Самоучитель

Александр Леоненков

UML 2

Объектно-ориентированный анализ и проектирование

Моделирование программных систем

Нотация и семантика последней версии UML 2

UML в Borland® Together® Designer

Язык объектных ограничений OCL

bhv

Александр Леоненков

Самоучитель UML 2

Санкт-Петербург

«БХВ-Петербург»

2007

УДК 681.3.06
ББК 32.973.26-018.1
Л47

Леоненков А. В.

Л47 Самоучитель UML 2. — СПб.: БХВ-Петербург, 2007. — 576 с.: ил.

ISBN 978-5-94157-878-8

Рассмотрена современная технология объектно-ориентированного анализа и проектирования программных систем и бизнес-процессов в контексте нотации унифицированного языка моделирования UML 2. Подробно изложены все понятия языка UML 2 в полном соответствии с оригинальной спецификацией последней версии этого языка. Приведены конкретные рекомендации по разработке канонических диаграмм языка и рассмотрены особенности разработки моделей с помощью CASE-средства Borland® Together® Designer. Описана нотация OCL — языка объектных ограничений, по которому практически отсутствует информация на русском.

Для системных и бизнес-аналитиков, архитекторов программ, руководителей проектов и информационных служб, корпоративных программистов и студентов

УДК 681.3.06
ББК 32.973.26-018.1

Группа подготовки издания:

Главный редактор	*Екатерина Кондукова*
Зам. главного редактора	*Игорь Шишигин*
Зав. редакцией	*Григорий Добин*
Редактор	*Анна Кузьмина*
Компьютерная верстка	*Натальи Караваевой*
Корректор	*Зинаида Дмитриева*
Дизайн серии	*Инны Тачиной*
Оформление обложки	*Елены Беляевой*
Зав. производством	*Николай Тверских*

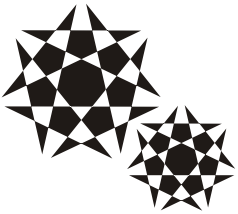
Лицензия ИД № 02429 от 24.07.00. Подписано в печать 27.12.06.
Формат 70×100 1/16. Печать офсетная. Усл. печ. л. 46,44.
Тираж 3000 экз. Заказ №
"БХВ-Петербург", 194354, Санкт-Петербург, ул. Есенина, 5Б.

Санитарно-эпидемиологическое заключение на продукцию
№ 77.99.02.953.Д.006421.11.04 от 11.11.2004 г. выдано Федеральной службой по надзору в сфере защиты прав потребителей и благополучия человека.

Отпечатано с готовых диапозитивов
в ГУП "Типография "Наука"
199034, Санкт-Петербург, 9 линия, 12

ISBN 978-5-94157-878-8

Оглавление



Глава 2

Основные элементы нотации языка UML 2.0

Язык UML 2.0 обладает весьма широкими возможностями, которые предназначены для решения разнообразных задач визуального моделирования из самых различных прикладных областей. Однако, обладая значительной избыточностью, не все его возможности моделирования являются обязательно необходимыми при разработке конкретных приложений. Соответствующая согласованность используемых конструкций языка UML 2.0 обеспечивается его модульным построением. Все это позволяет разработчикам выбирать только те элементы, которые представляют прямой интерес в контексте разрабатываемого проекта.

Язык UML 2.0 основан на некотором числе базовых понятий, которые могут быть изучены и применены большинством программистов и разработчиков, знакомых с методами объектно-ориентированного анализа и проектирования. При этом базовые понятия могут комбинироваться и расширяться таким образом, что специалисты объектного моделирования получают возможность самостоятельно разрабатывать модели больших и сложных систем в самых различных областях приложений.

2.1. Назначение языка UML 2.0

Язык UML 2.0, также как и его предыдущие версии, предназначен для решения следующих задач.

