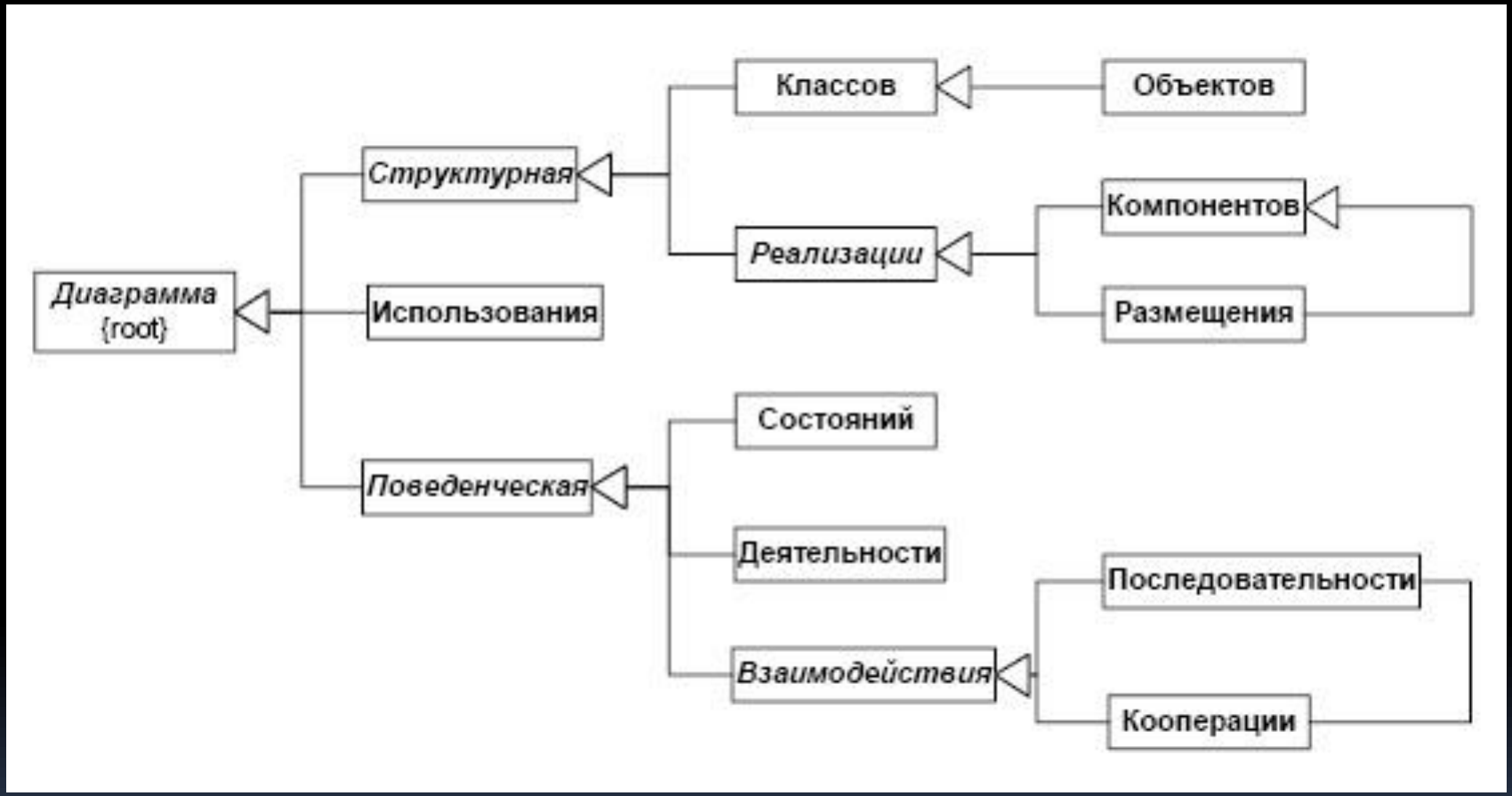
Анализ и проектирование на UML

Максим Валерьевич Хлопотов

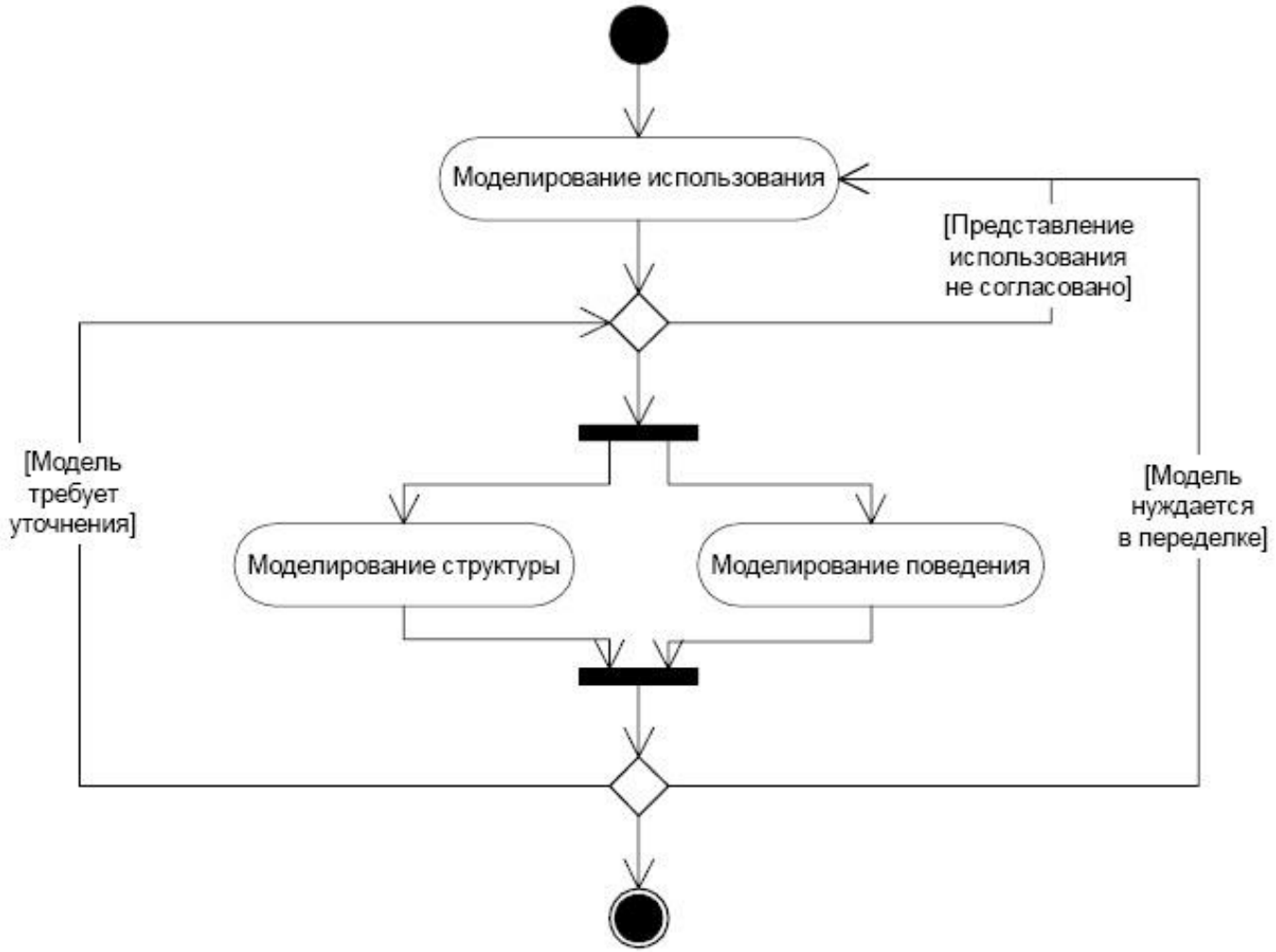
Темы лекционных занятий

1. Введение в UML
2. **Моделирование использования**
3. Моделирование структуры
4. Моделирование поведения
5. Дисциплина моделирования

Иерархия диаграмм UML



Процесс моделирования



Представления

Все аспекты моделируемой системы не удается описать с единой точки зрения.
Моделировать сложную систему следует с нескольких различных точек зрения, каждый раз принимая во внимание один аспект моделируемой системы и абстрагируясь от остальных.
Этот тезис является одним из основополагающих принципов UML.

Представления

Выделим три представления:
- представление использования (что делает система полезного?);
- представление структуры (из чего состоит система?);
- представление поведения (как работает система?).

Представления

Представление использования призвано отвечать на вопрос, что делает система полезного.

Определяющим признаком для отнесения элементов модели к представлению использования является, по нашему мнению, явное сосредоточение внимание на факте наличия у системы внешних границ, то есть выделение внешних действующих лиц, взаимодействующих с системой, и внутренних вариантов использования, описывающих различные сценарии такого взаимодействия.

Описывается диаграммой использования и может быть дополнена диаграммой деятельности (без использования объектов).

Диаграмма использования

Диаграмма использования является основным средством моделирования использования в UML.

