# ЛАБОРАТОРНАЯ РАБОТА № 1
# ФОРМИРОВАНИЕ МОДЕЛИ ФУНКЦИОНИРОВАНИЯ ПРОГРАММНОГО СРЕДСТВА С ИСПОЛЬЗОВАНИЕМ UML

## 1 Цель занятия

Научиться формировать диаграммы вариантов использования (use case diagram) и диаграммы кооперации для формирования модели процесса в рамках которого будет функционировать проектируемое программное средство

## 2 Общие теоретические сведения

2.1 Диаграмма вариантов использования (use case diagram)

Визуальное моделирование в UML можно представить как процесс декомпозиции наиболее обшей и абстрактной концептуальной модели исходной системы до логической, а затем и до физической модели соответствующей программной системы. Для достижения этих целей вначале строится модель в форме так называемой диаграммы вариантов использования (use case diagram), которая описывает функциональное назначение системы или, другими словами, то, что система будет делать в процессе своего функционирования.

Разработка диаграммы вариантов использования преследует цели:

– определить общие границы и контекст моделируемой предметной области на начальных этапах проектирования системы;

– сформулировать общие требования к функциональному поведению проектируемой системы;

– разработать исходную концептуальную модель системы для ее последующей детализации в форме логических и физических моделей;

– подготовить исходную документацию для взаимодействия разработчиков системы с ее заказчиками и пользователями.

Суть данной диаграммы состоит в следующем: проектируемая система представляется в виде множества сущностей или актеров, взаимодействующих с системой с помощью так называемых вариантов использования. При этом актером (actor) или действующим лицом является любая сущность, взаимодействующая с системой извне. Это может быть человек, техническое устройство, программа или любая другая система, которая может служить источником воздействия на моделируемую систему так, как определит сам разработчик. В свою очередь, вариант использования (use case) служит для описания сервисов, которые система предоставляет актеру.

В самом общем случае, диаграмма вариантов использования представляет собой граф специального вида, который является графической нотацией для представления конкретных вариантов использования, актеров, возможно некоторых интерфейсов, и отношений между этими элементами. При этом отдельные компоненты диаграммы могут быть заключены в прямоугольник, который обозначает проектируемую систему в целом. Следует отметить, что отношениями

данного графа могут быть только некоторые фиксированные типы взаимосвязей между актерами и вариантами использования, которые в совокупности описывают сервисы или функциональные требования к моделируемой системе.

2.1.1. Вариант использования

Каждый вариант использования определяет последовательность действий, которые должны быть выполнены проектируемой системой при взаимодействии ее с соответствующим актером. Диаграмма вариантов может дополняться пояснительным текстом, который раскрывает смысл или семантику составляющих ее компонентов. Такой пояснительный текст получил название примечания или сценария.

Отдельный вариант использования обозначается на диаграмме эллипсом, внутри которого содержится его краткое название или имя в форме глагола с пояснительными словами (рис. 1).

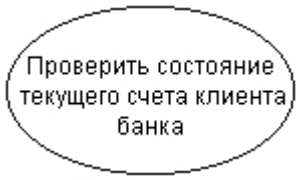


Рис. 1 Графическое обозначение варианта использования

Цель варианта использования заключается в том, чтобы определить законченный аспект или фрагмент поведения некоторой сущности без раскрытия внутренней структуры этой сущности. В качестве такой сущности может выступать исходная система или любой другой элемент модели, который обладает собственным поведением, подобно подсистеме или классу в модели системы.

Каждый вариант использования соответствует отдельному сервису, который предоставляет моделируемую сущность или систему по запросу пользователя (актера), т.е. определяет способ применения этой сущности. Сервис, который инициализируется по запросу пользователя, представляет собой законченную последовательность действий. Это означает, что после того как система закончит обработку запроса пользователя, она должна возвратиться в исходное состояние, в котором готова к выполнению следующих запросов.

Варианты использования описывают не только взаимодействия между пользователями и сущностью, но также реакции сущности на получение отдельных сообщений от пользователей и восприятие этих сообщений за пределами сущности. Варианты использования могут включать в себя описание особенностей способов реализации сервиса и различных исключительных ситуаций, таких как корректная обработка ошибок системы. Множество вариантов использования в целом должно определять все возможные стороны ожидаемого поведения системы. Для удобства множество вариантов использования может рассматриваться как отдельный пакет.

