения в модель и повторно генерировать код, чтобы избежать лишних затрат и проблем с синхронизацией. В традиционных процессах это равносильно тому, чтобы воспринимать код C как источник и воздерживаться от внесения изменений на уровне ассемблера. Затраты на верификацию, возникающие при внесении изменений, можно снизить, убедившись в том, что разработка разделена на подходящие по размеру компоненты ссылок на модели (model reference).

Автоматизация внедрения и построения кода

Генератор кода, такой как Real-Time Workshop Embedded Coder, производит не зависящий от целевого устройства код ANSI C/C++. Если в компании определен процесс разработки и компиляции кодов, на первом этапе внедрения автоматически сгенерированный код следует использовать как написанный вручную и интегрировать его в существующий процесс. Рекомендуется использовать возможности инструментов для автоматизации процесса экспорта кода в имеющуюся среду разработки программного обеспечения.

Если в компании нет определенного процесса разработки и компиляции или существует необходимость в его дальнейшей автоматизации, следует воспользоваться инструментами для связи со средой разработки (IDE Link) и пакетом целевой поддержки Target Support Package для получения полностью автоматизированного процесса генерации и загрузки приложения на целевое устройство из среды создания моделей. Этот подход позволяет разработчикам сконцентрироваться на своих проектах и быстро проводить итерации цикла разработки и верификацию непосредственно на целевом устройстве.

Уменьшение расширенной ручной настройки для снижения затрат на обслуживание

Real-Time Workshop Embedded Coder, как и другие генераторы кода, предоставляет множество встроенных возможностей конфигурирования для выполнения общих требований к встроенному программному обеспечению в областях стиля кодирования, определения интерфейса, определения данных и форматирования файлов. Если встроенных возможностей недостаточно, можно использовать расширенные настройки с учетом определенных задач. Это позволит обеспечить соответствие более специфическим требованиям к встроенному программному обеспечению.

Проведение расширенной настройки требует гораздо больше усилий для разработки и поддержки и может потенциально увеличить затраты, необходимые для улучшения вновь выпускаемых продуктов. Если возможно, следует использовать встроенные методы конфигурирования, чтобы упростить среду создания моделей и генерации кода. Помимо этого, вместо интегрирования существующих устаревших функций, там, где доступны эквивалентные блоки для создания моделей, следует использовать соответствующий блок, который часто обладает большей функциональностью и той же эффективностью. Например, встроенный блок таблицы поиска обеспечивает поддержку ряда алгоритмов поиска и типов данных, и претерпел ряд улучшений на протяжении многих лет для генерации высокоэффективного кода. В целом, рекомендуется расходовать время на разработку моделей, а не на поддержку инфраструктуры.

Этап верификации и валидации

При модельно-ориентированном проектировании модели и имитационное моделирование используются для ранней и непрерывной верификации и валидации разработок на протяжении всего процесса разработки. Тестирование интеграции программного обеспечения и системы на этапе верификации и валидации (V&V) осуществляется в конце цикла разработки.

Основные цели верификации и валидации состоят в том, чтобы обеспечить соответствие разработки (верификация разработки) и реализации (верификация кода) заявленным требованиям. При использовании модельно-ориентированного проектирования, разработка представляется в виде модели и в завершение процесса демонстрируется, что модель соответствует требованиям, и код, сгенерированный из модели (реализация) также удовлетворяет требованиям, как показано на рис. 5. Инте-

