Федеральное агентство связи

Федеральное государственное образовательное бюджетное учреждение высшего профессионального образования

ПОВОЛЖСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ ТЕЛЕКОММУНИКАЦИЙ И ИНФОРМАТИКИ

ЭЛЕКТРОННАЯ БИБЛИОТЕЧНАЯ СИСТЕМА

Самара

Федеральное агентство связи

Федеральное государственное образовательное бюджетное
учреждение высшего профессионального образования

Поволжский государственный университет
телекоммуникаций и информатики

Кафедра «Информационные системы и технологии»

Пальмов С.В.

МЕТОДЫ И СРЕДСТВА ПРОЕКТИРОВАНИЯ ИНФОРМАЦИОННЫХ СИСТЕМ И ТЕХНОЛОГИЙ

Часть 1. Изучение возможностей UML

методические указания
к лабораторным работам по дисциплине
для студентов очной формы обучения направления
«Информационные системы и технологии»

Самара, ИУНЛ ПГУТИ, 2013

Пальмов С.В. Методические указания к лабораторным работам по дисциплине Методы и средства проектирования информационных систем и технологий для студентов очной формы обучения направления (Информационные системы и технологии). Изучение возможностей UML. – Самара: ПГУТИ, 2013. – 54 с., ил.

Методические указания предназначены для студентов очного отделения специальности 230400 (Информационные системы и технологии) по дисциплине «Методы и средства проектирования информационных систем и технологий». Лабораторный цикл включает в себя восемь лабораторных работ, выполнение которых поможет студентам изучить основные возможности Унифицированного языка моделирования (UML).

Методические указания подготовлены на кафедре «Информационные системы и технологии».

Методические указания рекомендованы к изданию методическим Советом ПГУТИ (заседание №6 от 25.09.2013)

СОДЕРЖАНИЕ

Лабораторная работа №1. Построение диаграммы вариантов использования

Цель: Научиться строить диаграммы вариантов использования.

Введение

Краткая характеристика диаграммы вариантов использования

Визуальное моделирование в UML (Унифицированный Язык Моделирования) можно представить как некоторый процесс поуровневого спуска от наиболее обшей и абстрактной концептуальной модели исходной системы к логической, а затем и к 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. Диаграмма вариантов использования является исходным концептуальным представлением или концептуальной моделью системы в процессе ее проектирования и разработки.

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 называется 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. Другими словами, каждый вариант использования определяет некоторый набор действий, совершаемый системой при диалоге с актером. При этом ничего не говорится о том, каким образом будет реализовано взаимодействие актеров с системой.

Описание ПО StarUML

Эта и все последующие лабораторные работы выполняются средствами ПО StarUML. Данное программное обеспечение является бесплатным и может быть скачано с официального сайта http://staruml.sourceforge.net/en/.

StarUML - программный инструмент моделирования, который поддерживает UML

(StarUML ориентирован на UML версии 1.4 и поддерживает одиннадцать различных типов диаграмм, принятых в нотации UML 2.0).

Построение диаграмм вариантов использования в StarUML

После запуска программы будет предложено выбрать тип проекта. Необходимо выбрать Rational Approach.

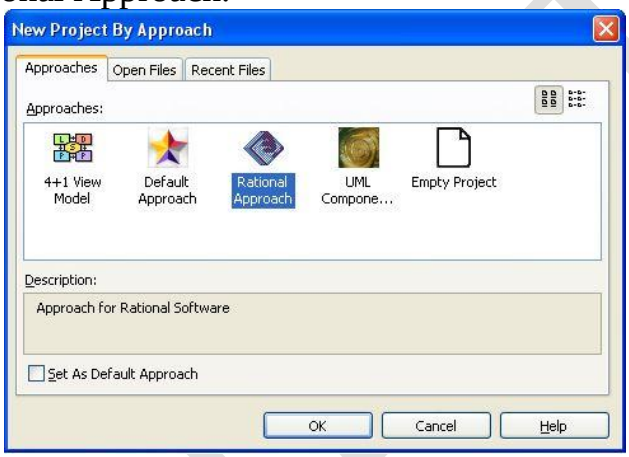

Рис. 1. Выбор типа проекта

Далее необходимо создать новую модель. Для этого выполните следующие действия: Model → Add → Model.

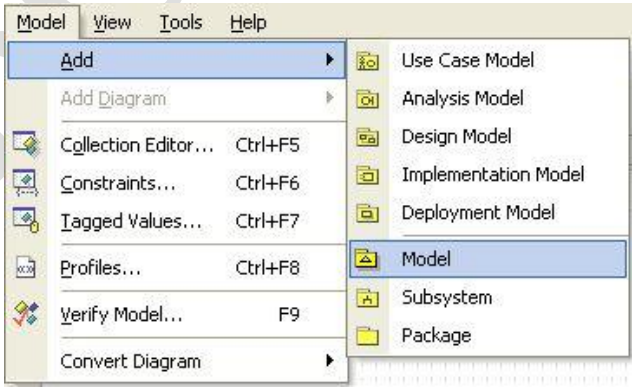

Рис. 2. Добавление модели

Созданная модель появится в окне Навигаторе моделей (Model Explorer). По умолчанию оно располагается в правом верхнем углу.

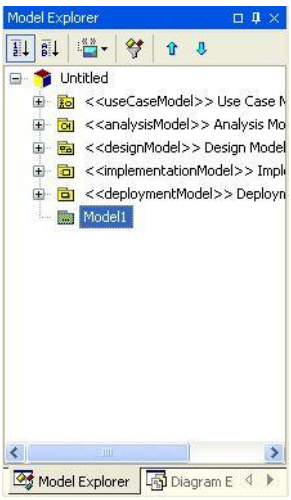

Рис. 3. Созданная модель

Теперь добавим диаграмму вариантов использования: щелчок ПКМ по созданной модели в Model Explorer → из контекстного меню выберите Add Diagram (Добавить диаграмму) → выберите Use Case Diagram (Диаграмма вариантов использования).

