18 марта 2023 г.

Гаврилов А. Г. Новоселова О. В.

Современные технологии и средства разработки программного обеспечения (второй семестр)
Раздел 6. Современные технологии и средства программной реализации программного продукта

Тема 16
Архитектура, управляемая моделями. Платформенно-зависимая модель ПП (PSM-модель). Технология построения PSM-модели программного продукта с помощью языка моделирования UML и CASE-средств

Содержание

1 Основные понятия 1

2 Архитектура разработки приложений на основе моделей 3
2.1 Типы моделей 3
2.2 Уровни моделей 3
2.3 Этапы разработки 4

3 Преобразование моделей PIM PSM 5

1 Основные понятия

Аббревиатура *MDA* расшифровывается как *Model Driven Architecture* архитектура, управляемая моделью. MDA это архитектура, описывающая способ разработки программного обеспечения. В рамках этой архитектуры создание приложений базируется на разработке модели приложения.

В основе новой архитектуры лежит идея о полном разделении этапов общего проектирования (моделирования) и последующей реализации приложения на конкретной программной платформе. (Рис. 1). Идея эта не нова: сначала при помощи специальных средств проектирования создается общая и независимая от способов реализации модель приложения, а затем осуществляется реализация программы в какой-либо среде разработки. При этом процесс разработки полностью основан на модели, которая должна содержать всю необходимую для программирования информацию. Очевидны преимущества, которые дает такой подход:

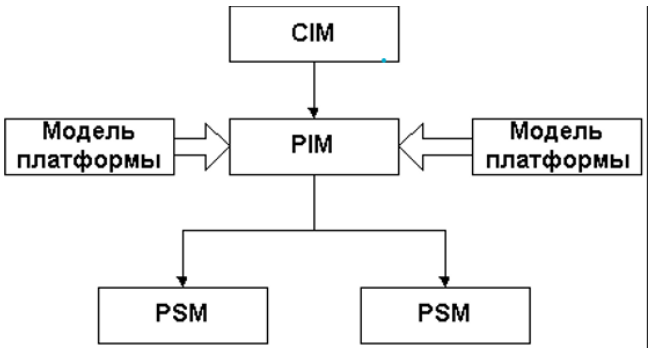

Рис. 1. Архитеркутра MDA

- Независимость модели от средств разработки обеспечивает возможность реализации на любой программной платформе.
- Приложение, реализованное в архитектуре MDA, может быть легко перенесено из одной операционной системы в другую.
- Существенна экономия ресурсов при реализации приложения для нескольких программных платформ одновременно.
- Архитектура позволяет до известной степени автоматизировать процесс программирования. Наличие подробной модели обеспечивает автоматическое создание типовых частей приложения, разработка которых поддается автоматизации. Например, создание пользовательского интерфейса, программирование типовых операций, создание базы данных и организация доступа к данным.

Архитектура MDA возникла не на пустом месте. Само ее появление и возможность реализации обусловило наличие ряда стандартов и технологий, на практике доказавших свою полезность. Концептуальной основой появления MDA стали спецификации OMA, ORB, CORBA. Перевести замысел в практическую плоскость позволили технологии объектно-ориентированного программирования (ООП), стандарт CWM, языки UML, XML, MOF. Работами по созданию новой архитектуры программирования занялся консорциум OMG (Object Management Group) [1].

По мнению создателей, архитектура MDA является новым витком эволюции технологий программирования, так как описывает процесс разработки в целом. Подготовленные читатели с высшим техническим образованием могут возразить, что процесс разработки программного обеспечения описан и стандартизован вдоль и поперек: в нашей стране существует ряд ГОСТов, другие страны также имеют соответствующие стандарты. Но новизна MDA заключается в том, что описание процесса разработки в ней выполнено с использованием современных средств представления и позволяет автоматизировать создание приложений. И весьма вероятно, что через некоторое время архитектура MDA станет общим промышленным стандартом в разработке программного обеспечения.