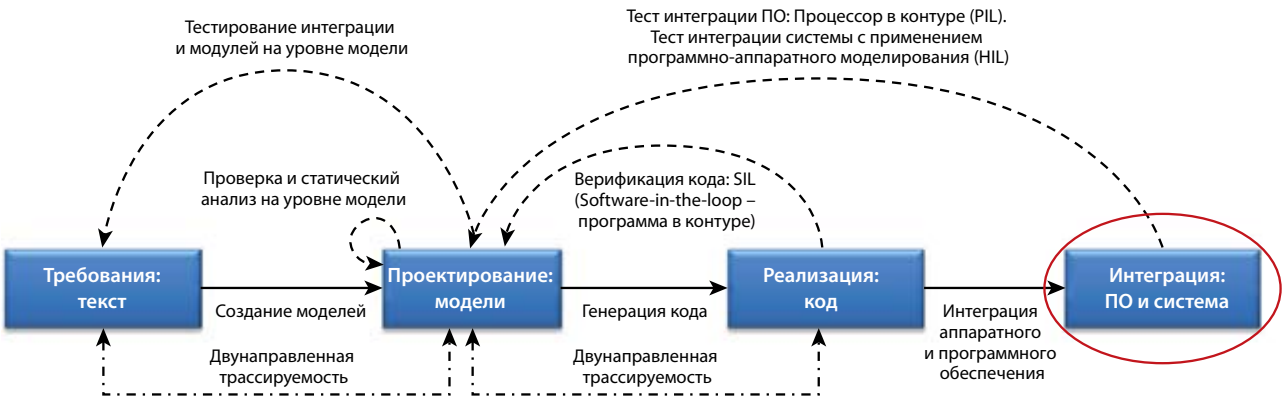


Рис. 5. Процесс верификации и валидации в модельно-ориентированном проектировании

грация программного обеспечения обычно включает в себя интеграцию и валидацию программного обеспечения, сгенерированного из моделей и существующего программного обеспечения. Одним из методов тестирования интеграции программного обеспечения является тестирование с процессором в контуре (PIL). Системная интеграция в общем случае включает в себя использование программно-аппаратного моделирования (HIL) и тестирования на готовом к выпуску аппаратном обеспечении.

Верификация действительно нужных элементов

Первый шаг по внедрению верификации и валидации в проект с использованием модельно-ориентированного проектирования состоит в том, чтобы оценить, какие действия по верификации и валидации необходимы для одновременного удовлетворения требований и временной экономии за счет модельно-ориентированного проектирования. Это определение должно опираться на цели проекта по времени, качеству и стоимости. Очевидная стратегия — выполнять ровно столько операций, сколько необходимо. Кроме того, чтобы снизить затраты на проект и время выхода на нормальный режим работы, нужно тщательно выбирать объемы автоматизации. К полной автоматизации следует прибегнуть только после определения и утверждения этапов процесса верификации и валидации. Промышленные и нормативные стандарты и рекомендации предоставляют целевые показатели процесса разработки, которые вносят дополнительные ограничения и требуют создания дополнительных объектов для демонстрации соответствия. Модельно-ориентированное проектирование поддерживает эти действия через автоматизацию и повторное использование моделей для облегчения ранней и непрерывной верификации. В целом, для оценки необходимости верификации и валидации нужно выполнить следующие действия:

1. Определить стандарты, которые следует принять, и показатели, которых необходимо достичь (промышленные и внутренние показатели). Например, уровень A стандарта DO-178B требует трассируемости объектного кода, в то время как другие уровни этого не требуют.
2. Проанализировать прошлые проекты по программному обеспечению, чтобы определить источник главных дефектов и затрат.
3. Определить дополнительные области для улучшения, такие как стоимость и план. Например, если тестирование единицы не укладывается в график из-за времени, необходимого для создания тестовых векторов для полномасштабного тестирования, эту проблему можно решить, добавив в цели процесса разработки автоматизацию генерации тестового вектора. И наоборот, если генерация тестового вектора вручную не вызывает затруднений, ее автоматизацию не следует рассматривать как первоочередную область для рассмотрения.
4. Определить действия и уровень автоматизации, необходимые для достижения целевых показателей.
5. Выбрать инструменты для облегчения требуемых действий и автоматизации.

Переключите внимание на проверку моделей

Проверка кода является важной частью традиционного процесса разработки. В модельно-ориентированном проектировании модели являются основным источником ошибок, ошибки трансляции при генерации кода минимальны [12], и автоматизированные технологии, такие как тестирование с программой в контуре (SIL), предоставляют возможности дополнительной проверки того, что код соответствует проектным требованиям. Большее внимание уделяется проверке моделей и формальной верификации и валидации. Как результат, время разработки тратится более продуктивно.