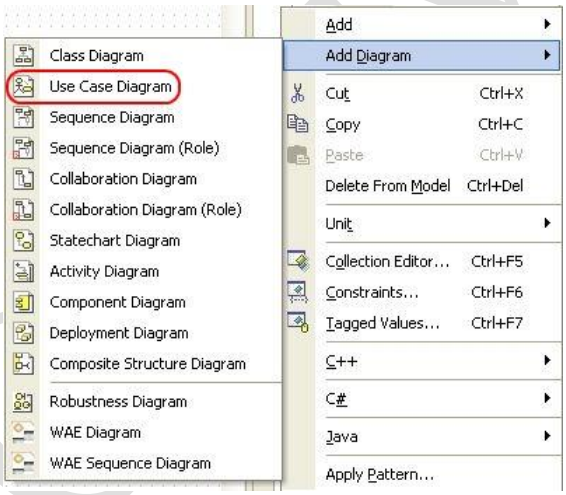

Рис. 4. Добавление диаграммы вариантов использования

Добавление элементов диаграммы

Чтобы добавить вариант использования (Use Case) выполните следующие действия: панель Toolbox (слева вверху) → закладка Use Case → Use Case. После этого щёлкните ЛКМ по рабочей области в том месте, где будет размещаться данный элемент. Имя варианта использования необходимо задать сразу после его добавления на диаграмму. Отредактировать имя – двойной щелчок ЛКМ по варианту использования.

Чтобы добавить актёра (Actor) выполните следующие действия: панель Toolbox (слева вверху) → закладка Use Case → Actor. После этого щёлкните ЛКМ по рабочей области в том месте, где будет размещаться данный элемент. Имя актёра необходимо задать сразу после его добавления на диаграмму. Отредактировать имя – двойной щелчок ЛКМ по актёру.

Чтобы добавить отношение на диаграмму вариантов использования выполните следующие действия: панель Toolbox (слева вверху) → закладка Use Case → (нужный тип отношения).

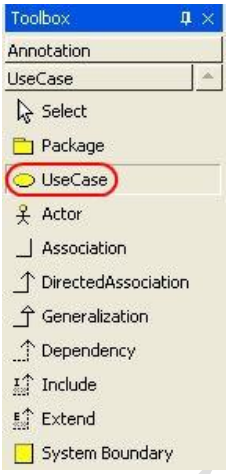

Рис. 5. Выбрано добавление варианта использования

Рис. 6. Добавленный вариант использования

Рис. 7. Добавленный актёр

Создание текстового сценария

Текстовые сценарии уточняют или детализируют последовательность действий, совершаемых системой при выполнении её вариантов использования.

Текстовые сценарии строят только для базовых вариантов использования.

Для построения может быть использован любой текстовый редактор, в котором имеется возможность построения таблиц.

Рис. 8. Отношения на диаграмме вариантов использования

Домашнее задание студентам для подготовки к выполнению лабораторной работы

Изучить по лекциям и учебной литературе особенности построения диаграмм вариантов использования.

Изучить принципы построения диаграмм вариантов использования в StarUML (см. руководство пользователя).

Варианты заданий

Вариант задания (тема) выбирается один раз. Все лабораторные работы выполняются по одной теме. Перечень тем приведён в конце методической разработки. Тема выбирается согласно номеру студента в списке группы.

Порядок выполнения лабораторной работы

1. Запустите StarUML и создайте модель. Назовите её Model1.
2. Постройте диаграмму вариантов использования, приведённую на рисунке 9.
3. Сохраните результаты работы.

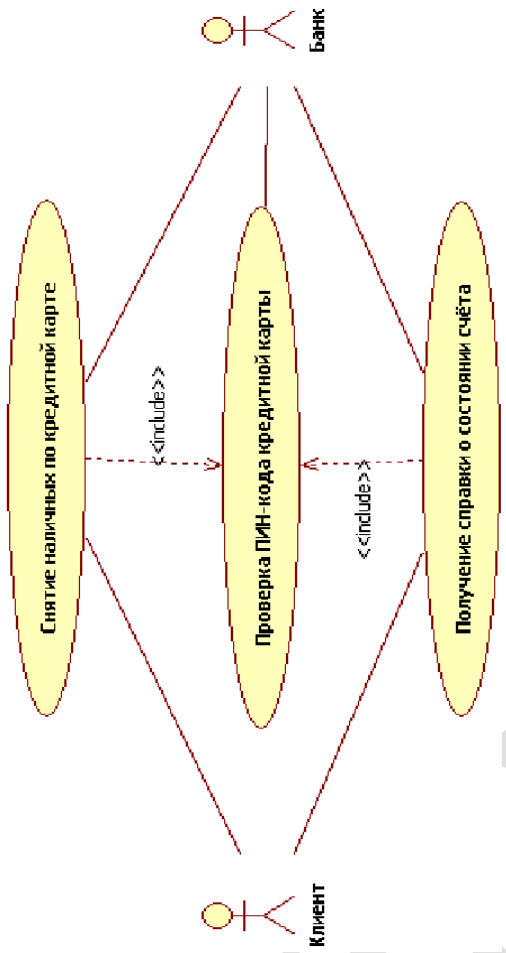

Рис. 9. Диаграмма вариантов использования (общая)

Вышеуказанная диаграмма является учебно-тренировочной и предназначена для лучшего изучения особенностей построения диаграмм вариантов использования.

4. Откройте любой текстовый редактор, в котором есть возможность работы с таблицами и создайте текстовый сценарий, приведённый в таблице 1.
5. Сохраните результаты работы.