На диаграмме использования применяются следующие типы сущностей:

- действующие лица;
- варианты использования;
- примечания;
- пакеты.

Диаграмма использования

Между этими сущностями устанавливаются следующие типы отношений:

- ассоциация между действующим лицом и вариантом использования;
- обобщение между действующими лицами;
- обобщение между вариантами использования;
- зависимости (двух стереотипов) между вариантами использования.

Моделирование использования

Наш язык и мышление устроены так, что самой простой, понятной и четкой формой изложения мыслей являются так называемые простые утверждения.

Простое утверждение имеет следующую грамматическую форму: подлежащее — сказуемое — прямое дополнение. В логических терминах: субъект — предикат — объект.

Например: начальник увольняет сотрудника, директор создает отдел.

Моделирование использования

По сути, именно простые утверждения и записаны на диаграмме использования.

Действующее лицо — это субъект, а вариант использования — предикат (вместе с объектом).

Моделирование использования предполагает явное формулирование требований к системе на самом начальном этапе разработки.

Диаграмма использования

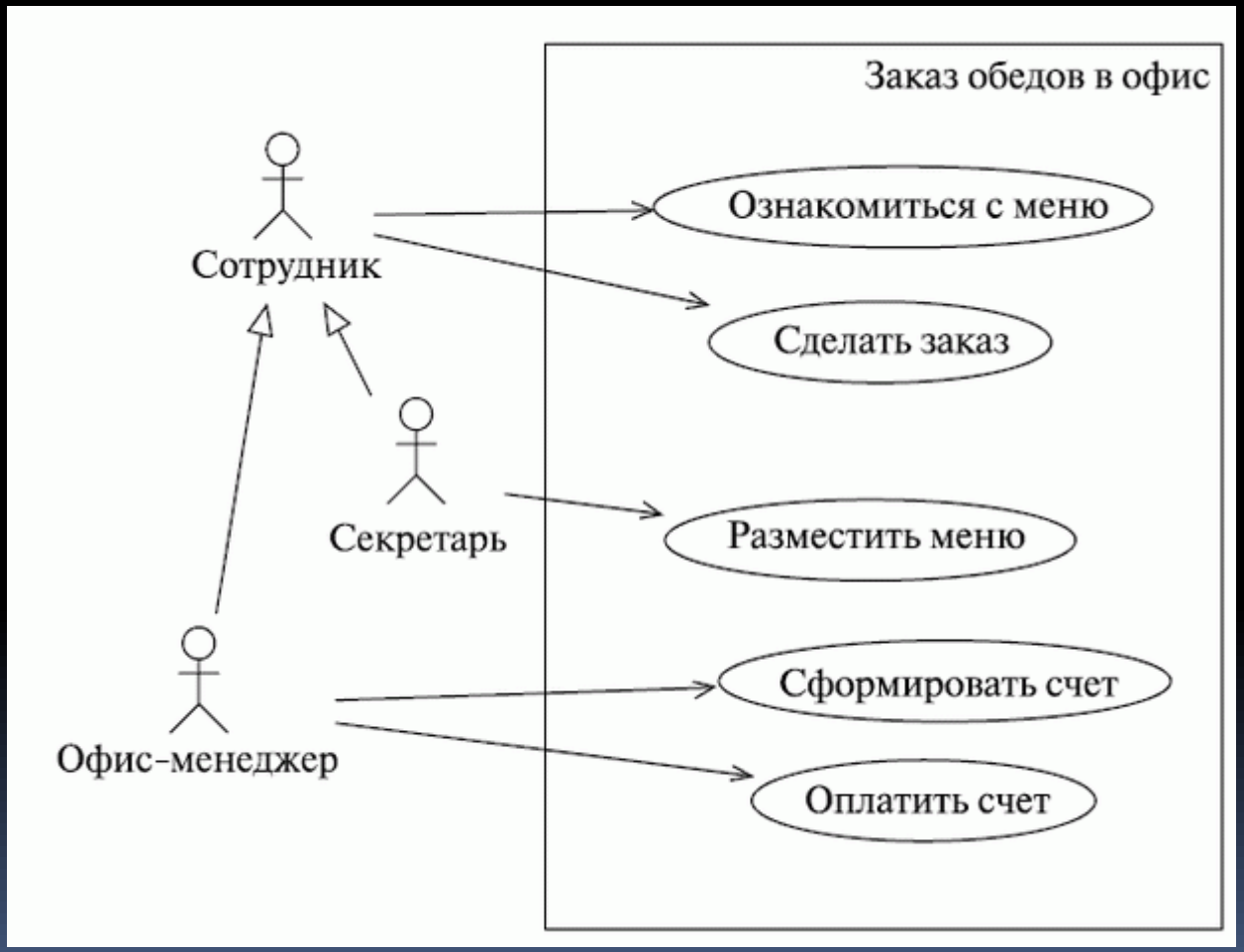
С синтаксической точки зрения *действующее лицо* — это стереотип классификатора, который обозначается специальным значком. Для действующего лица указывается только имя, идентифицирующее его в системе. Семантически действующее лицо — это множество логически взаимосвязанных ролей.