В метамодели UML вариант использования является подклассом классификатора, который описывает последовательности действий, выполняемых

отдельным экземпляром варианта использования. Эти действия включают изменения состояния и взаимодействия со средой варианта использования. Эти последовательности могут описываться различными способами, включая такие, как графы деятельности и автоматы.

Примерами вариантов использования могут являться следующие действия: проверка состояния текущего счета клиента, оформление заказа на покупку товара, получение дополнительной информации о кредитоспособности клиента, отображение графической формы на экране монитора и другие действия.

2.1.2. Актеры

Актер представляет собой любую внешнюю по отношению к моделируемой системе сущность, которая взаимодействует с системой и использует ее функциональные возможности для достижения определенных целей или решения частных задач. При этом актеры служат для обозначения согласованного множества ролей, которые могут играть пользователи в процессе взаимодействия с проектируемой системой. Каждый актер может рассматриваться как некая отдельная роль относительно конкретного варианта использования. Стандартным графическим обозначением актера на диаграммах является фигурка "человечка", под которой записывается конкретное имя актера (рис. 2).

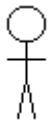

Рис. 2 Графическое обозначение актера

В некоторых случаях актер может обозначаться в виде прямоугольника класса с ключевым словом "актер" и обычными составляющими элементами класса. Имена актеров должны записываться заглавными буквами и следовать рекомендациям использования имен для типов и классов модели. При этом символ отдельного актера связывает соответствующее описание актера с конкретным именем. Имена абстрактных актеров, как и других абстрактных элементов языка UML, рекомендуется обозначать курсивом.

Имя актера должно быть достаточно информативным с точки зрения семантики. Примерами актеров могут быть: клиент банка, банковский служащий, продавец магазина, менеджер отдела продаж, пассажир авиарейса, водитель автомобиля, администратор гостиницы, сотовый телефон и другие сущности, имеющие отношение к концептуальной модели соответствующей предметной области.

В качестве актеров могут выступать другие системы, подсистемы проектируемой системы или отдельные классы. Важно понимать, что каждый актер определяет некоторое согласованное множество ролей, в которых могут выступать пользователи данной системы в процессе взаимодействия с ней. В каждый момент времени с системой взаимодействует вполне определенный пользователь, при этом

он играет или выступает в одной из таких ролей. Наиболее наглядный пример актера – конкретный пользователь системы со своими собственными параметрами аутентификации.

Так как в общем случае актер всегда находится вне системы, его внутренняя структура никак не определяется. Для актера имеет значение только его внешнее представление, т.е. то, как он воспринимается со стороны системы. Актеры взаимодействуют с системой посредством передачи и приема сообщений от вариантов использования. Сообщение представляет собой запрос актером сервиса от системы и получение этого сервиса. Это взаимодействие может быть выражено посредством ассоциаций между отдельными актерами и вариантами использования или классами.

Два и более актера могут иметь общие свойства, т.е. взаимодействовать с одним и тем же множеством вариантов использования одинаковым образом.

2.1.3. Интерфейсы

Интерфейс (interface) служит для спецификации параметров модели, которые видимы извне без указания их внутренней структуры. В языке UML интерфейс является классификатором и характеризует только ограниченную часть поведения моделируемой сущности. Применительно к диаграммам вариантов использования, интерфейсы определяют совокупность операций, которые обеспечивают необходимый набор сервисов или функциональности для актеров. Интерфейсы не могут содержать ни атрибутов, ни состояний, ни направленных ассоциаций. Они содержат только операции без указания особенностей их реализации. Формально интерфейс эквивалентен абстрактному классу без атрибутов и методов с наличием только абстрактных операций.