Таблица 1

Главный раздел	
Вариант использования	Снятие наличных по кредитной карте
Актёры	Клиент, Банк
Цель	Получение требуемой суммы наличными
Краткое описание	Клиент запрашивает требуемую сумму. Банкомат обеспечивает доступ к счёту клиента. Банкомат выдаёт клиенту наличные
Тип	Базовый

Ссылки на другие варианты использования	Включает в себя варианты использования: • проверка PIN-кода кредитной карточки
Типичный ход событий	
Действия актёров	Отклик системы
1. Клиент вставляет кредитную карточку в устройство чтения банкомата.	2. Банкомат проверяет кредитную карточку. 3. Банкомат предлагает ввести PIN-код. *Исключение №1*: Кредитная карточка недействительна
4. Клиент вводит PIN-код.	5. Банкомат проверяет PIN-код. 6. Банкомат отображает опции меню. *Исключение №2*: Клиент вводит неверный PIN-код
7. Клиент выбирает снятие наличных со своего счёта	8. Система делает запрос в Банк и выясняет текущее состояние счёта клиента. 9. Банкомат предлагает ввести требуемую сумму
10. Клиент вводит требуемую сумму. 11. Банк проверяет введённую сумму. *Исключение №3*: Требуемая сумма превышает сумму на счёте клиента	12. Банкомат изменяет состояние счёта клиента, выдаёт наличные и чек.
13. Клиент получает наличные и чек. 15. Клиент получает свою кредитную карточку.	14. Банкомат предлагает клиенту забрать его кредитную карточку. 16. Банкомат отображает сообщение о своей готовности к работе.
Исключения	
Исключение №1: Кредитная карточка недействительна	
	3. Банкомат отображает информацию о неверно вставленной кредитной карточке. 14. Банкомат возвращает клиенту его кредитную карточку.
Исключение №2: Клиент вводит неверный PIN-код	
4. Клиент вводит новый PIN-код	6. Банкомат отображает информацию о неверном PIN-коде
Исключение №3: Требуемая сумма превышает сумму на счёте клиента	
	12. Банкомат отображает информацию о превышении кредита

10. Клиент вводит новую требуемую сумму	

6. Запустите StarUML и создайте модель. Назовите её Model2.
7. Постройте диаграмму вариантов использования по выбранной теме. Диаграмма должна содержать не менее четырёх вариантов использования.
8. Сохраните результаты работы
9. Откройте любой текстовый редактор, в котором есть возможность работы с таблицами и создайте текстовый сценарий для каждого из базовых вариантов использования. В каждом из текстовых сценариев должно быть не менее трёх исключений. Количество шагов в каждом из текстовых сценариев в разделе «Типичный ход событий» должно быть равно 15 (допускаются небольшие отклонения).
10. Сохраните результаты работы.

Содержание отчёта

1. Титульный лист
2. Цель лабораторной работы
3. Результаты выполнения пунктов 2, 4, 7 и 9. Диаграммы необходимо сохранить в виде рисунков (в StarUML: File → Export Diagram), которые затем вставить в отчёт.
4. Выводы по работе.

Контрольные вопросы

1. Что такое UML? Область его применения?
2. Для чего используется диаграмма вариантов использования?
3. Перечислите и опишите основные элементы диаграммы вариантов использования.
4. Что такое текстовый сценарий?
5. Как изображаются основные элементы диаграммы вариантов использования? Как их добавить на диаграмму в StarUML?

Список литературы

Обязательная

1. Пальмов С.В. Конспект лекций по дисциплине «Методы и средства проектирования информационных систем и технологий».
2. Руководство пользователя для StarUML.

Дополнительная

1. Леоненков А. Самоучитель UML. СПб.: БХВ-Петербург, 2006.

Лабораторная работа №2. Построение диаграммы классов

Цель: Научиться строить диаграммы классов.

Введение

Краткая характеристика диаграммы классов

Центральное место в ООАП занимает разработка логической модели системы в виде диаграммы классов. Нотация классов в языке UML проста и интуитивно понятна всем, кто когда-либо имел опыт работы с CASE-инструментариями. Схожая нотация применяется и для объектов — экземпляров класса, с тем различием, что к имени класса добавляется имя объекта и вся надпись подчеркивается.

Нотация UML предоставляет широкие возможности для отображения дополнительной информации (абстрактные операции и классы, стереотипы, общие и частные методы, детализированные интерфейсы, параметризованные классы). При этом возможно использование графических изображений для ассоциаций и их специфических свойств, таких как отношение агрегации, когда составными частями класса могут выступать другие классы.

Диаграмма классов (class diagram) служит для представления статической структуры модели системы в терминологии классов объектно-ориентированного программирования. Диаграмма классов может отражать, в частности, различные взаимосвязи между отдельными сущностями предметной области, такими как объекты и подсистемы, а также описывает их внутреннюю структуру и типы отношений. На данной диаграмме не указывается информация о временных аспектах функционирования системы. С этой точки зрения диаграмма классов является дальнейшим развитием концептуальной модели проектируемой системы.

Диаграмма классов представляет собой некоторый граф, вершинами которого являются элементы типа "классификатор", которые связаны различными типами структурных отношений. Следует заметить, что диаграмма классов может также содержать интерфейсы, пакеты, отношения и даже отдельные экземпляры, такие как объекты и связи. Когда говорят о данной диаграмме, имеют в виду статическую структурную модель проектируемой системы. Поэтому диаграмму классов принято считать графическим представленном таких структурных взаимосвязей логической модели системы, которые не зависят или инвариантны от времени.

Диаграмма классов состоит из множества элементов, которые в совокупности отражают декларативные знания о предметной области. Эти знания интерпретируются в базовых понятиях языка UML, таких как классы, интерфейсы и отношения между ними и их составляющими компонентами. При этом отдельные компоненты этой диаграммы могут образовывать пакеты для представления более общей модели системы. Если диаграмма классов является ча-

стью некоторого пакета, то ее компоненты должны соответствовать элементам этого пакета, включая возможные ссылки на элементы из других пакетов.
В общем случае пакет статической структурной модели может быть представлен в виде одной или нескольких диаграмм классов. Декомпозиция некоторого представления на отдельные диаграммы выполняется с целью удобства и графической визуализации структурных взаимосвязей предметной области. При этом компоненты диаграммы соответствуют элементам статической семантической модели. Модель системы, в свою очередь, должна быть согласована с внутренней структурой классов, которая описывается на языке UML.

Класс

Класс (class) в языке UML служит для обозначения множества объектов, которые обладают одинаковой структурой, поведением и отношениями с объектами из других классов. Графически класс изображается в виде прямоугольника, который дополнительно может быть разделен горизонтальными линиями на разделы или секции. В этих разделах могут указываться имя класса, атрибуты (переменные) и операции (методы).