Прежде чем перейти к описанию собственно архитектуры, необходимо привести некоторые общие термины и определения.

[1] Консорциум OMG создан сообществом IT-компаний, специализирующихся на разработке программного и аппаратного обеспечения. Основной задачей консорциума является стандартизация и спецификация в сфере информационных технологий. При этом деятельность консорциума в значительной степени ориентирована на разработку перспективных стандартов. Программистам известны такие разработки OMG, как OMA, CORBA, UML

Модель — описание или спецификация системы и ее окружения, созданная для определенных целей. Часто является комбинацией текстовой и графической информации. Текст может быть описан специализированным или естественным языком.

Управление на основе модели — процесс разработки системы, использующий модель для понимания, конструирования, распространения и других операций.

Платформа — набор подсистем и технологий, которые представляют собой единый набор функциональности, используемой любым приложением без уточнения деталей реализации.

Вычислительная независимость — качество модели, обозначающее отсутствие любых деталей структуры и процессов.

Платформенная независимость — качество модели, обозначающее ее независимость от свойств любой платформы.

Вычислительно-независимая модель — модель, скрывающая любые детали реализации и процессов системы; описывает только требования к системе и ее окружению.

Платформенно-независимая модель — модель, скрывающая детали реализации системы, зависимые от платформы, и содержащая элементы, не изменяющиеся при взаимодействии системы с любой платформой.

Платформенно-зависимая модель — модель системы с учетом деталей реализации и процессов, зависимых от конкретной платформы.

Модель платформы — набор технических характеристик и описаний технологий и интерфейсов, составляющих платформу.

Преобразование модели — процесс преобразования одной модели системы в другую модель той же системы.

2 Архитектура разработки приложений на основе моделей

Архитектура MDA описывает и структурирует поэтапный процесс разработки любых программных систем на основе создания и использования моделей. При этом используется несколько типов моделей, создаваемых и преобразуемых на различных этапах разработки. Процесс разработки по MDA это последовательное (поэтапное) продвижение от одной модели системы к другой. При этом каждая последующая модель преобразуется из предыдущей и дополняется новыми деталями.

2.1 Типы моделей

Вычислительно-независимая модель (Computation Independent Model, CIM) описывает общие требования к системе, словарь используемых понятий и условия функционирования (окружение). Модель не должна содержать никаких сведений технического характера, описаний структуры и функционала системы. CIM максимально общая и независимая от реализации системы модель. Спецификация MDA подчеркивает, что CIM должна быть построена так, чтобы ее можно было преобразовать в платформенно-независимую модель. Поэтому CIM рекомендуется выполнять с использованием унифицированного языка моделирования UML.

Платформенно-независимая модель (Platform Independent Model, PIM) описывает состав, структуру, функционал системы. Модель может содержать сколь угодно подробные сведения, но они не должны касаться вопросов реализации системы на конкретных платформах. Модель PIM создается на основе CIM. Для создания модели используется унифицированный язык моделирования UML.

Модель платформы описывает технические характеристики, интерфейсы, функции платформы. Зачастую модель платформы представлена в виде технических описаний и руководств. Модель платформы используется при преобразовании модели PIM в модель PSM. Для целей MDA описание модели платформы должно быть представлено на унифицированном языке моделирования UML.

2.2 Уровни моделей

В зависимости от уровня детализации платформы, модели (кроме модели платформы) могут содержать сведения о различных функциональных частях системы. В этом случае говорят об уровнях модели. Обычно различают следующие основные уровни модели.

Уровень бизнес-логики содержит описание основного функционала приложения, обеспечивающего исполнение его назначения. Как правило, уровень бизнес-логики хуже всего поддается автоматизации. Он составляет львиную долю кода приложения, который приходится писать вручную.

Уровень данных описывает структуру данных приложения, используемые источники, форматы данных, технологии и механизмы доступа к данным. Для приложений .NET чаще всего используются возможности ADO.NET.