- Предоставить в распоряжение всех пользователей легко воспринимаемую и выразительную нотацию для визуального моделирования, специально предназначенную для разработки и документирования моделей сложных систем самого различного целевого назначения.
  - Речь идет о том, что важным фактором дальнейшего развития и повсеместного использования методологии ООАП является интуитивная

ясность и понятность основных конструкций соответствующего языка моделирования. Язык UML 2.0 включает в себя не только абстрактные конструкции, для представления метамоделей систем, но и целый ряд конкретных понятий, имеющих вполне определенную семантику. Это позволяет языку UML 2.0 одновременно достичь не только универсальности представления моделей для самых различных приложений, но и возможности описания достаточно тонких деталей реализации этих моделей применительно к конкретным системам.

- Для адекватного понимания базовых конструкций языка UML 2.0 важно не только владеть навыками объектно-ориентированного программирования, но и хорошо представлять себе общую проблематику процесса разработки моделей систем. Именно интеграция этих представлений образует новую парадигму в методологии ООАП, практическим следствием и центральным стержнем которой является язык UML 2.0.

❒ Снабдить исходные понятия языка UML 2.0 возможностью расширения и специализации для более точного представления моделей систем в конкретной предметной области.

- Хотя язык UML 2.0 является формальным языком спецификаций, формальность его описания отличается от синтаксиса как традиционных формально-логических языков, так и известных языков программирования. Разработчики из OMG предполагают, что язык UML 2.0 как никакой другой может быть приспособлен для конкретных предметных областей. Это становится возможным по той причине, что в самом описании языка UML 2.0 заложен механизм расширения базовых понятий, который является самостоятельным элементом и имеет собственное описание в форме правил расширения.
- В то же время разработчики из OMG считают крайне нежелательным переопределение базовых понятий языка. Это может привести к неоднозначной интерпретации их семантики и возможной путанице. Базовые понятия языка UML 2.0 не следует изменять больше, чем это необходимо для их расширения. Все пользователи должны быть способны строить модели систем для большинства обычных приложений с использованием только базовых конструкций языка UML 2.0 без применения механизма расширения. При этом новые понятия и графические обозначения целесообразно применять только в тех ситуациях, когда имеющихся базовых конструкций явно недостаточно для построения моделей системы.
- Язык UML 2.0 допускает также специализацию базовых понятий. Речь идет о том, что в конкретных приложениях пользователи должны уметь

дополнять имеющиеся базовые понятия новыми характеристиками или свойствами, которые не противоречат семантике этих понятий.

- Описание языка UML 2.0 должно поддерживать такую спецификацию моделей, которая не зависит от конкретных языков программирования и инструментальных средств проектирования программных систем.
  - Речь идет о том, что конструкции языка UML 2.0 не должны зависеть от особенностей их реализации на той или иной платформе или языке программирования. Другими словами, хотя отдельные понятия языка UML 2.0 семантически связаны с последними, их жесткая интерпретация в форме конструкций программирования не может быть признана корректной. Разработчики из OMG считают необходимым свойством языка UML 2.0 его контекстно-программную и платформенную независимость.
  - С другой стороны, язык UML 2.0 должен обладать потенциальной возможностью реализации своих конструкций на том или ином языке программирования. Конечно, в первую очередь имеются в виду языки, поддерживающие концепцию ООП, такие как Java, C++, C#, Object Pascal/Delphi. Именно это свойство языка UML 2.0 делает его современным и конструктивным средством решения задач моделирования сложных систем. В то же время, предполагается, что для программной поддержки конструкций языка UML 2.0 могут быть разработаны специальные инструментальные CASE-средства. Наличие последних имеет принципиальное значение для широкого распространения и использования языка UML 2.0.
- Описание языка UML 2.0 должно включать в себя семантический базис для понимания общих особенностей ООАП.
  - Говоря об этой особенности, имеют в виду самодостаточность языка UML 2.0 для понимания не только его базовых конструкций, но что не менее важно — понимания общих принципов ООАП. В этой связи необходимо отметить, что поскольку язык UML 2.0 не является языком программирования, а служит средством для решения задач объектно-ориентированного моделирования систем, описание языка UML 2.0 должно по возможности включать в себя все необходимые понятия методологии ООАП.
  - С другой стороны, какие бы то ни было ссылки на дополнительные источники, содержащие важную для понимания языка UML 2.0 информацию, по мнению разработчиков из OMG, должны быть исключены. Это позволит избежать неоднозначного толкования принципиальных особенностей методологии ООАП и их реализации в форме базовых конструкций языка UML 2.0.