Отношения между классами

Кроме внутреннего устройства или структуры классов на соответствующей диаграмме указываются различные отношения между классами. При этом совокупность типов таких отношений фиксирована в языке UML и предопределена семантикой этих типов отношений. Базовыми отношениями или связями в языке UML являются:

- Отношение зависимости (dependency relationship)
- Отношение ассоциации (association relationship)
- Отношение обобщения (generalization relationship)
- Отношение реализации (realization relationship)

Построение диаграмм классов в StarUML

Диаграмма классов (class diagram) добавляется аналогично диаграмме вариантов использования (см. лр.№1).

Добавление элементов диаграммы

Чтобы добавить класс (class) выполните следующие действия: панель Toolbox (слева вверху) → закладка Class → Class. После этого щёлкните ЛКМ по рабочей области в том месте, где будет размещаться данный элемент. Имя класса необходимо задать сразу после его добавления на диаграмму. Отредактировать имя – двойной щелчок ЛКМ по классу

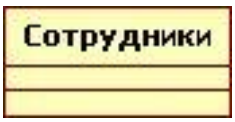

Рис. 10. Добавленный класс

Отношения на диаграмму классов добавляются аналогично диаграмме вариантов использования (см. лр.№1).

Чтобы добавить интерфейс (interface) выполните следующие действия: панель Toolbox (слева вверху) → закладка Class → Interface. После этого щёлкните ЛКМ по рабочей области в том месте, где будет размещаться данный элемент. Имя интерфейса необходимо задать сразу после его добавления на диаграмму. Отредактировать имя – двойной щелчок ЛКМ по интерфейсу.

Домашнее задание студентам для подготовки к выполнению лабораторной работы

Изучить по лекциям и учебной литературе особенности построения диаграмм классов.

Изучить принципы построения диаграмм классов в StarUML (см. руководство пользователя).

Варианты заданий

См. лр№1.

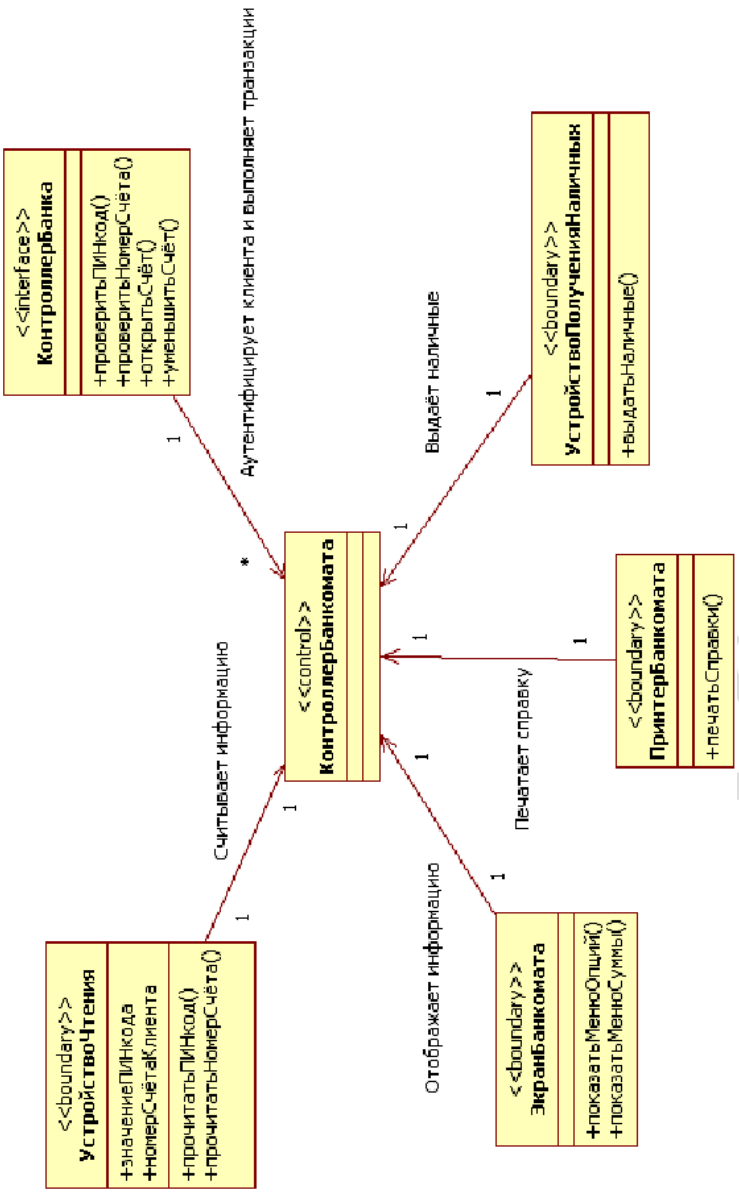

Рис. 11. Диаграмма классов (общая)

Порядок выполнения лабораторной работы

1. Запустите StarUML и откройте файл, который Вы создали в предыдущей работе. Выберите Model1.
2. Постройте диаграмму классов, приведённую на рисунке 11.
3. Сохраните результаты работы.

Вышеуказанная диаграмма является учебно-тренировочной и предназначена для лучшего изучения особенностей построения диаграмм классов.

4. Выберите Model2.
5. Постройте диаграмму классов по выбранной теме. Диаграмма должна содержать не менее пяти классов.
6. Сохраните результаты работы.

Содержание отчёта

1. Титульный лист
2. Цель лабораторной работы
3. Результаты выполнения пунктов 2 и 5. Диаграммы необходимо сохранить в виде рисунков (в StarUML: File → Export Diagram), которые затем вставить в отчёт.
4. Выводы по работе.

Контрольные вопросы

1. Для чего используется диаграмма классов?
2. Перечислите и опишите основные элементы диаграммы классов.
3. Как изображаются основные элементы диаграммы? Как их добавить на диаграмму в StarUML?
4. Напишите шаблон имени атрибута класса.
5. Напишите шаблон имени операции класса.

Список литературы

Обязательная

1. Пальмов С.В. Конспект лекций по дисциплине «Методы и средства проектирования информационных систем и технологий».
2. Руководство пользователя для StarUML.

Дополнительная

1. Леоненков А. Самоучитель UML. СПб.: БХВ-Петербург, 2006.