ПЕРЕХОД КЛЮЧЕВЫХ ПРОЦЕССОВ ПОДДЕРЖКИ

В производственной среде команды инженеров часто работают над несколькими связанными проектами и параллельно улучшают несколько компонентов. Отсутствие

грамотного управления версиями и конфигурациями компонентов приводит к ошибкам построения и интеграции, уменьшая эффективность проекта. Во многих компаниях управление конфигурированием (CM) уже стало формальной частью процесса разработки. В традиционном процессе разработки код является центром управления конфигурированием, так как он часто наиболее точно отражает разработку, но модельно-ориентированное проектирование переводит внимание на модель, и стратегию управления конфигурированием нужно подстроить для облегчения работы инженеров с моделями. Так как модель и другие элементы разработки реализованы в текстовом формате, с ними можно обходиться как с файлами исходного кода, и необходимость в общем улучшении существующей инфраструктуры управления конфигурированием отсутствует. Следующие рекомендации могут помочь определить, к каким элементам следует применять управление конфигурированием.

1. Разработка состоит из модели, словаря данных и библиотек моделей. Для всех этих элементов следует применять управление конфигурированием, чтобы иметь возможность полного воссоздания разработки. Инструменты поддержки модельно-ориентированного проектирования также позволяют связать разработку с требованиями, чтобы достичь двунаправленной трассируемости между ними. К документации по требованиям также следует применять управление конфигурированием для поддержания ее целостности. Разработка должна иметь тестовую среду, включающую модели тестовых программ, тестовые векторы и ожидаемые выходные сигналы.

   Тестовая среда, управляемая вместе с моделью, гарантирует возможность быстрого и эффективного проведения модельно-ориентированной верификации, включая регрессионное тестирование.

2. Важно, что скрипты автоматизации инструментов и настройки для генерации кода вместе с исходными файлами для существующего кода приложения и файлами, описывающими драйверы ввода-вывода и стеки коммуникаций, должны быть сконфигурированы с моделью, чтобы приложение можно было последовательно воссоздать. Необходимость управления конфигурированием сгенерированного кода при каждом выпуске вместе с моделью зависит от требований компании и проекта к управлению конфигурированием. Например, если имеет место требование, что все файлы, включающие объектный код и образ приложения, должны быть сконфигурированы, то сгенерированный код также следует сконфигурировать.

Набор инструментов для поддержки модельно-ориентированного проектирования обычно включает продукты нескольких производителей, так что открытость является обязательным требованием к любому инструменту для его эффективной интеграции. С управлением конфигурированием процесс ведется аналогичным образом. Например, для поддержки основных возможностей, необходимых для управления конфигурированием, среда MATLAB предоставляет открытый интерфейс прикладного программирования (API), позволяющий взаимодействовать со многими общими системами контроля изменений. Стратегии управления конфигурированием варьируются от компании к компании. Ссылка [14] рекомендует ряд практических стратегий, ассоциированных с моделью для создания конфигураций, включая стратегии по определению набора файлов.

Управление изменениями является более широким понятием, чем управление конфигурированием, и представляет собой процесс, в котором инженерные изменения формально запрашиваются и проверяются, чтобы определить, следует ли их проводить, и затем результаты внесенного изменения документируются. Модельно-ориентированное проектирование никак не влияет на этот процесс. Тем не менее возможности автоматизации, предоставляемые инструментами поддержки модельно-ориентированного проектирования, могут упростить и ускорить выполнение таких задач, как проверка разработки и регрессионное тестирование.

Заключение

За последнее десятилетие многие организации, как небольшие, так очень крупные, успешно перешли на модельно-ориентированное проектирование. Поставщики инструментов, такие как MathWorks, играют ключевую роль в оказании помощи организациям по успешному преодолению трудностей этого перехода. Рассматриваемые стратегии представляют коллективные знания из разных отраслей и ситуаций. Цель статьи заключается в том, чтобы предоставить читателю практические стратегии, которые были успешно применены самыми разными организациями, и облегчить разработку плана перехода к модельно-ориентированному проектированию, учитывающего специфику предприятия.

Существует огромное количество материалов, позволяющих получить структурированные знания и об инструментах и методах, необходимых для модельно-ориентированного проектирования. Помимо знаний, использование опыта других людей, помогавших организациям успешно осуществить переход и первое внедрение, поможет снизить время, необходимое на обучение, избежать общих ошибок и ускорить получение прибыли от инвестиций. Применение лучших практик, выбранных на основе опыта поставщиков инструментов и решений, снижает затраты, необходимые для модельно-ориентированного проектирования.