На диаграмме вариантов использования интерфейс изображается в виде маленького круга, рядом с которым записывается его имя (рис. 3, а). В качестве имени может быть существительное, которое характеризует соответствующую информацию или сервис (например, "датчик", "сирена", "видеокамера"), но чаще строка текста (например, "запрос к базе данных", "форма ввода", "устройство подачи звукового сигнала"). Если имя записывается на английском, то оно должно начинаться с заглавной буквы I, например, ISecurelnformation, ISensor (рис. 3, б).

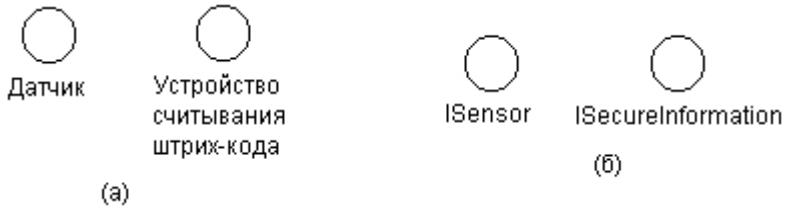


Рис. 3. Графическое изображение интерфейсов на диаграммах вариантов использования

Графический символ отдельного интерфейса может соединяться на диаграмме сплошной линией с тем вариантом использования, который его поддерживает.

Сплошная линия в этом случае указывает на тот факт, что связанный с интерфейсом вариант использования должен реализовывать все операции, необходимые для данного интерфейса, а возможно и больше (рис. 4, а). Кроме этого, интерфейсы могут соединяться с вариантами использования пунктирной линией со стрелкой (рис. 4, б), означающей, что вариант использования предназначен для спецификации только того сервиса, который необходим для реализации данного интерфейса.


Рис. 4. Графическое изображение взаимосвязей интерфейсов с вариантами использования

Важность интерфейсов заключается в том, что они определяют стыковочные узлы в проектируемой системе, что совершенно необходимо для организации коллективной работы над проектом. Более того, спецификация интерфейсов способствует "безболезненной" модификации уже существующей системы при переходе на новые технологические решения. В этом случае изменению подвергается только реализация операций, но никак не функциональность самой системы. А это обеспечивает совместимость последующих версий программ с первоначальными при спиральной технологии разработки программных систем.

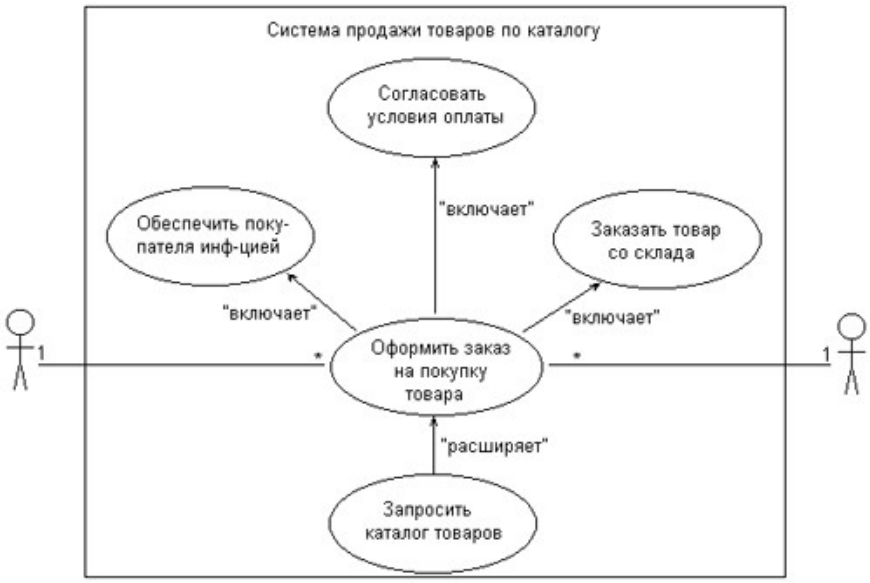


Рис. 5 Пример диаграммы вариантов использования

## 2.2 Пример решения задачи по формированию диаграммы вариантов использования

### 2.3 Формирование диаграммы кооперации

Диаграмма кооперации предназначена для спецификации структурных аспектов взаимодействия. Главная особенность диаграммы кооперации заключается

в возможности графически представить не только последовательность взаимодействия, но и все структурные отношения между объектами, участвующими в этом взаимодействии.