Лабораторная работа №3. Построение диаграммы последовательности

Цель: Научиться строить диаграммы последовательности.

Введение

Краткая характеристика диаграммы последовательности

Одной из характерных особенностей систем различной природы и назначения является взаимодействие между собой отдельных элементов, из которых образованы эти системы. Речь идет о том, что различные составные элементы систем не существуют изолированно, а оказывают 'определенное влияние друг на друга, что и отличает систему как целостное образование от простой совокупности элементов.

В языке UML взаимодействие элементов рассматривается в информационном аспекте их коммуникации, т. е. взаимодействующие объекты обмениваются между собой некоторой информацией. При этом информация принимает форму законченных сообщений. Другими словами, хотя сообщение и имеет информационное содержание, оно приобретает дополнительное свойство оказывать направленное влияние на своего получателя. Это полностью согласуется с принципами ООАП, когда любые виды информационного взаимодействия между элементами системы должны быть сведены к отправке и приему сообщений между ними.

Для моделирования взаимодействия объектов в языке UML используются соответствующие диаграммы взаимодействия. Говоря об этих диаграммах, имеют в виду два аспекта взаимодействия. Во-первых, взаимодействия объектов можно рассматривать во времени, и тогда для представления временных особенностей передачи и приема сообщений между объектами используется диаграмма последовательности. Этот вид канонических диаграмм является предметом изучения настоящей лекции.

Временной аспект поведения может иметь существенное значение при моделировании синхронных процессов, описывающих взаимодействия объектов. Именно для этой цели в языке UML используются диаграммы последовательности.

Во-вторых, можно рассматривать структурные особенности взаимодействия объектов. Для представления структурных особенностей передачи и приема сообщений между объектами используется диаграмма кооперации.

Объекты

На диаграмме последовательности изображаются исключительно те объекты, которые непосредственно участвуют во взаимодействии и не показываются возможные статические ассоциации с другими объектами. Для диаграммы последовательности ключевым моментом является именно динамика взаимодействия объектов во времени. При этом диаграмма последовательности имеет

как бы два измерения. Одно — слева направо в виде вертикальных линий, каждая из которых изображает линию жизни отдельного объекта, участвующего во взаимодействии. Графически каждый объект изображается прямоугольником и располагается в верхней части своей линии жизни. Внутри прямоугольника записываются имя объекта и имя класса, разделенные двоеточием. При этом вся запись подчеркивается, что является признаком объекта, который, как известно, представляет собой экземпляр класса.

Построение диаграмм последовательности в StarUML

Диаграмма последовательности (sequence diagram) добавляется аналогично диаграмме вариантов использования (см. лр.№1).

Добавление элементов диаграммы

Чтобы добавить объект (object) выполните следующие действия: панель Toolbox (слева вверху) → закладка Sequence → Object. После этого щёлкните ЛКМ по рабочей области в том месте, где будет размещаться данный элемент. Имя объекта необходимо задать сразу после его добавления на диаграмму. Отредактировать имя – двойной щелчок ЛКМ по объекту.

Рис. 12. Добавленный объект

Чтобы добавить сообщение (stimulus) на диаграмму выполните следующие действия: панель Toolbox (слева вверху) → закладка Sequence → Stimulus. После этого щёлкните ЛКМ по рабочей области в том месте, где будет размещаться данный элемент. Имя сообщения необходимо задать сразу после его добавления на диаграмму. Отредактировать имя – двойной щелчок ЛКМ по объекту.

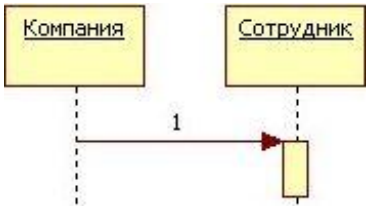

Рис. 13. Добавленные объекты и сообщение

Домашнее задание студентам для подготовки к выполнению лабораторной работы

Изучить по лекциям и учебной литературе особенности построения диаграмм последовательности.

Изучить принципы построения диаграмм последовательности в StarUML (см. руководство пользователя).

Варианты заданий

См. лр.№1.

Порядок выполнения лабораторной работы

1. Запустите StarUML и откройте файл, который Вы создали в предыдущей работе. Выберите Model1.
2. Постройте диаграмму последовательности, приведённую на рисунке 14.
3. Сохраните результаты работы.
4. Выберите Model2.
5. Постройте диаграмму последовательности для типичного хода событий (см. текстовые сценарии вариантов использования из лр.№1) по выбранной теме для каждого варианта использования (один вариант использования – одна диаграмма для типичного хода событий!).
6. Постройте диаграммы последовательности для каждого исключения (см. текстовые сценарии вариантов использования из лр.№1).
7. Сохраните результаты работы.

Содержание отчёта

1. Титульный лист
2. Цель лабораторной работы
3. Результаты выполнения пунктов 2, 5 и 6. Диаграммы необходимо сохранить в виде рисунков (в StarUML: File → Export Diagram), которые затем вставить в отчёт.
4. Выводы по работе.

Контрольные вопросы

1. Для чего используется диаграмма последовательности?
2. Перечислите и опишите основные элементы диаграммы последовательности.
3. Как изображаются основные элементы диаграммы? Как их добавить на диаграмму в StarUML?
4. В чём отличие класса от экземпляра класса?
5. Каким образом определено понятие времени в диаграмме последовательности?