Уровень пользовательского интерфейса описывает возможности приложения по взаимодействию с пользователями, а также состав форм приложения, функциональность элементов управления (например, контроль ввода данных). Легкость автоматизации этого уровня зависит от того, насколько унифицированы пользовательские операции. Если удается создать типовые шаблоны элементов управления для основных операций, появляется возможность автоматической генерации форм и их содержимого при создании приложения из модели.

2.3 Этапы разработки

Процесс разработки разбивается на три этапа.

На первом этапе разрабатывается вычислительно-независимая модель (CIM). Часто модель, создаваемую на этом этапе, также называют доменной или бизнес-моделью. Цель данного этапа разработка общих требований к системе, создание общего словаря понятий, описание окружения, в котором система будет функционировать. Сущности, описываемые в модели CIM этого этапа, должны тщательно анализироваться и отрабатываться. Право на включение в модель должны иметь только те элементы, которые будут использованы и развиты на последующих этапах разработки. Для создания модели CIM на данном этапе можно использовать любые средства. Однако для совместимости с последующими этапами весьма желательно иметь описание модели на языке UML. Следует учитывать, что модель CIM первого этапа представляет собой скорее общую концепцию системы и не является насущно необходимой для процесса разработки приложения. При создании небольших программных систем этот этап можно опустить, однако при работе со сложными проектами он становится почти обязательным. К примеру, разработка текстового технического задания, казалось бы, никак не помогает собственно процессу программирования, зато, существенно способствует пониманию задачи в целом и позволяет избежать грубых ошибок проектирования в дальнейшем.

На втором этапе разрабатывается платформенно-независимая модель (PIM). Она может разрабатываться с нуля в случае отсутствия модели первого этапа или основываться на CIM. Преобразование CIM в PIM осуществляется на основе описания на зыке UML, созданного на первом этапе. Здесь в него добавляются элементы, описывающие бизнес-логику, общую структуру системы, состав и взаимодействие подсистем, распределение

функционала по элементам, общее описание и требования к пользовательскому интерфейсу. Модель PIM этого этапа обязательно включается во все автоматизированные среды разработки приложений на основе MDA.

На третьем этапе создаются платформенно-зависимые модели (PSM). Их число соответствует числу программных платформ, на которых будет функционировать приложение. Кроме этого, возможны случаи, когда приложение (или его составные части) должны работать на нескольких платформах одновременно. Модель PSM создается путем преобразования модели PIM с учетом требований модели платформы. Процесс преобразования описывается ниже. На этапе создания модели PSM разработка приложения согласно архитектуре MDA заканчивается. Считается, что правильно построенная PSM содержит техническую информацию, достаточную для генерации исходного кода (там, где это возможно) и необходимых ресурсов приложения. Здесь эстафетную палочку должна подхватить среда разработки, реализующая MDA. В нашем случае Delphi 9 обеспечивает генерацию кода и компиляцию приложений ECO. С формальной точки зрения это уже не относится к компетенции MDA, но для разработчика преобразование PSM в исполняемый код приложения непосредственное продолжение процесса разработки. Однако, архитектура MDA описывает еще один вариант прохождения третьего этапа, который называется прямым преобразованием в код. Спецификация MDA для этого случая сообщает, что могут существовать инструментарии, напрямую преобразующие модель PIM в исполняемый код приложения. Модель PSM при этом может создаваться как контрольное описание, позволяющее проверить результат прямого преобразования.

3 Преобразование моделей PIM PSM

Наиболее сложным и ответственным этапом при разработке приложений в рамках архитектуры MDA является преобразование модели PIM в модель PSM (Рис. 2). Именно на этом этапе общее описание системы на языке UML приобретает вид, пригодный для воплощения приложения на конкретной платформе. Как уже говорилось, в процессе принимает участие модель платформы. Преобразование моделей проходит три последовательные стадии:

1. Разработка схемы преобразования (mapping).