С прагматической точки зрения главным является то, что действующие лица находятся вне проектируемой системы (или рассматриваемой части системы).

Действующие лица

В качестве имен действующих лиц рекомендуется использовать существительное (возможно с определяющим словом), а в качестве имен вариантов использования — глагол (возможно, с дополнением).

Пример нотации



Варианты использования

Семантически вариант использования — это описание множества возможных последовательностей действий (событий), приводящих к значимому для действующего лица результату.

Прагматика варианта использования состоит в том, что среди всех последовательностей действий, могущих произойти при работе приложения, выделяются такие, в результате которых получается явно видимый и достаточно важный для действующего лица результат.

Варианты использования

Выбор вариантов использования сильно влияет на качество модели. Формальные методы выбора предложить трудно — помогают только опыт и чутьё.

Некоторые пункты ТЗ естественным образом переводятся в варианты использования.

Ассоциация

Ассоциация между действующим лицом и вариантом использования показывает, что действующее лицо тем или иным способом взаимодействует (предоставляет исходные данные, потребляет результат) с вариантом использования.

Ассоциация обозначает, что действующее лицо так или иначе, но обязательно непосредственно участвует в выполнении каждого из сценариев, описываемых вариантом использования.

Обобщение

Обобщение между действующими лицами показывает, что одно действующее лицо наследует все свойства (в частности, участие в ассоциациях) другого действующего лица.

С помощью обобщения между действующими лицами легко показать иерархию категорий пользователей системы, в частности, иерархию прав доступа к выполняемым функциям и хранимым данным.

Обобщение

Обобщение между вариантами использования показывает, что один вариант использования является частным случаем (подмножеством множества сценариев) другого варианта использования.

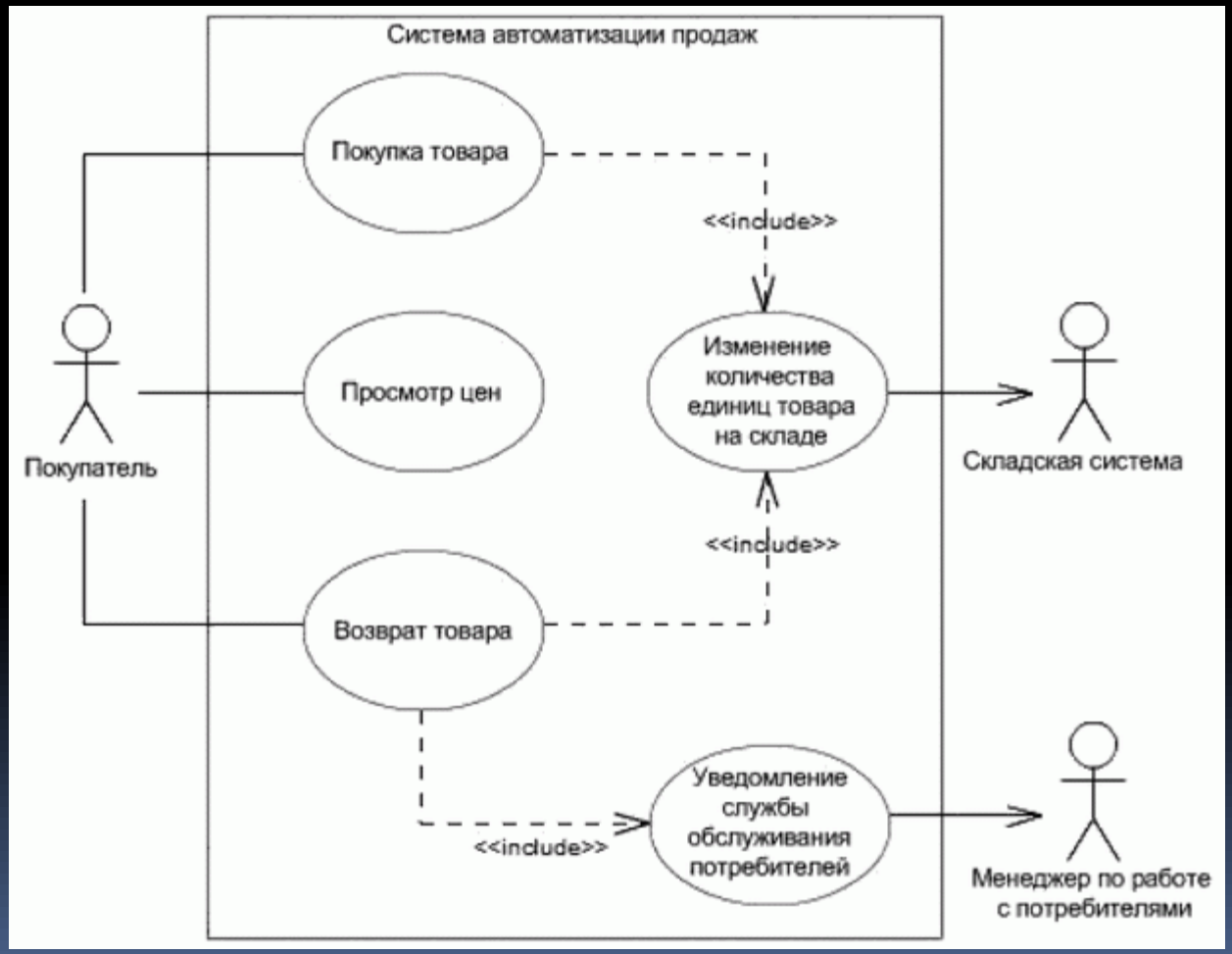
Зависимости

Зависимость между вариантами использования показывает, что один вариант использования зависит от другого варианта использования.