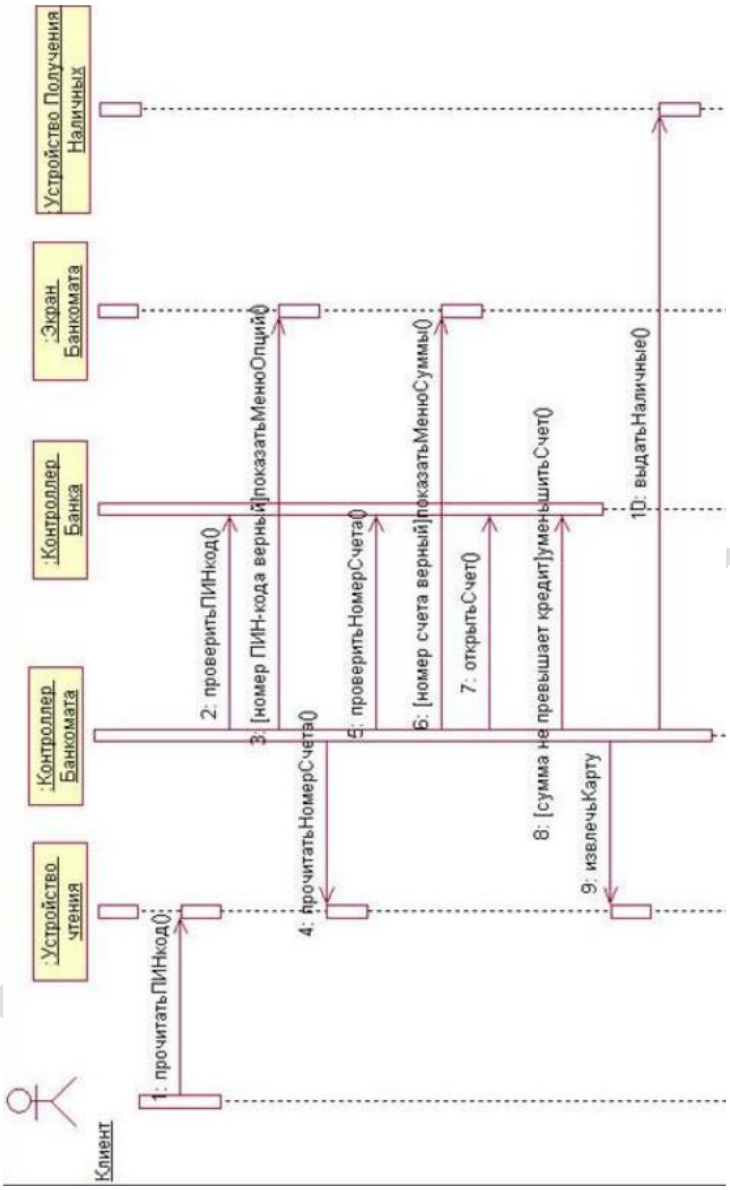

Рис. 14. Диаграмма последовательности (общая)

Список литературы

Обязательная

1. Пальмов С.В. Конспект лекций по дисциплине «Методы и средства проектирования информационных систем и технологий».
2. Руководство пользователя для StarUML.

Дополнительная

1. Леоненков А. Самоучитель UML. СПб.: БХВ-Петербург, 2006.

Лабораторная работа №4. Построение диаграммы кооперации

Цель: Научиться строить диаграммы кооперации.

Введение

Краткая характеристика диаграммы кооперации

Как отмечалось ранее, особенности взаимодействия элементов моделируемой системы могут быть представлены на диаграммах последовательности и кооперации. Если первая служит для визуализации временных аспектов взаимодействия, то 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.

Прежде всего, на диаграмме кооперации в виде прямоугольников изображаются участвующие во взаимодействии объекты, содержащие имя объекта, его класс и, возможно, значения атрибутов. Далее, как и на диаграмме классов, 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.

В отличие от диаграммы последовательности, на диаграмме кооперации изображаются только отношения между объектами, играющими определенные роли во взаимодействии. С другой стороны, на этой диаграмме не указывается время в виде отдельного измерения. Поэтому последовательность взаимодействий и параллельных потоков может быть определена с помощью порядковых номеров. Следовательно, если необходимо явно специфицировать взаимосвязи между объектами в реальном времени, лучше это делать на диаграмме последовательности.

Поведение системы может описываться на уровне отдельных объектов, которые обмениваются между собой сообщениями, чтобы достичь нужной цели или реализовать некоторый сервис. С точки зрения аналитика или конструктора важно представить в проекте системы структурные связи отдельных объектов между собой. Такое статическое представление структуры системы как совокупности взаимодействующих объектов и обеспечивает диаграмма кооперации.

Таким образом, с помощью диаграммы кооперации можно описать полный контекст взаимодействий как своеобразный временной "среза" совокупности объектов, взаимодействующих между собой для выполнения определенной задачи или бизнес-цели программной системы.

Кооперация

Понятие кооперации (collaboration) является одним из фундаментальных понятий в языке UML. Оно служит для обозначения множества взаимодействующих с определенной целью объектов в общем контексте моделируемой системы. Цель самой кооперации состоит в том, чтобы специфицировать особенности реализации отдельных наиболее значимых операций в системе. Кооперация определяет структуру поведения системы в терминах взаимодействия участников этой кооперации.

Кооперация может быть представлена на двух уровнях:

- На уровне спецификации — показывает роли классификаторов и роли ассоциаций в рассматриваемом взаимодействии.
- На уровне примеров — указывает экземпляры и связи, образующие отдельные роли в кооперации.

Объекты

Объект (object) является отдельным экземпляром класса, который создается на этапе выполнения программы. Он может иметь свое собственное имя и конкретные значения атрибутов.

Связи

Связь (link) является экземпляром или примером произвольной ассоциации. Связь как элемент языка UML может иметь место между двумя и более объектами. Бинарная связь на диаграмме кооперации изображается отрезком прямой линии, соединяющей два прямоугольника объектов. На каждом из концов этой линии могут быть явно указаны имена ролей данной ассоциации. Рядом с линией в ее средней части может записываться имя соответствующей ассоциации.

Связи не имеют собственных имен, поскольку полностью идентичны как экземпляры ассоциации. Другими словами, все связи на диаграмме кооперации могут быть только анонимными и записываются без двоеточия перед именем ассоциации. Для связей не указывается также и кратность. Однако другие обозначения специальных случаев ассоциации (агрегация, композиция) могут присутствовать на отдельных концах связей.

Сообщения

При построении диаграммы кооперации они имеют некоторые дополнительные семантические особенности. Сообщение на диаграмме кооперации специфицирует коммуникацию между двумя объектами, один из которых передает другому некоторую информацию. При этом первый объект ожидает, что после получения сообщения вторым объектом последует выполнение некоторо-

го действия. Таким образом, именно сообщение является причиной или стимулом для начала выполнения операций, отправки сигналов, создания и уничтожения отдельных объектов. Связь обеспечивает канал для направленной передачи сообщений между объектами от объекта-источника к объекту-получателю.

