CASE-средства фирмы Rational Software

- CASE RATIONAL ROSE поддерживает объектный анализ и проектирование сложных программных систем.

Диаграммы деятельности (Activity diagram)

- Диаграммы деятельности (Activity diagram) используются для описания последовательности различных видов деятельности субъектов или объектов, а также могут быть использованы и для описания их состояний.
- Событием называется любое происшествие, которое может быть причиной изменения состояния субъекта или объекта, или перехода от одного вида деятельности к другому виду. События могут вызывать некоторые действия.
- Действием называется операция, которая с практической точки зрения, требует нулевого времени на выполнение, например включение сигнала тревоги.
  Переход может происходить и по условию.
- Условие есть логическое выражение, включающее некоторые величины. Переход в последующее состояние или к следующему виду деятельности допускается только в случае истинности этого выражения.
- События, действия, условия можно добавить к переходу, используя для его описания спецификацию.

Пример элемента переход (state transition) между элементами деятельность и состояние

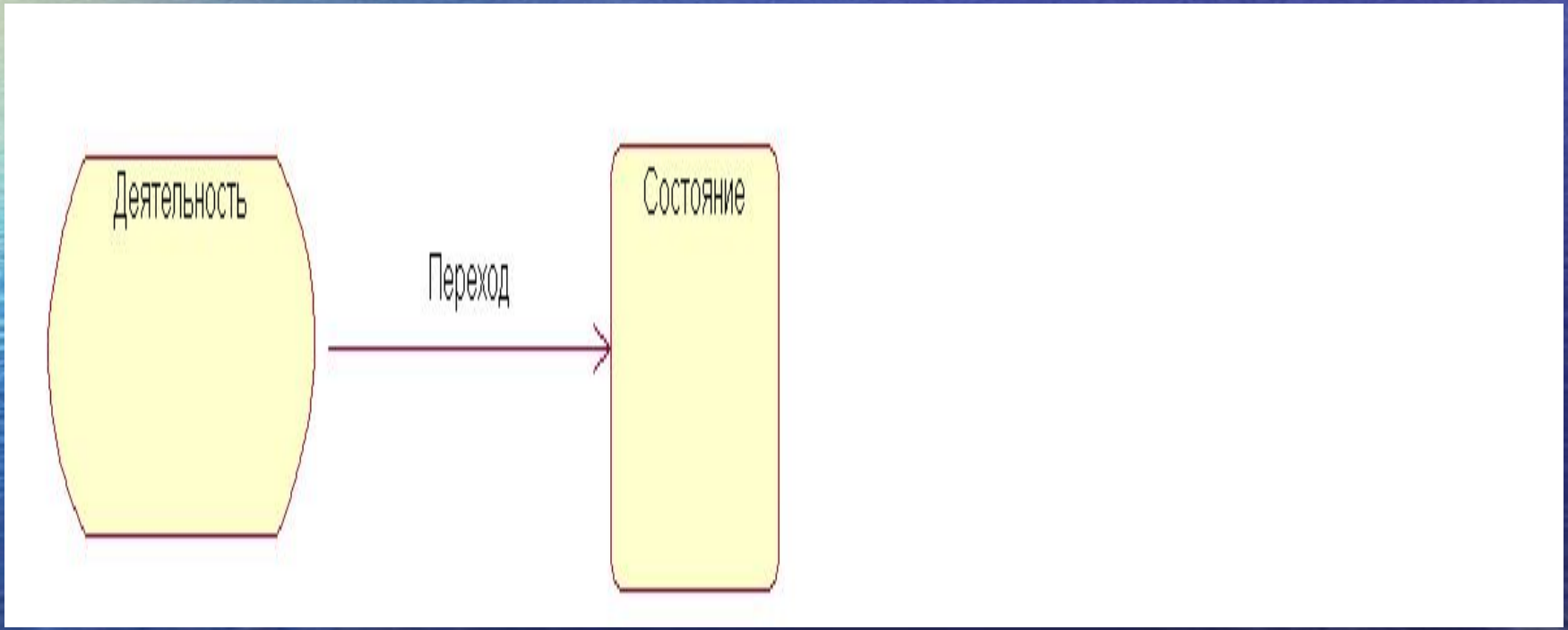

Диаграмма Use Case

- Разработка моделей функций программных систем основывается на использовании диаграмм Use Case.
- При моделировании функций системы Use Case диаграмма описывает собственно функции моделируемой системы, внешние по отношению к моделируемой системе субъекты и объекты, взаимодействующие с системой и не являющейся ее частью, и связи между этими субъектами, объектами и функциями.
- Модель функций системы определяется как иерархия диаграмм.
- Первый уровень иерархии - Use Case System может включать следующие компоненты: систему или подсистемы (Use Case Packages).
- Последующие уровни:
- систему или подсистемы (Use Case Packages) или функции (Use Case);
- субъектов и объектов, взаимодействующих с системой и не являющейся ее частью,а именно актеров (Actors),
- связи между компонентами диаграммы (Relationships).
- Диаграммы, декомпозирующие подсистемы, могут включать подсистемы или функции, субъектов и объектов, взаимодействующих с системой, и связи между ними.
- Диаграммы, декомпозирующие функции, могут включать функции, субъектов и объектов, взаимодействующих с системой, и связи между ними.
- На изображении системы или подсистемы указывается ее наименование.

Диаграмма Use Case

- Функции (Use Case) изображаются на диаграммах Use Case как овал. Под овалом указывается имя функции. Имя функции может включать неформальное описание последовательности действий. Рекомендуется именовать функции с использованием глаголов.
- Актером на диаграммах Use Case является субъект или объект, взаимодействующий с системой.
- Актер может, например, вводить информацию в систему, получать информацию из системы, производить прочие взаимодействия с системой.
- Имя актера есть роль, которую он выполняет в системе, например, менеджер, продавец, система учета, администратор системы.
- Связи на диаграммах Use Case имеют место:
- Между подсистемами.
- . Между актером и функцией.
- . Между функциями.
- Между подсистемами имеет место связь, которая есть зависимость Связь обозначается прерывистой линией со стрелкой. Связь должна проводится от зависимой подсистемы к независимой подсистеме.