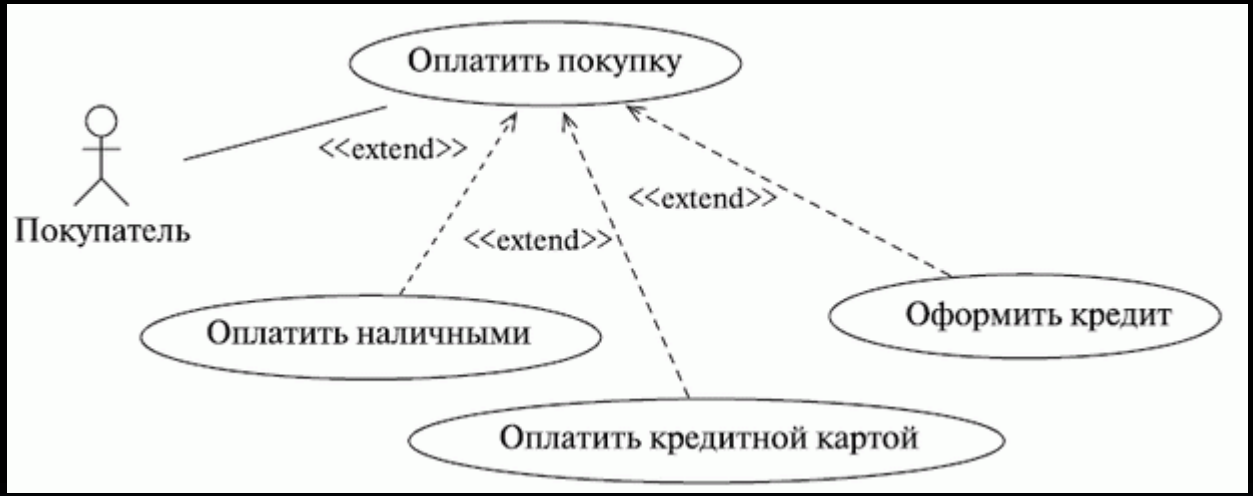
2 стандартных стереотипа зависимости:

- include — показывает, что сценарий независимого варианта использования включает в себя в качестве подпоследовательности действий сценарий зависимого варианта использования;
- extend — показывает, что в сценарий зависимого варианта использования может быть в определенном месте вставлен в качестве подпоследовательности действий сценарий независимого варианта использования.

Пример


Система автоматизации продаж
Покупка товара
<<include>>
Покупатель
Просмотр цен
Изменение количества единиц товара на складе
Складская система
<<include>>
Возврат товара
<<include>>
Уведомление службы обслуживания потребителей
Менеджер по работе с потребителями

Пример


Оплатить покупку
Покупатель
<<extend>>
<<extend>>
<<extend>>
Оплатить наличными
Оформить кредит
Оплатить кредитной картой

Реализация вариантов использования

После того, как построено представление использования, то есть выделены действующие лица, варианты использования и установлены отношения между ними, встает естественный вопрос: что дальше?

Представление использования, если оно тщательно продумано и детально прорисовано, является формой технического задания, содержащей достаточно информации для дальнейшего проектирования.

Реализация вариантов использования

Действующие лица находятся вне системы — с ними ничего делать не нужно.

Таким образом, переход от моделирования использования к другим видам моделирования состоит в уточнении, детализации и конкретизации вариантов использования.

В представлении использования мы показали, что делает система, теперь нужно определить, как это делается. Это обычно называется ***реализацией вариантов использования***.

Реализация вариантов использования

Вариант использования — это описание множества последовательностей действий, доставляющих значимый для действующего лица результат.

Наиболее часто используемый метод описания множества последовательностей действий состоит в указании алгоритма, выполнение которого доставляет последовательность действий из требуемого множества.

Реализация вариантов использования

- текстовые описания;
- псевдокод;
- диаграмма деятельности;
- *диаграммы взаимодействия.*

- Вариант использования должен доставлять значимый результат, значит, если результата нет, то что-то спроектировано не так, как нужно.

Пример текстового описания

Вариант использования «Увольнение по собственному желанию»

1. Сотрудник пишет заявление
2. Начальник подписывает заявление
3. Если есть неиспользованный отпуск, то бухгалтерия рассчитывает компенсацию
4. Бухгалтерия рассчитывает выходное пособие
5. Системный администратор удаляет учетную запись
6. Менеджер штатного расписания обновляет базу данных

Текстовые описания

Достоинства:

- просты, всем понятны, легко и быстро составляются.

Недостатки:

- неполны, неточны, ненаглядны

Псевдокод

- Если программа предназначена для выполнения компьютером, то она должна быть записана на сугубо формальном языке, который называют в этом случае *языком программирования*.
- Если программа предназначена исключительно для чтения и, может быть, выполнения человеком, то можно применить менее формальный (и более удобный) язык, который в этом случае обычно называют *псевдокодом*.

Псевдокод

- Обычно в псевдокод включают смесь общеизвестных ключевых слов языков программирования и неформальные выражения на естественном языке, обозначающие выполняемые действия.
- Эти выражения должны быть понятны человеку, который пишет или читает программу на псевдокоде, но совсем не обязаны быть допустимыми выражениями языка программирования.
- Текст на псевдокоде похож на код программы на языке программирования, но таковым не является.

Псевдокод

Достоинства способа:

- понятен, привычен и доступен любому.

Недостатки:

- плохо согласуется с парадигмой объектно-ориентированного программирования;
- отсутствуют наглядная визуализация, строгость и точность языка проектирования и реализации, поддержка распространенными инструментальными средствами;
- практически невозможно использовать повторно.

Диаграмма деятельности

- Описать алгоритм можно с помощью диаграммы деятельности.
- С одной стороны, диаграмма деятельности — это полноценная диаграмма UML, с другой стороны, диаграмма деятельности немногим отличается от блок-схемы

Диаграмма деятельности

Используются для моделирования процесса выполнения операций.

Частный случай диаграмм состояний.

Диаграмма деятельности представляется в форме графа деятельности, вершинами которого являются *состояния действия* или *деятельности*, а дугами - переходы от одного *состояния действия* к другому.