Построение диаграмм кооперации в StarUML

Диаграмма кооперации (collaboration diagram) добавляется аналогично диаграмме вариантов использования (см. лр.№1).

Добавление элементов диаграммы

Чтобы добавить объект (object) выполните следующие действия: панель Toolbox (слева вверху) → закладка Collaboration → Object. После этого щёлкните ЛКМ по рабочей области в том месте, где будет размещаться данный элемент. Имя объекта необходимо задать сразу после его добавления на диаграмму. Отредактировать имя – двойной щелчок ЛКМ по объекту.

Чтобы добавить связь (link) на диаграмму выполните следующие действия: панель Toolbox (слева вверху) → закладка Collaboration → Link. После этого щёлкните ЛКМ по рабочей области в том месте, где будет размещаться данный элемент. Имя сообщения необходимо задать сразу после его добавления на диаграмму. Отредактировать имя – двойной щелчок ЛКМ по связи.

Чтобы добавить сообщение (stimulus) на диаграмму выполните следующие действия: панель Toolbox (слева вверху) → закладка Collaboration → ForwardStimulus (или ReverseStimulus). После этого щёлкните ЛКМ по рабочей области в том месте, где будет размещаться данный элемент. Имя сообщения необходимо задать сразу после его добавления на диаграмму. Отредактировать имя – двойной щелчок ЛКМ по сообщению.

Рис. 15. Добавленные объекты, связь и сообщение

Домашнее задание студентам для подготовки к выполнению лабораторной работы

Изучить по лекциям и учебной литературе особенности построения диаграмм кооперации.

Изучить принципы построения диаграмм кооперации в StarUML (см. руководство пользователя).

Варианты заданий

См. лр.№1.

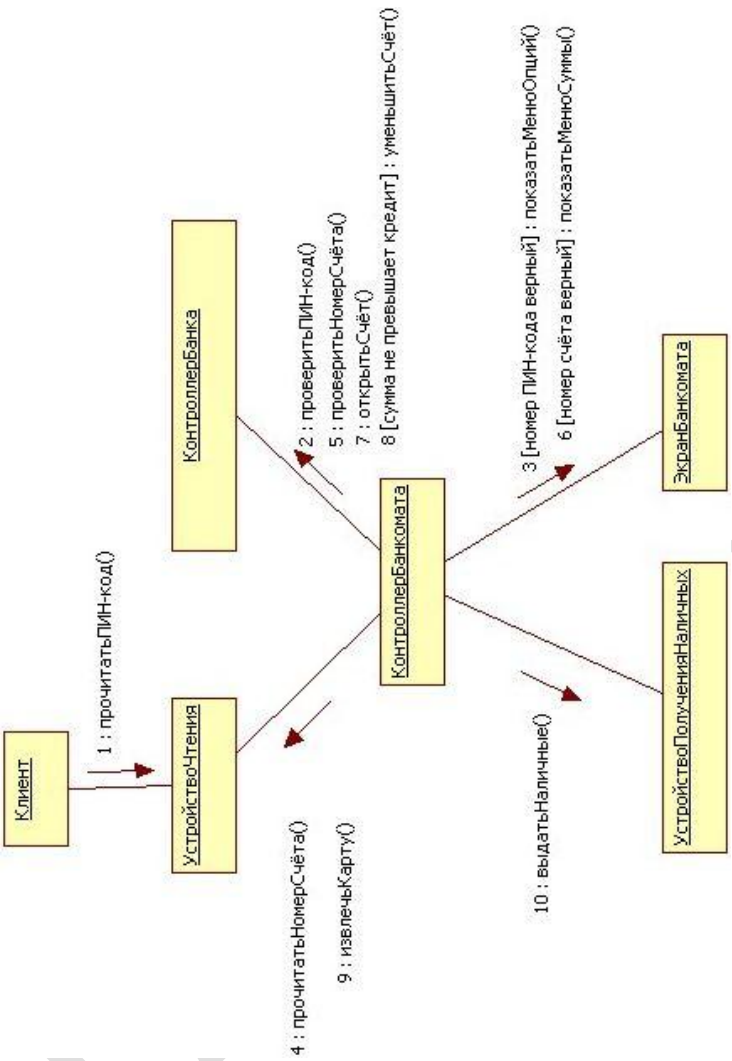

Рис. 16. Диаграмма кооперации (общая)

Диаграмма, приведённая на рис. 16, является учебно-тренировочной и предназначена для лучшего изучения особенностей построения диаграмм кооперации.

Порядок выполнения лабораторной работы

1. Запустите StarUML и откройте файл, который Вы создали в предыдущей работе. Выберите Model1.
2. Постройте диаграмму кооперации, приведённую на рисунке 16.
3. Сохраните результаты работы.
4. Выберите Model2.
5. Постройте диаграмму кооперации для типичного хода событий (см. текстовые сценарии вариантов использования из лр.№1) по выбранной теме для каждого варианта использования (один вариант использования – одна диаграмма для типичного хода событий!).
6. Постройте диаграммы кооперации для каждого исключения (см. текстовые сценарии вариантов использования из лр.№1).
7. Сохраните результаты работы.

Содержание отчёта

1. Титульный лист
2. Цель лабораторной работы
3. Результаты выполнения пунктов 2, 5 и 6. Диаграммы необходимо сохранить в виде рисунков (в StarUML: File → Export Diagram), которые затем вставить в отчёт.
4. Выводы по работе.

Контрольные вопросы

1. Для чего используется диаграмма кооперации?
2. Перечислите и опишите основные элементы диаграммы кооперации.
3. Как изображаются основные элементы диаграммы? Как их добавить на диаграмму в StarUML?
4. В чём отличие класса от экземпляра класса?

Список литературы

Обязательная

1. Пальмов С.В. Конспект лекций по дисциплине «Методы и средства проектирования информационных систем и технологий».
2. Руководство пользователя для StarUML.

Дополнительная

1. Леоненков А. Самоучитель UML. СПб.: БХВ-Петербург, 2006.