На диаграмме кооперации в виде прямоугольников изображаются участвующие во взаимодействии объекты, содержащие имя объекта, его класс и, возможно, значения атрибутов. Далее указываются ассоциации между объектами в виде различных соединительных линий. При этом можно явно указать имена ассоциации и ролей, которые играют объекты в данной ассоциации. Дополнительно могут быть изображены динамические связи — потоки сообщений. Они представляются также в виде соединительных линий между объектами, над которыми располагается стрелка с указанием направления, имени сообщения и порядкового номера в общей последовательности инициализации сообщений.

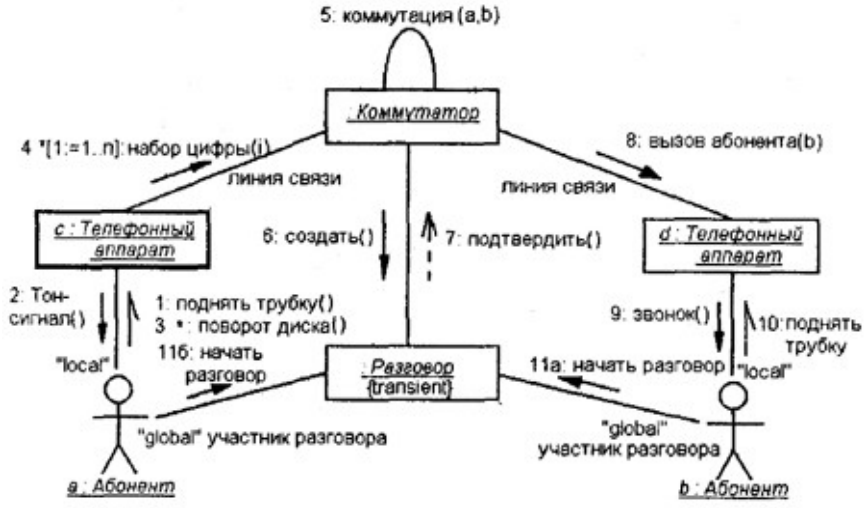


Рис. 6 Пример диаграммы кооперации для моделирования телефонного разговора

**3 Задачи для самостоятельного решения студентами**

3.1 Получить у преподавателя один из следующих вариантов проектируемой программной системы:

**Вариант 1**

Программное средство, обслуживающее автомат, заваривающий кофе. Программа должна уметь получать сигнал с датчиков нажатия кнопок «эспрессо», «американо», «латте», «капучино» (интерфейсы), считывать сигнал с кнопки «сахар» (актёр), считывать данные с купюроприёмника, анализировать достаточность внесённых денег, запускать необходимые программы варки кофе, запускать программу добавки сахара, запускать механизм заливки кофе в стаканчик и выдачи стаканчика, выдавать предупреждения, диагностические сообщения и прочее.

**Вариант 2**

Программное средство Call-центра технической поддержки сотового оператора, обеспечивающее клиента (актёр) функциями связи с менеджером (актёр), выдачи баланса, консалтинга по тарифам, подключения / отключения услуг и не

менее 5 других функций. Функция со справкой по тарифам должна предполагать внутреннее голосовое меню.

**Вариант 3**

Программное средство банкомата, выполняющего только функцию выдачи денег. Взаимодействуют клиент, система обслуживания, купюроприёмник, кнопочная панель, определяющая снимаемую сумму и кнопки подтверждения или отмены. Функции клиента – вставить карту, ввести код, запросить баланс, набрать сумму, получить деньги. Функции система обслуживания – получать и распознавать сигналы с датчиков кнопок, выводить на кран распознанные цифры, выводить на экран необходимые приглашения и сообщения (на схеме они должны быть расшифрованы), анализировать наличие в банкомате средств, отдавать в купюроприёмник информацию о нужном количестве денег на выдачу, формировать и распечатывать выписку, функция купюроприёмника – выдача денег; функции кнопок – формирование соответствующих сигналов нажатия.