Состояния деятельности и действия

- Состояние деятельности - состояние в графе деятельности, которое служит для представления процедурной последовательности действий, требующих определенного времени.
- Состояние действия - специальный случай состояния с некоторым входным действием и, по крайней мере, одним выходящим из состояния переходом.

Переход

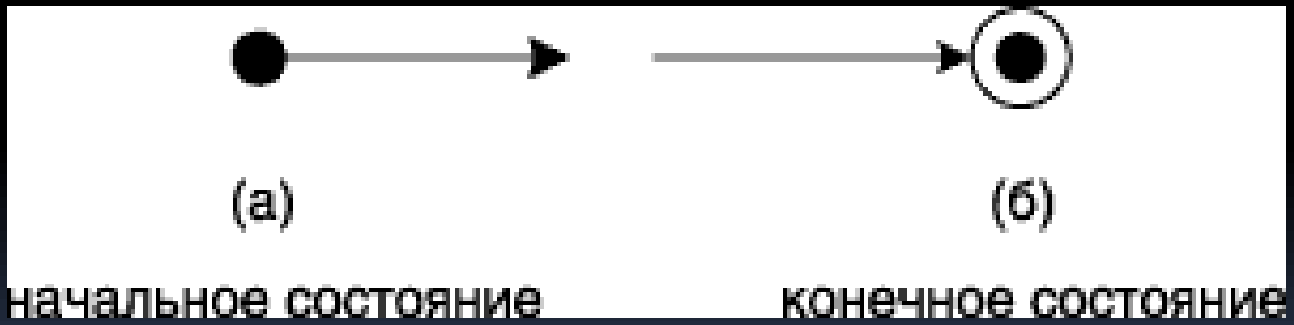
- *Переход* – отношение между двумя состояниями, которое указывает на то, что объект в первом состоянии должен выполнить определенные действия и перейти во второе состояние.
- Переход осуществляется при наступлении некоторого события: окончания выполнения деятельности (do activity), получении объектом сообщения или приемом сигнала.
- Переход изображается сплошной линией со стрелкой, которая выходит из исходного *состояния* и направлена в целевое *состояние*. Каждый *переход* может быть помечен строкой текста с именем события.

Сторожевое условие

- Логическое условие, записанное в прямых скобках и представляющее собой булевское выражение – называется *сторожевое условие*.
- При этом булевское выражение должно принимать одно из двух взаимно исключающих значений: "истина" или "ложь".

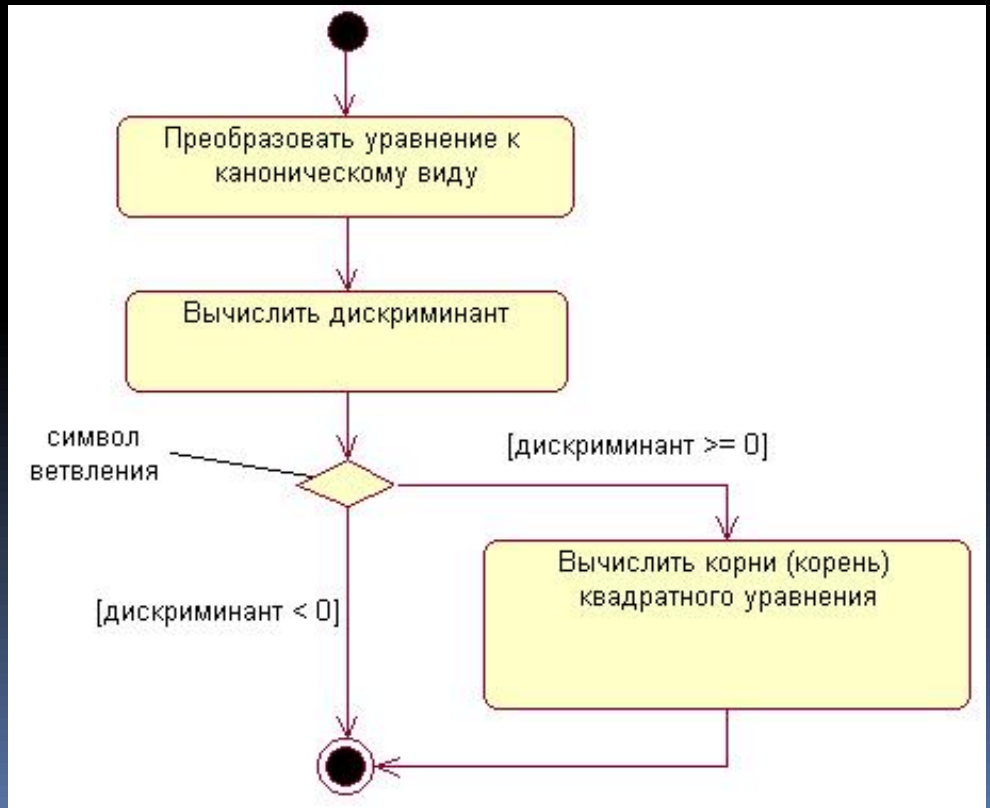
Псевдосостояние

- вершина в графе, которая имеет форму состояния, но не обладает поведением. Примерами псевдосостояний в UML являются начальное и конечное *состояния*.



Ветвление

- Если из состояния действия выходит единственный переход, то его не помечают. Если переходов несколько, для каждого из них должно быть явно записано собственное сторожевое условие.
- Графически ветвление на диаграмме деятельности обозначается символом решения (decision), изображаемого в форме небольшого ромба, внутри которого нет текста



Слияние и разделение

для представления параллельных процессов используется специальный символ
для *разделения* и *слияния* параллельных вычислений или потоков управления.