- ❒ Поощрять развитие рынка программных инструментальных средств, поддерживающих методологию ООАП и концепцию MDA.
  - Более 800 ведущих производителей программных и аппаратных средств, усилия которых сосредоточены в рамках OMG, видят перспективы развития современных информационных технологий и основу своего коммерческого успеха в широком продвижении на рынок инструментальных средств, поддерживающих объектные технологии. С этой точки зрения языку UML 2.0 отводится роль базового концептуального средства для описания и документирования различных объектных технологий.
- ❒ Способствовать распространению объектных технологий и соответствующих понятий ООАП.
  - Если попытаться составить объективную картину возможностей языка UML 2.0, то можно прийти к следующему заключению. Следует признать, что усилия достаточно большой группы разработчиков, направленные на интеграцию в рамках языка UML 2.0 многих известных техник визуального моделирования, привели к серьезному усложнению языка UML 2.0 по сравнению с другими известными нотациями и его предыдущими версиями. Платой за сложность является действительно высокая гибкость и широкие изобразительные возможности языка UML 2.0. В свою очередь, использование языка UML 2.0 для решения всевозможных практических задач будет только способствовать его дальнейшему совершенствованию, а значит, и дальнейшему развитию объектных технологий и практики ООАП.
- ❒ Интегрировать в себя новейшие и наилучшие достижения практики ООАП.

  Язык UML непрерывно совершенствуется разработчиками, и основой этой работы является его дальнейшая интеграция с современными технологиями моделирования. При этом различные методы системного моделирования получают свое прикладное осмысление в контексте методологии ООАП. В последующем эти методы могут быть включены в состав очередных версий языка UML в форме дополнительных базовых понятий, наиболее адекватно и полно отражающие наилучшие достижения практики ООАП.

Чтобы решить указанные выше задачи, за свою историю язык UML претерпел определенную эволюцию. В результате описание самого языка UML 2.0 стало нетривиальным, поскольку семантика базовых понятий включает в себя целый ряд перекрестных связей с другими понятиями и конструкциями языка. В связи с этим последовательное рассмотрение основных конструкций

языка UML 2.0 стало практически невозможным, т. к. одни и те же понятия могут использоваться при построении различных диаграмм и представлений. В то же время каждое из представлений модели обладает собственными семантическими особенностями, которые накладывают отпечаток на семантику базовых понятий языка в целом.

Как результат, разработчикам нет необходимости знать весь язык UML 2.0, чтобы эффективно его использовать. Кроме того, большинство единиц языка может быть разделено на многократные приращения, каждое из которых добавляет к предыдущим дополнительные возможности моделирования. Такая дискретная декомпозиция языка UML 2.0 предназначена для того, чтобы сделать более легким его изучение и использование на практике. При этом отдельные приращения описываются в форме слияния пакетов, что находит свое выражение при рассмотрении отдельных типов канонических диаграмм.

Учитывая эти особенности, принятая в книге последовательность изучения языка UML 2.0 основывается на рассмотрении основных графических средств моделирования, а именно — канонических диаграмм. Все понятия, необходимые для построения соответствующих диаграмм, вводятся по мере необходимости. В данной главе приводится описание общих особенностей языка UML 2.0, которые в той или иной степени оказывают влияние на понимание всех его базовых конструкций и канонических диаграмм.

2.2. Общая структура языка UML 2.0

С самой общей точки зрения описание языка UML 2.0 состоит из двух взаимодействующих частей: семантики и нотации.

Семантика (semantics) — система правил и соглашений, определяющих толкование и придание смысла конструкциям некоторого языка.

Нотация (notation) — система условных обозначений, принятая в некотором языке для изображения и визуализации модели.