Литература

1. Paul F. Smith, Sameer M. Prabhu, Jonathan H. Friedman, The MathWorks, Inc. Best Practices for Establishing a Model-Based Design Culture, SAE Paper 2007-01-0777.
2. J.B. Dabney, «Return on Investment of Independent Verification and Validation Study Preliminary Phase 2B Report». Fairmont, W.V.: NASA IV&V Facility, 2003.
3. Venture Development Corporation. «Embedded Software Strategic Market Intelligence Report», Volume 4, December 2007, VDC.
4. Paul Yanik, «Migration from Simulation to verification with ModelSim.» EDA Tech Forum, 2004
5. Aberdeen Group. «The Design Reuse Benchmark Report Seizing the Opportunity to Shorten Product Development», February 2007.
6. Kerry Grand, Vinod Reddy, Gen Sasaki, and Eric Dillaber, The MathWorks, Inc. «Large Scale Modeling for Embedded Applications», SAE Paper 2010-01-0938.
7. Tom Erkkinen and Bill Potter, MathWorks Inc. «Model-Based Design for DO-178B with Qualified Tools», AIAA Modeling and Simulation Technologies Conference and Exhibit 2009, AIAA Paper 2009-6233.
8. Tom Erkkinen, The MathWorks, Inc. Scott Breiner, John Deere. «Automatic Code Generation — Technology Adoption Lessons Learned from Commercial Vehicle Case Studies», SAE Paper 2007-01-4249.
9. Jeffrey M. Thate and Larry E. Kendrick, Caterpillar, Inc. Siva Nadarajah, The MathWorks, Inc. «Caterpillar Automatic Code Generation», SAE Paper 2004-01-0894.
10. Edward Kit, Addison-Wesley, «Software Testing in the Real World».
11. Brett Murphy, Amory Wakefield, Jon Friedman, The MathWorks, Inc. «Best Practices for Verification, Validation, and Test in Model-Based Design,» SAE Paper 2008-01-1469.
12. Bill Potter, «Achieving Six Sigma Software Quality Through the Use of Automatic Code Generation», 2005 MathWorks International Aerospace and Defense Conference: http://www.mathworks.com/aerospace-defense/miadc05/presentations/potter.pdf
13. Mirko Conrad, The MathWorks, Inc. Guido Sandmann, The MathWorks GmbH. «A Verification and Validation Workflow for IEC 61508 Applications», SAE Paper 2009-01-0271.
14. Gavin Walker, Jon Friedman, and Rob Aberg, «Configuration Management Within Model-Based Design», SAE Paper 2007-01-1775.
15. Peter J. Schubert, Packer Engineering, Inc. Lev Vitkin and Frank Winters, Delphi Electronics & Safety. «Executable Specs: What Makes One, and How are They Used?» SAE Paper 2006-01-1357.
16. Arvind Hosagrahara, Paul Smith, The MathWorks, Inc. «Measuring Productivity and Quality in Model-Based Design», SAE Paper 2005-01-1357.
17. Jinming Yang, Jason Bauman, Al Beydoun, Lear Corporation. «An Effective Model-Based Development Process Using Simulink/Stateflow for Automotive Body Control Electronics», SAE Paper 2006-01-3501.
18. The MathWorks, «BAE Systems Achieves 80% Reduction in Software-Defined Radio Development Time with Model-Based Design», http://www.mathworks.com, May 2006.
19. The MathWorks, «Control Algorithm Modeling Guidelines Using MATLAB, Simulink, and Stateflow Version 2.1», http://www.mathworks.com/automotive/standards/maab.html, July, 2007.
20. The MathWorks, «Simulink Report Generator 3.7», http://www.mathworks.com, September 2009.
21. The MathWorks, «Real-Time Workshop Embedded Coder 5 — Developing Embedded Targets», http://www.mathworks.com, September 2009.