**Вариант 4**

Программное средство Call-центра салона красоты, обеспечивающее взаимодействие клиента, менеджера и базы учёта. Call-центр выдаёт стандартное приветствие и предоставляет клиенту голосовое меню, с помощью которого он может получить информацию об услугах салона, записаться на сеанс к стилисту, вейзажисту, массажисту, парикмахеру и т.д., связаться с менеджером, отменить, изменить время сеанса, записаться на повторный приём, получить справку. Менеджер может напрямую работать с базой учёта, внося туда сведения о клиентах и мастерах.

**Вариант 5**

Программное средство формирования командировочного удостоверения преподавателя вуза. Программное средство управляется бухгалтером. Преподаватель заполняет электронную анкету с типом командировки (по России или международная), местом назначения, временем пребывания, целью и способом проезда к месту командировки. Руководство вуза выдаёт (или не выдаёт) разрешение на командировку, в результате чего на основе анкетных данных преподавателя и реквизитов вуза, полученных из учётной базы данных, формируется проект приказа на командировку преподавателя, который получает преподаватель и предоставляет бухгалтеру. Бухгалтер оценивает по справочникам примерную стоимость проезда, проживания и величину суточных расходов и формирует величину аванса, который начисляется программным средством на зарплатную карточку преподавателя. По истечении командировки преподаватель предоставляет бухгалтеру чеки, подтверждающие использование аванса. Если потрачено меньше, чем запланировано, то осуществляется удержание с преподавателя разницы, если больше – то доначисление.

**Вариант 6**

Программное средство библиотечного каталога вуза, взаимодействующее со студентами, библиотекарями и преподавателями. Функции преподавателей – поиск учебников, заказ на закупку учебников; функции библиотекарей – внесение (исключение) записей в базу данных (или подтверждение по запросам студентов через интернет), распределение полномочий; функции студентов – алфавитный, тематический и не менее 2-х видов других поисков, запрос электронной версии, запрос на заказ учебника, запрос на справку, связь с библиотекарем через интернет или телефон.

**Вариант 7**

Программное средство регистратуры детской стоматологической клиники, имеющее агентов: секретарь, врачи и клиенты. Функции секретаря – внесение, исключение записей в базе данных (или подтверждение по запросам клиентов через интернет), запрос расписания врачей, определение времени приёма по справочнику, исключение или перенос записей по требованию врача с уведомлением клиентов, связь с врачом. Функции врача – анализ заявок на обслуживание и формирование предложений по оптимизации записей при наличии ошибок или иным обстоятельствам, заполнение карточки по результатам приёма; функции клиентов – поиск информации о расписании врачей, получение информации об опыте, образовании и достижениях врачей, запись на приём (по телефону, лично у секретаря или через интернет), запрос на справку у врача, связь с секретарём через интернет или телефон.

**Вариант 8**

Программное средство регистрации клиентов в гостинице, включающее агентов: администратор, персонал, клиент. Функции администратора: внесение паспортных и анкетных данных клиента, получение сведений о свободном номерном фонде, бронирование или немедленное предоставление ключа доступа в комнату клиента, выписка клиента, выдача справок клиенту, приём заявок на вызов персонала гостиницы (горничные, плотники, слесаря и т.д.); функции персонала – регистрация времени посещения клиента и сделанных работ; функции клиента – бронирование по телефону, лично у администратора или через интернет номера в гостинице; заполнение через интернет анкеты для заселения или передача информации администратору, получение справки у администратора, связь с администратором, получение карты доступа в комнату, использование карты доступа в номер.

3.2 Полностью раскрыть все варианты использования, включая те, которые не упомянуты в задании. Вариантов использования в одном варианте должно быть не менее 15-20. Актёров (интерфейсов) не менее 3-х. Логика функционирования программного средства должна быть полностью описана.

В результате должен быть сформирован список актёров (интерфейсов) и их функций (вариантов использования) в виде уровневого списка или таблицы.

3.3 Изучить текст теоретической части методических указаний.

3.4 Сформировать диаграмму вариантов использования с использованием программы Rational Rose, полностью описывающую функциональное назначение программного средства согласно варианту, назначенному преподавателем.

3.5 Сформировать диаграмму кооперации с использованием программы Rational Rose согласно варианту, назначенному преподавателем.