В рамках языка UML 2.0 семантика и нотация тесно взаимосвязаны. Семантика языка UML 2.0 описывается с использованием некоторого подмножества нотации UML 2.0. В свою очередь, нотация UML 2.0 описывает соответствие или отображение графической нотации в базовые понятия семантики. Таким образом, с функциональной точки зрения эти две части дополняют друг друга. При этом семантика языка UML 2.0 описывается на основе некоторой метамодели, имеющей три отдельных представления: абстрактный

синтаксис, правила корректного построения выражений и семантику. Рассмотрение семантики языка UML 2.0 предполагает некоторый "полуформальный" стиль изложения, который объединяет естественный и формальный языки для представления базовых понятий и правил их расширения.

Семантика определяется для двух категорий объектных моделей: структурных моделей (статических моделей) и моделей поведения (динамических моделей).

Структурные модели (structured models) — модели, предназначенные для описания статической структуры сущностей или элементов некоторой системы, включая их классы, интерфейсы, атрибуты и отношения.

Модели поведения (behavioral models) — модели, предназначенные для описания процесса функционирования элементов системы, включая их методы и взаимодействие между ними, а также процесс изменения состояний отдельных элементов и системы в целом.

Для решения самого широкого диапазона задач моделирования в языке UML 2.0 разработана достаточно полная семантика для всех элементов графической нотации. При этом требования семантики языка UML 2.0 конкретизируются при построении отдельных типов диаграмм, последовательное рассмотрение которых служит темой *части II* книги. Нотация языка UML 2.0 включает в себя описание отдельных семантических элементов, которые могут применяться при построении диаграмм.

Формальное описание самого языка UML 2.0 основывается на некоторой общей иерархической структуре модельных представлений, состоящей из четырех уровней:

- мета-метамодель (M3);
- метамодель (M2);
- модель (M1);
- объекты или экземпляры (M0).

Уровень *мета-метамодели* образует исходную формально-логическую основу для всех возможных метамодельных представлений. Главное предназначение этого уровня состоит в том, чтобы определить язык для спецификации метамодели. Мета-метамодель определяет модель языка UML 2.0 на самом высоком уровне абстракции и является наиболее компактным ее описанием. С другой стороны, мета-метамодель может специфицировать несколько метамоделей, чем достигается потенциальная гибкость включения дополни-

тельных понятий. Хотя в книге этот уровень не рассматривается, он наиболее тесно связан с теорией формальных языков. Примерами понятий этого уровня служат метакласс, метаатрибут, метаоперация.

Метамодель является экземпляром или конкретизацией мета-метамодели. Главная задача этого уровня — определить язык для спецификации моделей. Данный уровень является более конструктивным, чем предыдущий, поскольку обладает более развитой семантикой базовых понятий. Все основные понятия языка UML 2.0 — это понятия уровня метамодели. Примерами таких понятий являются класс, атрибут, операция, компонент, ассоциация и многие другие. Рассмотрению семантики и графической нотации понятий уровня метамодели собственно и посвящена настоящая книга.

Модель в контексте языка UML 2.0 является экземпляром метамодели в том смысле, что любая конкретная модель системы должна использовать только понятия метамодели, конкретизировав их применительно к данной ситуации. Это уровень для описания информации о конкретной предметной области. Однако поскольку при построении модели используются понятия языка UML 2.0, то необходима полная согласованность понятий уровня модели с базовыми понятиями языка UML 2.0 уровня метамодели. Примерами понятий уровня модели могут служить, например, имена полей проектируемой базы данных, такие как: имя и фамилия сотрудника, возраст, должность, адрес, телефон. При этом данные понятия используются лишь как имена соответствующих информационных атрибутов.

Конкретизация понятий модели происходит на уровне *объектов* или *экземпляров*. При этом совокупность объектов рассматривается как отдельный экземпляр модели времени выполнения, поскольку содержит конкретную информацию относительно того, чему в действительности соответствуют те или иные понятия модели. Примером объекта может служить следующая запись в проектируемой базе данных: "Игорь Греков, 40 лет, модельер, ул. Системная, 1-2, 123-4567". Рассмотренную 4-уровневую структуру модельных представлений языка UML 2.0 можно проиллюстрировать следующим образом (рис. 2.1).