Контактная информация

Eric Dillaber, Eric.Dillaber@mathworks.com

Wensi Jin, Wensi.Jin@mathworks.com

Larry Kendrick, Larry.Kendrick@mathworks.com

Vinod Reddy, Vinod.Reddy@mathworks.com

Департамент MathWorks компании Softline

115114 г. Москва, Дербеневская наб., д. 7, стр. 8

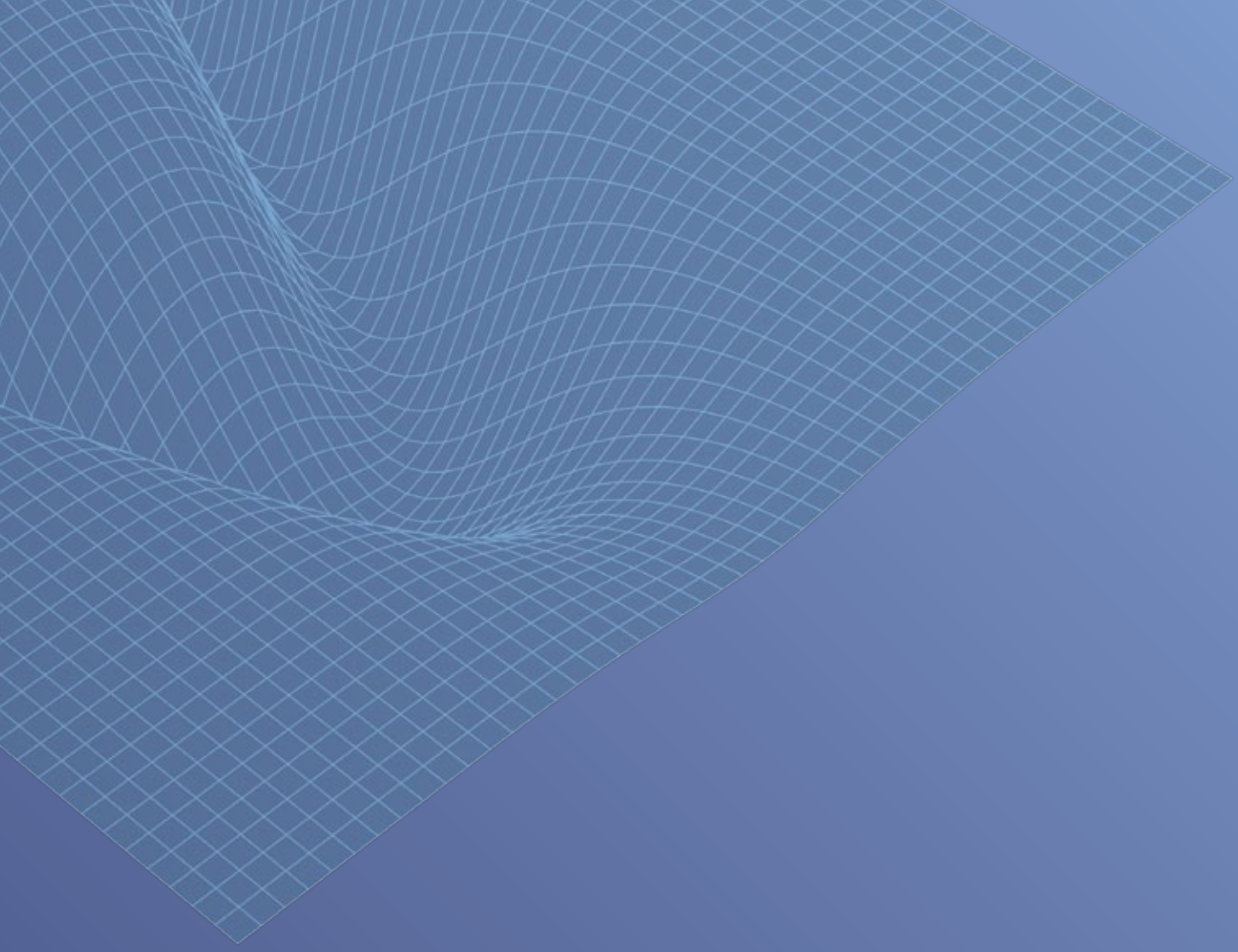
Деловой квартал «Новоспасский Двор»

Email: matlab@softline.ru

Тел.: +7 (495) 232-00-23 доб. 0609

http://www.sl-matlab.ru

http://matlab.exponenta.ru



Департамент MathWorks компании Softline

115114 г. Москва, Дербеневская наб., д. 7, стр. 8
Деловой квартал «Новоспасский Двор»
Email: matlab@softline.ru
Тел.: +7 (495) 232-00-23 доб. 0609
http://www.sl-matlab.ru
http://matlab.exponenta.ru