   Первоначально необходимо разработать схему преобразования элементов модели PIM в элементы модели PSM. Для каждой платформы создается собственная схема преобразования, которая напрямую зависит от возможностей платформы. Схема преобразования затрагивает как содержание модели (совокупность элементов и их свойства), так и саму модель (метамодель, используемые типы). В схеме преобразования нужным типам модели, свойствам метамодели, элементам модели PIM ставятся в соответствие типы модели, свойства метамодели, элементы модели PSM. При преобразовании моделей может использоваться несколько схем преобразования.

2. Маркирование (marking).

   Для связывания используются марки (mark) самостоятельные структуры данных, принадлежащие не моделям, а схемам преобразования и содержащие информацию о созданных связях. Наборы марок могут быть объединены в тематические шаблоны, которые возможно использовать в различных схемах преобразования. Процесс задания марок называется маркированием. В простейшем случае один элемент модели PIM соединяется маркой с одним элементом модели PSM. В более сложных случаях один элемент модели PIM может иметь несколько марок из разных схем преобразования. Что касается преобразования метамодели, то в большинстве случаев марки

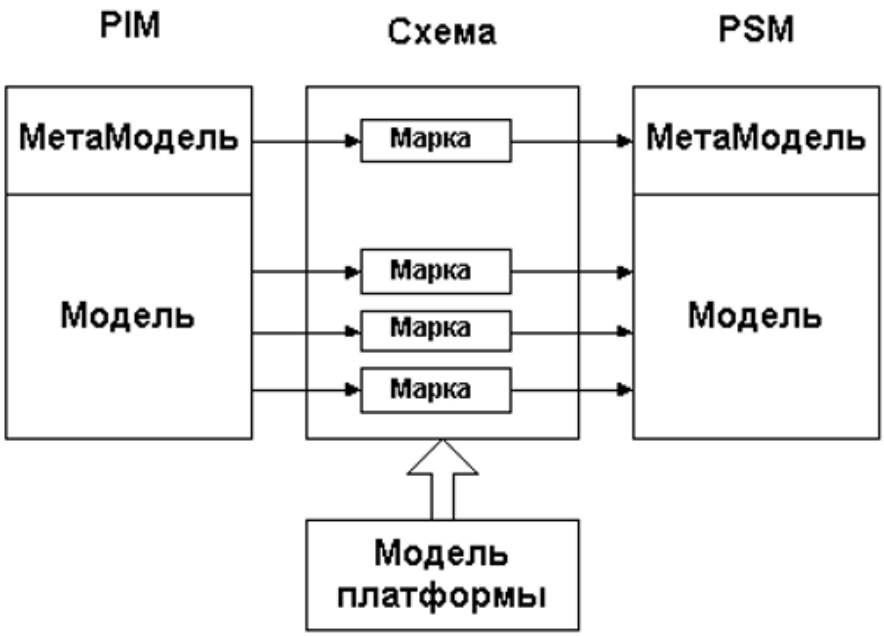

Рис. 2. Преобразование PIM модели в PSM

могут расставляться автоматически. А вот для элементов модели часто требуется вмешательство разработчика. В процессе маркирования необходимо использовать сведения о платформе. Эти сведения содержатся в модели платформы.

3. собственно преобразование (transformation).

   Процесс преобразования моделей заключается в переносе маркированных элементов модели и метамодели PIM в модель и метамодель PSM. Процесс преобразования должен документироваться в виде карты переноса элементов модели и метамодели. Способ преобразования моделей может быть:

   - ручной;
   - с использованием профилей;
   - с настроенной схемой преобразования;
   - автоматический.

Список литературы

[1] Лаврищева, Е. М. Программная инженерия и технологии программирования сложных систем : учебник для вузов / Е. М. Лаврищева. — 2-е изд., испр. и доп. — Москва : Издательство Юрайт, 2023. — 432 с. — (Высшее образование). — ISBN 978-5-534-07604-2. — Текст : электронный // Образовательная платформа Юрайт [сайт]. — URL: https://urait.ru/bcode/513067 (дата обращения: 07.03.2023).