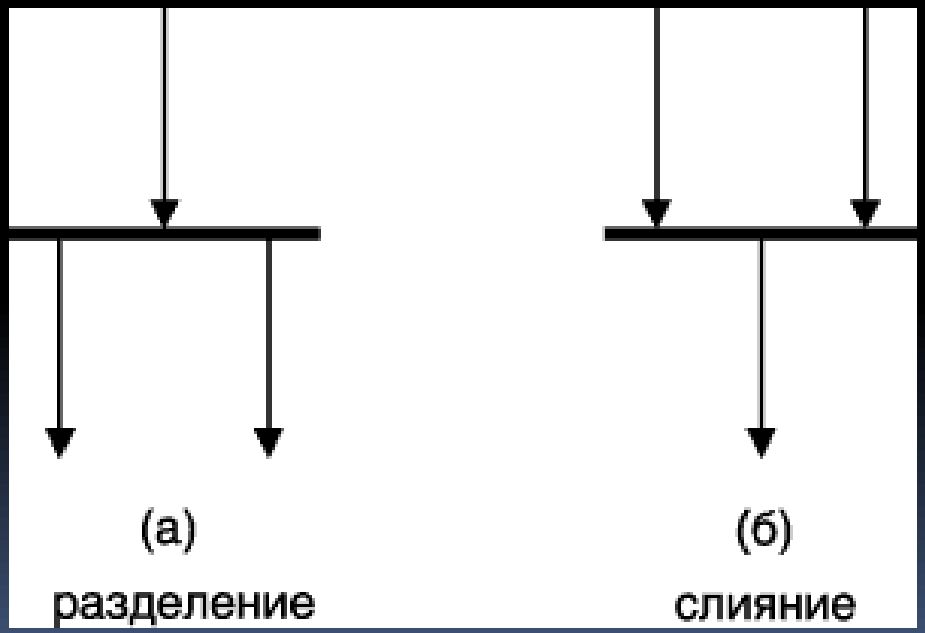


Диаграмма деятельности (пример)

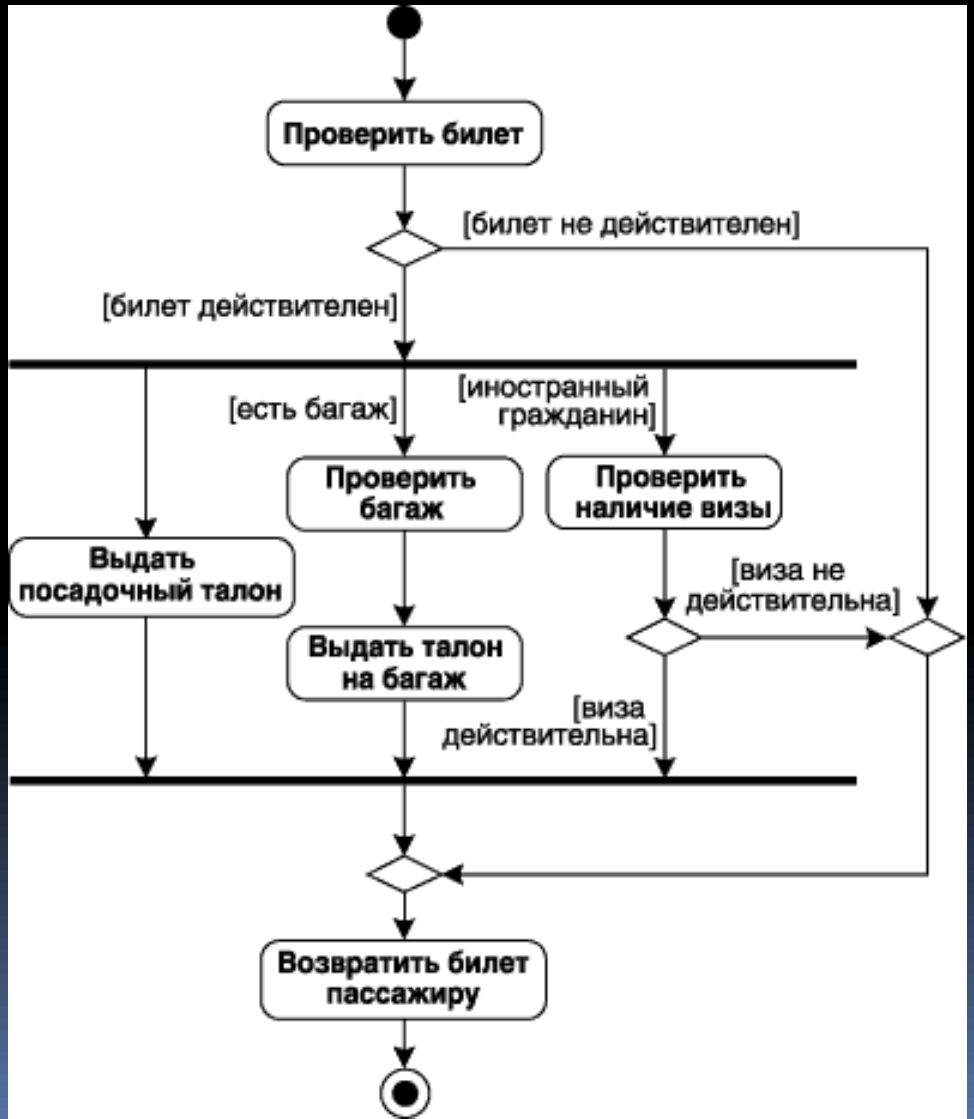


Диаграмма деятельности

- Реализация варианта использования диаграммой деятельности является компромиссным способом ведения разработки — в сущности, это проектирование сверху вниз в терминах и обозначениях UML.
- Эту диаграмму можно показать заказчику, чтобы проверить, действительно ли проектируемая нами логика работы системы соответствует тому бизнес-процессу, который существует в реальности.

Диаграмма деятельности

- Применение диаграмм деятельности для реализации вариантов использования не слишком приближает к появлению целевого артефакта — программного кода, однако может привести к более глубокому пониманию существа задачи и даже открыть неожиданные возможности улучшения приложения, которые было трудно усмотреть в первоначальной постановке задачи.

Моделирование использования

- Реализация варианта использования диаграммой деятельности является компромиссным способом ведения разработки – в сущности, это проектирование сверху вниз в терминах и обозначениях UML.
- Другие способы реализации вариантов использования: текстовое описание прецедентов, диаграммы взаимодействия (диаграмма последовательности).