Описание семантики языка UML 2.0 предполагает рассмотрение базовых понятий только уровня *метамодели (M2)*, который представляет собой пример или частный случай уровня мета-метамодели. Метамодель UML 2.0 является по своей сути скорее логической моделью, чем физической или моделью реализации. Особенность логической модели заключается в том, что она концентрирует внимание на декларативной или концептуальной семантике, опуская детали конкретной физической реализации моделей. При этом отдельные реализации, использующие данную логическую метамодель, должны быть согласованы с ее семантикой, а также поддерживать возможности импорта и экспорта отдельных логических моделей.

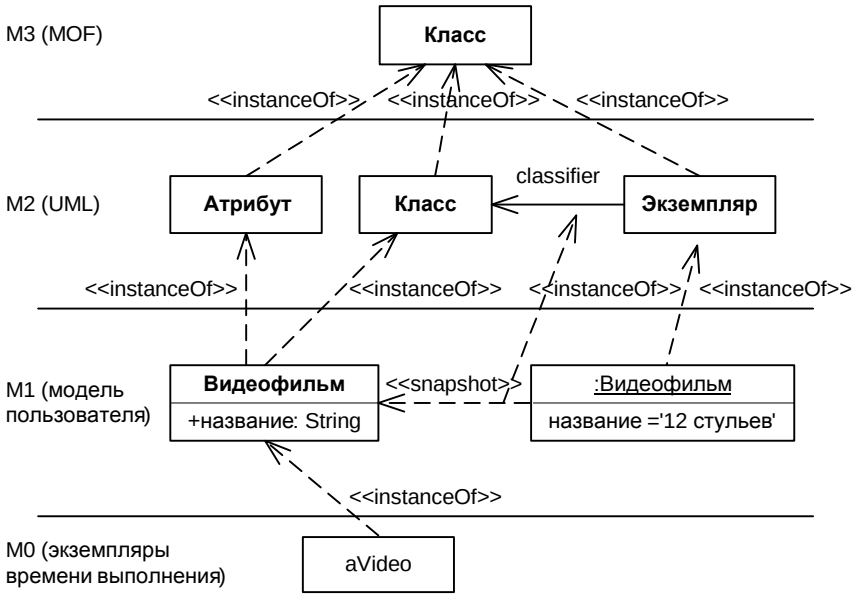


Рис. 2.1. Иллюстрация 4-уровневой структуры модельных представлений языка UML 2.0

В то же время, логическая метамодель может быть реализована различными способами для обеспечения требуемого уровня производительности и надежности соответствующих инструментальных средств. В этом заключается недостаток логической модели, которая не содержит на уровне семантики требований, обязательных для ее эффективной последующей реализации. Однако согласованность метамодели с конкретными моделями реализации является обязательной для всех разработчиков программных средств, обеспечивающих поддержку языка UML 2.0.

Метамодель языка UML 2.0 имеет довольно сложную структуру, которая включает в себя метаклассы, метаассоциации и стереотипы, число которых возрастает с появлением новых версий языка. Чтобы справиться с этой сложностью языка UML 2.0, все его элементы организованы специальным образом в логически связанные подмножества, получившие название пакетов. Поэтому рассмотрение языка UML 2.0 на метамодельном уровне заключается в описании его наиболее общих логических блоков или *пакетов*.

Говоря о пакетах в контексте общего описания языка, мы, по сути дела, приступаем к рассмотрению графической нотации языка UML 2.0. При этом следует заметить, что для описания языка UML 2.0 используются средства самого этого языка. Одним из таких средств является *пакет*, который в общем случае служит для группировки любых элементов модели. Пакеты могут находиться в различных отношениях и быть вложенными друг в друга.

2.3. Пакеты в языке UML 2.0

Пакет — основной способ организации элементов метамодели языка UML 2.0. Каждый пакет владеет всеми своими элементами, т. е. теми элементами, которые включены в него. Про соответствующие элементы пакета говорят, что они принадлежат пакету или входят в него. В языке UML 2.0 для визуализации пакетов разработана специальная символика или графическая нотация, которой мы и будем пользоваться в дальнейшем. Именно с описания этой системы обозначений мы приступим к изучению основных элементов языка.