- Между актером и функцией устанавливается связь, которая называется ассоциацией. Связь показывает взаимодействие между актером и функцией.

Диаграмма Use Case

- Связь может быть двунаправленной. Связь обозначается сплошной линией со стрелкой или без нее.
- На диаграммах могут используются несколько типов связей между функциями, например, uses (использует), extends (расширяет).
- Другими словами, некоторые функции в системе могут использовать другие функции. Некоторые функции могут выполняться при наступлении определенных условий или быть опциональными. В первом случае используются связь uses, во втором случае - extends.
- Связи uses и extends обозначают линией со стрелкой в виде не закрашенного треугольника. Для связи uses стрелка направлена к функции, которую используют.
- Для связи extends стрелка направлена к функции, которая включает функцию, используемую опционально или по наступлении определенного условия. Наименование связи uses и extends заключаются в двойные скобки <<>>.

Диаграмма классов

- В объектно-ориентированной методологии анализа и проектирования систем строительными блоками являются классы и объекты. С точки зрения восприятия человеком объектом может быть:
- осязаемый или видимый предмет;
- нечто, воспринимаемое мышлением;
- нечто, на что направлена мысль или действие.
- Объект моделирует часть окружающей действительности и таким образом существует во времени и пространстве. Объект представляет собой конкретный опознаваемый предмет, единицу или сущность (реальную или абстрактную), имеющую четко определенное функциональное назначение в данной предметной области.
- Можно дать еще одно определение объекта: Объект обладает состоянием, поведением и идентичностью. Структура и поведение схожих объектов определяет общий для них класс. Термины экземпляр класса и объект идентичны.
- Состояние объекта определяется набором свойств или атрибутами и связями, которые может иметь объект с другими объектами. Является типичным изменение состояния объекта во времени.
- Объекты не существуют изолированно, а подвергаются воздействию или сами воздействуют на другие объекты. Определенное воздействие одного объекта на другой с целью вызвать соответствующую реакцию называется операцией. В объектно-ориентированных языках операции, выполняемые над объектом,

Диаграмма классов

- Поведение - это то, как объект действует и реагирует. Поведение выражается в терминах состояния объекта и передачи сообщения.
- Идентичность это такое свойство объекта, которое отличает его от всех других объектов.
- Понятия объекта и класса настолько тесно связаны, что невозможно говорить об объекте безотносительно к его классу. Однако существует важное различие этих двух понятий. В то время как объект обозначает конкретную сущность, определенную во времени и пространстве, класс определяет абстракцию существенного в объекте.
- Класс есть описание группы объектов, обладающих общими свойствами (атрибутами), общим поведением, общими связями с другими объектами и общей семантикой.
- Класс является шаблоном для создания объекта. Каждый объект является экземпляром некоторого класса. Принято в языке UML обозначать класс прямоугольником, состоящим из трех частей. В верхней части прямоугольника указывается название класса, в средней - атрибуты, в нижней - операции.