Диаграмма деятельности

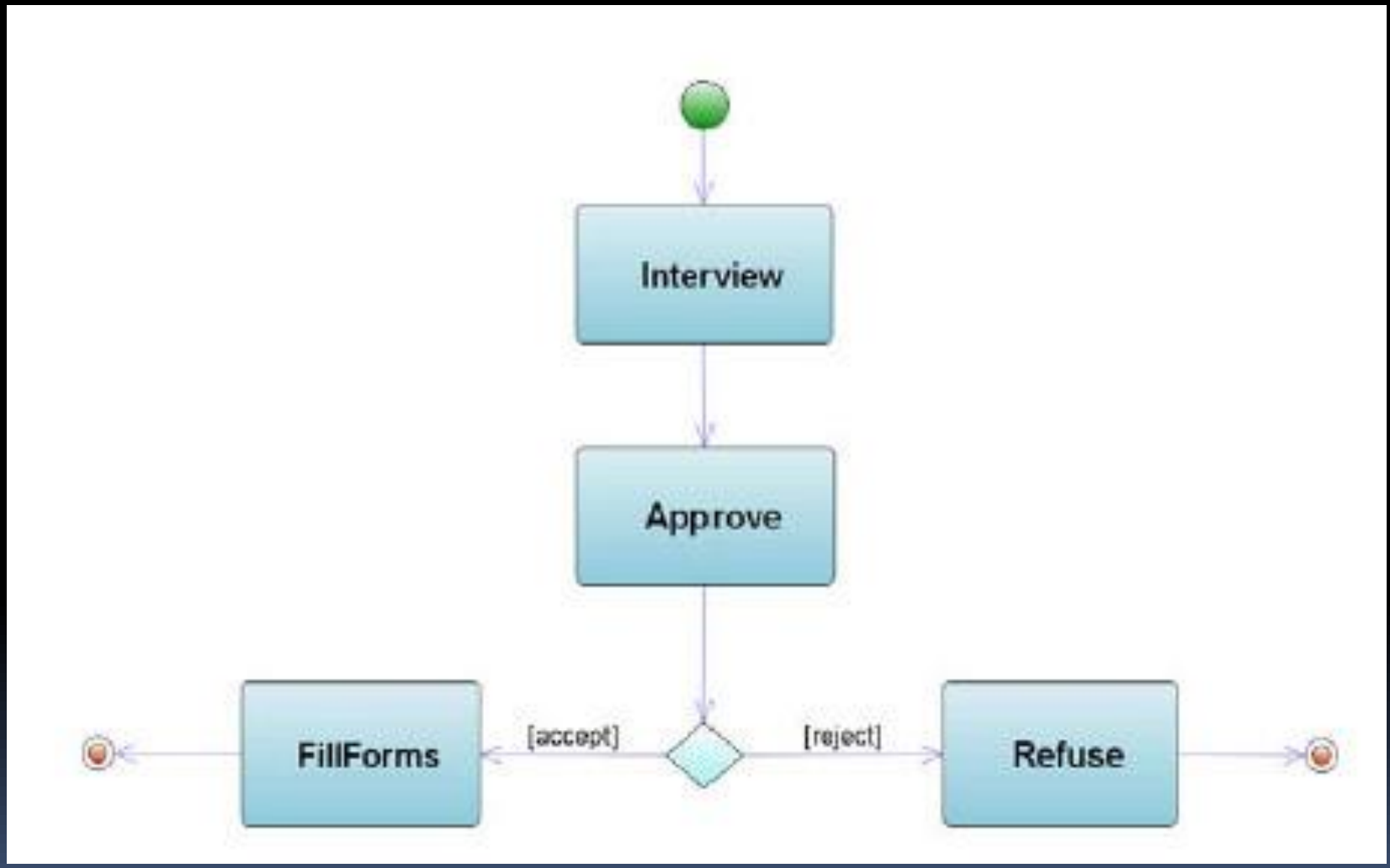
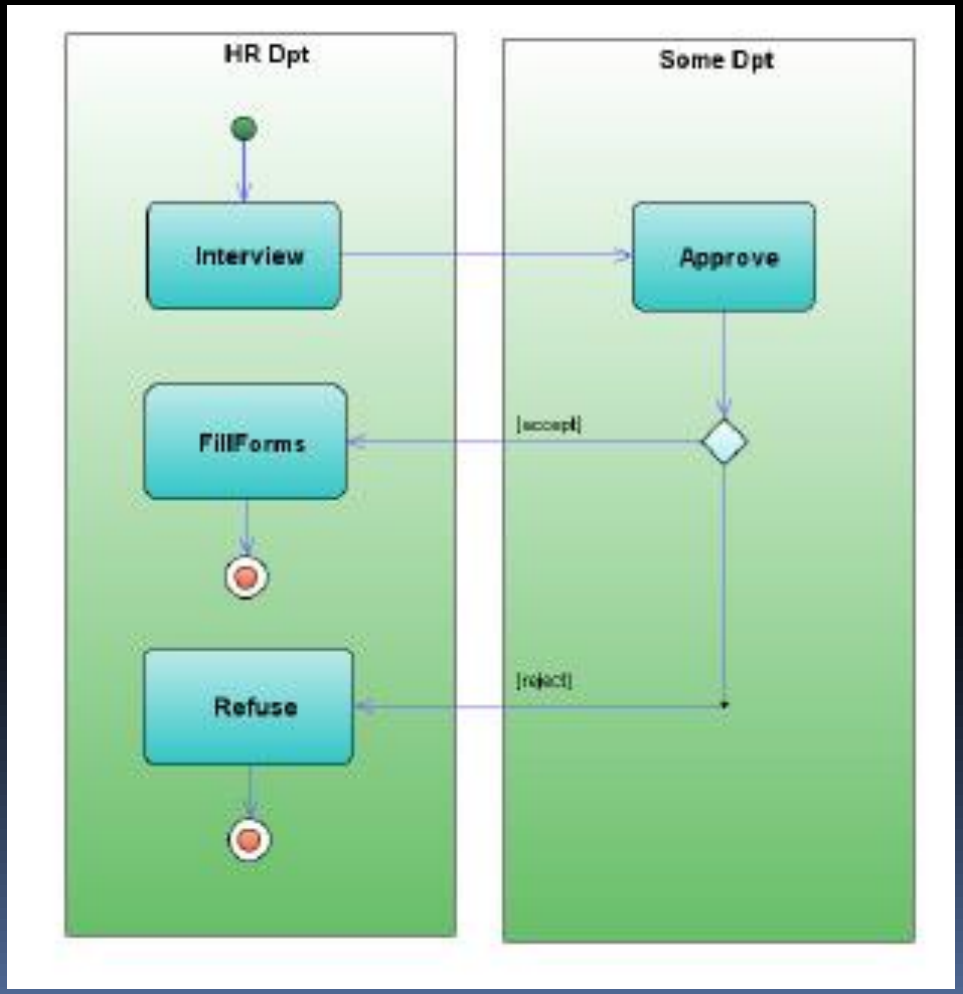


Диаграмма деятельности (дорожки)



Идентификация действующих лиц

Упрощенная версия банкомата (automatic teller machine, ATM) предлагает пользователю следующие сервисы:
1. Управление финансами владельцев карт при помощи карт ридера и диспенсера.
2. Информирование о состоянии счета, возможность пополнять счет и оплачивать услуги для владельцев карт банка, который обслуживает банкомат.
Не забудьте также:
3. Все транзакции должны быть безопасными. С этой целью происходит аутентификация через Visa authorisation system (для простоты ограничимся только этой системой).
4. Банкомат нуждается в регулярном обслуживании. Например, требуется пополнить диспенсер.