Для графического изображения пакетов на диаграммах применяется специальный графический символ — большой прямоугольник с небольшим прямоугольником, присоединенным к левой части верхней стороны первого (рис. 2.2). Визуально символ пакета напоминает пиктограмму папки на экране монитора. Внутри большого прямоугольника могут изображаться элементы, которые принадлежат этому пакету. Если такие элементы не указаны, то внутри большого прямоугольника записывается имя пакета, которое должно быть уникальным в пределах рассматриваемой модели (рис. 2.2, *а*). Если же элементы пакета указаны, то имя пакета записывается в верхнем маленьком прямоугольнике (рис. 2.2, *б*).

Рис. 2.2. Графическое изображение пакетов в языке UML 2.0

Говоря об имени пакета, следует остановиться на общем соглашении об именах в языке UML 2.0. В данном случае именем пакета может быть строка текста, содержащая произвольное число букв, цифр и некоторых специальных знаков. С целью удобства спецификации пакетов принято в качестве их имен использовать существительные, возможно, с пояснительными словами, например, контроллер, графический интерфейс, форма ввода данных.

Сами по себе пакеты находят ограниченное применение, поскольку содержат лишь информацию о входящих в их состав элементах модели. Не менее важно представить графически отношения, которые могут иметь место между отдельными пакетами. Как и в теории графов, для визуализации отношений

в языке UML 2.0 применяются отрезки линий, внешний вид которых имеет смысловое содержание или семантику.

Одним из типов отношений между пакетами является отношение вложенности или включения пакетов друг в друга. С одной стороны, в языке UML 2.0 это отношение может быть изображено без использования линий простым размещением одного пакета-прямоугольника внутри другого пакета-прямоугольника (рис. 2.3). Так, в данном случае `Пакет A` содержит в себе два подпакета: `Пакет B` и `Пакет C`.

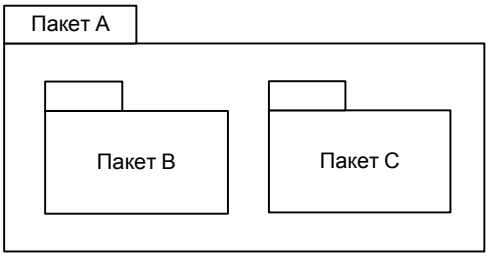


Рис. 2.3. Графическое изображение вложенности пакетов

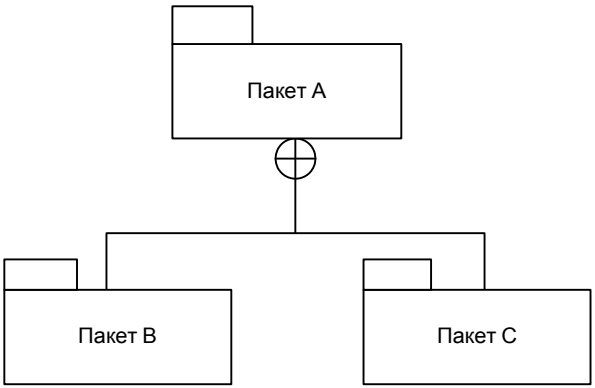


Рис. 2.4. Альтернативное графическое изображение вложенности пакетов

С другой стороны, это же отношение может быть изображено с помощью отрезков линий аналогично графическому представлению дерева. В этом случае пакет, содержащий другие пакеты, или пакет-контейнер изображается в верхней части рисунка, а его подпакеты — уровнем ниже. Пакет-контейнер соединяется с подпакетами сплошной линией, на конце которой, примыкающей к пакету контейнеру, изображается специальный символ — ⊕. Этот сим-

вол означает, что подпакеты являются "собственностью" или частью контейнера, и, кроме этих подпакетов, контейнер не содержит никаких других подпакетов. Рассмотренный выше пример (см. рис. 2.3) может быть представлен альтернативно следующим образом (рис. 2.4).

На графических диаграммах между пакетами могут указываться и другие типы отношений, которые будут описаны при рассмотрении диаграммы пакетов *(см. главу 6)*.