Диаграмма классов

- Для описания классов в Rational Rose, как и для описания взаимодействия между функциями на диаграммах Use Case используются стереотипы.Стереотипы позволяют конструировать новые виды классов.
- Класс используется для отражения взаимодействия системы и ее окружения или взаимодействия внутри системы, а также для описания алгоритмов функционирования системы. Хорошей практикой является создание для управляющих классов диаграмм состояний и переходов.
- Классы и объекты не существуют изолированно, они могут взаимодействовать друг с другом. Существуют следующие основные виды отношений или связей между классами:
- ассоциация;
- агрегация;
- наследование.
- Ассоциация есть смысловая связь. Связь является двунаправленной. Связь не объясняет, как классы общаются друг с другом, отмечается только смысловая зависимость между классами.
- В UML эта связь изображается сплошной прямой линией. Ассоциация может быть поименована. Имя ассоциации обозначается глаголом. Рекомендуется указывать имя ассоциации так, чтобы оно читалось корректно слева направо или сверху вниз.
- Состояние и поведение класса определяет исполняемые им роли. Роли могут размещаться как на одном конце связи, так и на обоих концах. Имя роли помещается на связи рядом с классом. Роль может быть использована вместо имени связи. Если требуется, указывается и наименование связи и роли.

Диаграмма классов

- Количество объектов класса, принимающих участие в связи называется мощностью связи. Мощность указывается на каждом конце ассоциативной связи. Мощность означает число связей между одним объектом в начале линии связи с объектами в конце линии связи. Мощность может обозначаться следующим образом:
- 1 - точно один объект;
- 0...* - ноль или больше объектов;
- 1..*- один или больше объектов;
- 0..1 - ноль или один объект;
- 5..8 - специфический диапазон 5,6,7,8;
- 4..7,9 - комбинация 4,5,6,7, или 9 объектов.
- Класс может иметь ассоциативную связь с самим собой. Такая ассоциативная связь называется рефлексивной.
- Агрегация (Aggregation Relationships) Агрегация обозначает связь часть целого (part of).Например, самолет состоит из крыльев, двигателей, шасси и прочих частей.
- В UML эта связь изображается сплошной прямой линией с добавлением на конце ромба. Ромб указывает на целое. Агрегация есть частный случай ассоциации.
- Наследование (Generalize/Inherits Relationship) Наследование это такое отношение между классами, когда один класс повторяет структуру и поведение другого класса – одиночное наследование или других классов - множественное наследование.

Диаграмма классов

- Класс, структура и поведение которого наследуется, называется суперклассом. Подкласс обычно расширяет или ограничивает существующую структуру и поведения класса. Наследование это связь "is a". Так как наследование не является связью между разными объектами, она может не именоваться, на ней не указываются роли и мощность.
- Не существует ограничения в числе уровней наследования. Однако имеется мнение ограничивать число уровней наследования тремя - пятью.
- В UML наследование изображается стрелкой с не закрашенным треугольником, обращенным к суперклассу.
- В случае, когда много подклассов наследуют свойства суперкласса для обозначения иерархии удобно использовать дерево иерархии. Диаграмма классов используется на стадии анализа чтобы отображать понятия (роли и обязанности сущностей) изучаемой предметной области.
- Диаграмма классов также может использоваться на стадии проектирования - чтобы передать структуру классов, формирующих архитектуру системы.
- Главная диаграмма классов Main Class Diagram есть совокупность пакетов классов.
- Каждый пакет имеет свою главную диаграмму классов Package Main Class Diagram, состоящую из пакетов и классов.

Диаграмма классов

- Создаются и другие диаграммы классов, например, детализирующие структуру и поведение одного или более классов в подсистеме, отражающие иерархию наследования.
- Моделирование Главной диаграммы классов Main Class Diagram с использованием Rational Rose автоматически создает главную диаграмму классов и помещает ее в раздел логического проектирования модели.
- Для того чтобы добавить пакеты в главную диаграмму классов:
- Щелкните по пакету на панели инструментов;
- Щелкните по полю рисования диаграммы в месте, где требуется разместить пакет.
- Повторяйте шаги 1, 2 пока все пакеты не будут размещены на поле диаграммы.
- Зависимый пакет называется клиентом, независимый - сервером. Зависимость пакета А от пакета В означает, что один или более классов в пакете А инициирует взаимодействие с одним или более классами в пакете В.
- Элементы главной диаграммы пакета классов формируются следующим образом.
- Дважды щелкните по пакету на диаграмме классов;
- Rose откроет пакет и создаст главную диаграмму для пакета;
- Щелкните по классу на панели инструментов;
- Щелкните по полю рисования диаграммы в месте, где требуется разместить класс. Повторяйте шаги 3, 4, пока все классы не будут размещены на поле диаграммы;
- Щелкните по связи на панели инструментов;
- Соедините классы связями.
- Повторяйте шаги 5, 6, пока все связи не будут построены на поле диаграммы.