Идентификация действующих лиц

Утверждение1 позволяет идентифицировать первое очевидное действующее лицо: «cardholder». При этом «card reader» и «cash dispenser» являются неотъемлемой частью ATM. Поэтому не являются действующими лицами.

Утверждение 2 идентифицирует дополнительные сервисы предлагаемые только владельцам карт банка, которому принадлежит банкомат. Определим второе действующее лицо: bank customer.

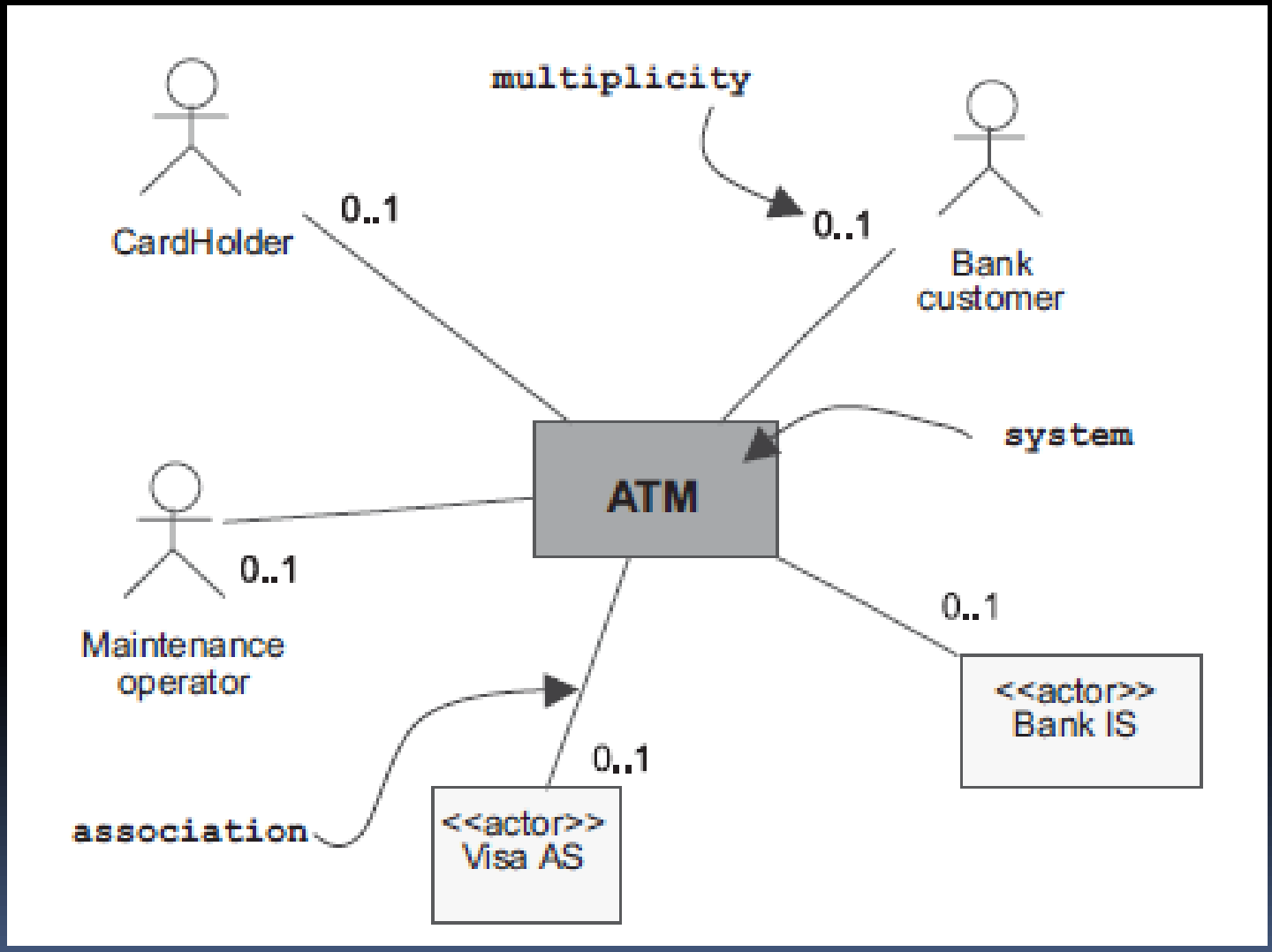
Идентификация действующих лиц

В утверждении 3 сказано, что транзакции должны быть безопасными. Что их делает таковыми?
Для этого существуют другие внешние сущности. В частности системы авторизации , с которыми взаимодействует банкомат напрямую.
Интервью с экспертом позволило выделить две таких сущности:
• Visa authorisation system (VISA AS) для авторизации и проведения транзакций для владельцев карты Visa;
• информационная система банка (Bank IS) для авторизации всех транзакций, которые проводит владелец карты банка.

Идентификация действующих лиц

Наконец, утверждение 4 напоминает нам, что банкомат требует обслуживания – пополнение диспенсера банкнотами, извлечение забытых и застрявших карт и т.п. Эти задачи решает ещё одно действующее лицо: maintenance operator.

Идентификация действующих лиц



Идентификация вариантов использования

CardHolder:
• Withdraw money
Bank customer:
• Withdraw money.
• Consult the balance of one or more accounts.
• Deposit cash.
• Deposit cheques.
Maintenance operator:
• Refill dispenser.
• Retrieve cards that have been swallowed.
• Retrieve cheques that have been deposited.
Visa authorisation system (AS):
• None.
Bank information system (IS):
• None.

Выводы

- Составление диаграмм использования — это первый шаг моделирования.
- Основное назначение диаграммы использования — показать, что делает система во внешнем мире.
- Диаграмма использования не зависит от программной реализации системы и поэтому не обязана соответствовать структуре классов, модулей и компонентов системы.
- Идентификация действующих лиц и вариантов использования — ключ к дальнейшему проектированию.

Выводы

- В зависимости от выбранной парадигмы проектирования и программирования применяются различные способы реализации вариантов использования.
- Реализация варианта использования диаграммой деятельности является компромиссным способом ведения разработки — в сущности, это проектирование сверху вниз в терминах и обозначениях UML.