2.4. Основные пакеты метамодели языка UML 2.0

Основой представления языка UML 2.0 на метамодельном уровне является спецификация его основных логических частей в форме пакетов: `Элементы ядра`, `Абстракции`, `Основы`, `Конструкции` и `Простейшие Типы` (рис. 2.5).

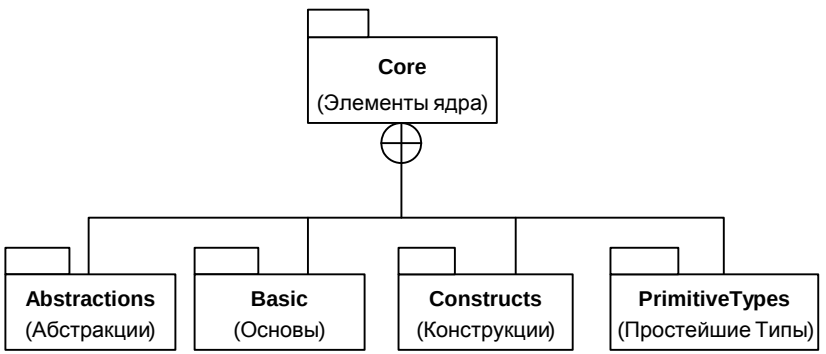


Рис. 2.5. Основные пакеты метамодели языка UML 2.0

Эти пакеты, в свою очередь, также включают в себя другие подпакеты, которые содержат описание фундаментальных понятий языка UML 2.0. Далее дается краткая характеристика элементов каждого из перечисленных подпакетов, входящих в состав пакета `Элементы ядра`. Более полное рассмотрение отдельных пакетов метамодели будет представлено в главах, посвященных изучению отдельных видов канонических диаграмм. Последние аккумулируют в себе не только различные представления моделируемой системы, но и более детально раскрывают семантические особенности применения базовых конструкций языка UML 2.0 в процессе построения конкретных моделей.

2.4.1. Пакет *Абстракции*

Пакет Абстракции является наиболее фундаментальным из всех подпакетов, которые входят в пакет Элементы ядра языка UML 2.0. Он определяет основные абстрактные элементы, необходимые для разработки объектных моделей. При этом абстрактные элементы метамодели не имеют экземпляров или примеров и используются исключительно для уточнения других элементов модели.

В этот пакет входят основные подпакеты языка UML 2.0: Elements (Элементы), Ownerships (Права собственности), Namespaces (Пространства имен), Comments (Комментарии), Relationships (Отношения), Visibilities (Видимости), Expressions (Выражения), StructuralFeatures (Структурные характеристики), BehavioralFeatures (Характеристики поведения), Classifiers (Классификаторы), Constraints (Ограничения), Generalizations (Обобщения), Instances (Экземпляры), MultiplicityExpressions (Выражения кратности), Redefinitions (Переопределения), Changeabilities (Способности к изменениям), Super (Верхний уровень), Literals (Литералы), Multiplicities (Кратности), TypedElements (Типизированные элементы).

2.4.2. Пакет *Основы*

Пакет Основы содержит минимальное подмножество элементов языка UML 2.0, которые называются *метаклассами* и используются для дальнейшей конкретизации при моделировании структуры систем. Метаклассы данного пакета и отношения между ними описывают следующие диаграммы:

- диаграмма типов, которая специфицирует метаклассы, имеющие отношение к именованным и типизированным элементам: Type (Тип), Element (Элемент), TypedElement (Типизированный элемент), NamedElement (Именованный элемент), Comment (Комментарий);
- диаграмма классов, которая определяет конструкции, необходимые для моделирования классов: Class (Класс), Property (Свойство), Operation (Операция), Parameter (Параметр), MultiplicityElement (Элемент кратности);
- диаграмма типов данных, которая специфицирует метаклассы, которые определяют типы данных: Type (Тип), PrimitiveType (Простейший Тип), Enumeration (Перечисление), DataType (Тип Данных);
- диаграмма пакетов, которая определяет базовые конструкции, имеющие отношение к пакетам и их содержимому: Package (Пакет) и Type (Тип).