Диаграмма классов

- Диаграмма классов может быть создана и в разделе моделирования диаграмм Use Case. Эта диаграмма обычно подсоединяется к функции (use case) и отражает классы, участвующие в реализации этой функции.
- Пока диаграмма выбрана, введите имя диаграммы;
- Дважды щелкните по диаграмме в окне просмотра Use Case, для того чтобы открыть поле диаграммы;
- Щелкните по классу на панели инструментов или в разделе модели Logical View;
- Щелкните по полю рисования диаграммы в месте, где требуется разместить класс. Повторяйте шаги 5,6, пока все классы не будут размещены на поле диаграммы;
- Щелкните по связи на панели инструментов;
- . Соедините классы связями.
- Повторяйте шаги 7,8, пока все связи не будут построены на поле диаграммы.
- Для документирования классов используется спецификация.
- Спецификация класса используются для неграфического описания элемента изображения. Особо следует подчеркнуть, что спецификация должна отражать то, что не выражено в графических элементах диаграммы.
- Технология документирования диаграммы классов с использованием текстовых документов аналогична технологии документирования текстовых документов на диаграммах Use Case. Заполнение спецификации атрибутов и операций классов должно учитывать язык реализации.

Диаграммы последовательностей и взаимодействия

- Диаграммы последовательностей и взаимодействия используются для описания сценария взаимодействия объектов реализующих конкретную функцию, определенную на диаграмме Use Case.
- Диаграммы последовательностей используются для описания сценария взаимодействия объектов во времени.
- В UML объект на диаграмме последовательностей обозначается прямоугольником. Внутри прямоугольника указывается имя объекта, которое подчеркивается. Объекты располагаются на диаграмме горизонтально. Под каждым объектом располагается вертикальная пунктирная линия. Сообщение между объектами обозначаются сплошной линией со стрелкой. Линия проводится от объекта, который посылает сообщение, к объекту, который получает сообщение.
- Сообщение это или какое-либо действие или функции на языке реализации.
- Сложные сценарии могут быть дополнены пояснениями. Пояснения могут быть подписаны к любому сообщению слева от диаграммы на соответствующем уровне простым текстом, с элементами структуризации, или с использованием языка реализации.

Диаграмма компонент

- Диаграмма компонент отражает физическую структуру модели. Диаграмма компонент отражает организацию и связи среди компонент программного обеспечения, таких например, как исходные тексты программ, объектные модули, исполняемые модули, библиотеки динамической компоновки.
- Диаграмма компонент включает.
- Подсистемы компонент.
- Собственно компоненты.
- Интерфейс.
- Связи между компонентами.
- Большие системы могут быть разложены на несколько сотен, даже тысячи модулей. Пытаться разобраться в физической структуре такой системы без ее дополнительного структурирования практически невозможно.
- На практике разработчики всегда стремятся следовать неформальному соглашению: собирать связанные между собой модули в структуры типа каталогов. По этим соображениям и введено понятие подсистемы среди модулей или компонент.
- Подсистемы представляют собой совокупности логически связанных модулей или компонент.
- Подсистемы модулей обозначаются в UML аналогично подсистемам классов и функций с использованием пакета. В качестве имени подсистемы используется имя директории, в которой хранятся компоненты.
- Подсистемы модулей могут иметь между собой связи.
- Компонентами являются исходные тексты программ, объектные модули, исполняемые модули, библиотеки динамической компоновки.
- Между компонентами или модулями может существовать связь. Связь которая, которая

Диаграммы размещения

- Диаграммы размещения используются, для отражения конфигурации оборудования и программного обеспечения в реально действующей системе. Основные элементы диаграммы:
- процессоры;
- устройства;
- соединения.
- Процессор (иначе компьютер) - часть аппаратуры, способная выполнять программы. Устройство это часть оборудования, на котором программы не выполняются. На диаграммах каждый компьютер и устройство должны иметь свое имя. Никаких ограничений на имена процессоров и устройств нет, так как они обозначают "железо", а не программы. Можно дополнить значок процессора или компьютера списком процессов или программ, выполняющихся на нем. Соединения на диаграмме изображается линией.