Лабораторная работа №5. Построение диаграммы состояний

Цель: Научиться строить диаграммы состояний.

Введение

Краткая характеристика диаграммы состояний

Для моделирования поведения на логическом уровне в UML могут использоваться сразу несколько канонических диаграмм: состояний, деятельности, последовательности и кооперации, каждая из которых фиксирует внимание на отдельном аспекте функционирования системы. В отличие от других диаграмм диаграмма состояний описывает процесс изменения состояний только одного класса, а точнее — одного экземпляра определенного класса, т. е. моделирует все возможные изменения в состоянии конкретного объекта. При этом изменение состояния объекта может быть вызвано внешними воздействиями со стороны других объектов или извне. Именно для описания реакции объекта на подобные внешние воздействия и используются диаграммы состояний.

Главное предназначение этой диаграммы — описать возможные последовательности состояний и переходов, которые в совокупности характеризуют поведение элемента модели в течение его жизненного цикла. Диаграмма состояний представляет динамическое поведение сущностей, на основе спецификации их реакции на восприятие некоторых конкретных событий. Системы, которые реагируют на внешние действия от других систем или от пользователей, иногда называют реактивными. Если такие действия инициируются в произвольные случайные моменты времени, то говорят об асинхронном поведении модели.

Хотя диаграммы состояний чаще всего используются для описания поведения отдельных экземпляров классов (объектов), но они также могут быть применены для спецификации функциональности других компонентов моделей, таких как варианты использования, актеры, подсистемы, операции и методы.

Диаграмма состояний по существу является графом специального вида, который представляет некоторый автомат. Понятие автомата в контексте UML обладает довольно специфической семантикой, основанной на теории автоматов. Вершинами этого графа являются состояния и некоторые другие типы элементов автомата (псевдосостояния), которые изображаются соответствующими графическими символами. Дуги графа служат для обозначения переходов из состояния в состояние. Диаграммы состояний могут быть вложены друг в друга, образуя вложенные диаграммы более детального представления отдельных элементов модели. Для понимания семантики конкретной диаграммы состояний необходимо представлять не только особенности поведения моделируемой сущности, но и знать общие сведения по теории автоматов.

Автоматы

Автомат (state machine) в UML представляет собой некоторый формализм для моделирования поведения элементов модели и системы в целом. В метамодели UML автомат является пакетом, в котором определено множество понятий, необходимых для представления поведения моделируемой сущности в виде дискретного пространства с конечным числом состояний и переходов. С другой стороны, автомат описывает поведение отдельного объекта в форме последовательности состояний, которые охватывают все этапы его жизненного цикла, начиная от создания объекта и заканчивая его уничтожением. Каждая диаграмма состояний представляет некоторый автомат.

Простейшим примером визуального представления состояний и переходов на основе формализма автоматов может служить рассмотренная выше ситуация с исправностью технического устройства, такого как компьютер. В этом случае вводятся в рассмотрение два самых общих состояния: "исправен" и "неисправен" и два перехода: "выход из строя" и "ремонт". Графически эта информация может быть представлена в виде изображенной ниже диаграммы состояний компьютера.

Рис. 17. Простейший пример диаграммы состояний для технического устройства типа компьютер

Состояние

Понятие состояния (state) является фундаментальным не только в метамодели языка UML, но и в прикладном системном анализе.

Состояние может быть задано в виде набора конкретных значений атрибутов класса или объекта, при этом изменение их отдельных значений будет отражать изменение состояния моделируемого класса или объекта.

Переход

Простой переход (simple transition) представляет собой отношение между двумя последовательными состояниями, которое указывает на факт смены одного состояния другим. Пребывание моделируемого объекта в первом состоянии может сопровождаться выполнением некоторых действий, а переход во второе состояние будет возможен после завершения этих действий, а также после удовлетворения некоторых дополнительных условий. В этом случае говорят, что переход срабатывает, Или происходит срабатывание перехода. До срабатывания перехода объект находится в предыдущем от него состоянии, называемым исходным состоянием, или в источнике (не путать с начальным состоя-

нием — это разные понятия), а после его срабатывания объект находится в последующем от него состоянии (целевом состоянии).

Событие

Формально, событие представляет собой спецификацию некоторого факта, имеющего место в пространстве и во времени. Про события говорят, что они "происходят", при этом отдельные события должны быть упорядочены во времени. После наступления некоторого события нельзя уже вернуться к предыдущим событиям, если такая возможность не предусмотрена явно в модели.

Сторожевое условие

Сторожевое условие (guard condition), если оно есть, всегда записывается в прямых скобках после события-триггера и представляет собой некоторое булевское выражение. Из контекста диаграммы состояний должна явно следовать семантика этого выражения, а для записи выражения может использоваться синтаксис языка объектных ограничений, основы которого изложены в приложении.

Построение диаграмм состояния в StarUML

Диаграмма состояния (statechart diagram) добавляется аналогично диаграмме вариантов использования (см. лр.№1).

Добавление элементов диаграммы

Чтобы добавить состояние (state) выполните следующие действия: панель Toolbox (слева вверху) → закладка Statechart → State. После этого щёлкните ЛКМ по рабочей области в том месте, где будет размещаться данный элемент. Имя состояния необходимо задать сразу после его добавления на диаграмму. Отредактировать имя – двойной щелчок ЛКМ по состоянию.

Чтобы добавить переход (transition) на диаграмму выполните следующие действия: панель Toolbox (слева вверху) → закладка Statechart → Transition. После этого щёлкните ЛКМ по рабочей области в том месте, где будет размещаться данный элемент. Имя перехода необходимо задать сразу после его добавления на диаграмму. Отредактировать имя – двойной щелчок ЛКМ по переходу.

Рис. 18. Добавленные состояния и переход

Домашнее задание студентам для подготовки к выполнению лабораторной работы

Изучить по лекциям и учебной литературе особенности построения диаграмм состояний.

Изучить принципы построения диаграмм состояний в StarUML (см. руководство пользователя).

Варианты заданий

См. лр.№1.

Порядок выполнения лабораторной работы

1. Запустите StarUML и откройте файл, который Вы создали в предыдущей работе. Выберите Model1.
2. Постройте диаграмму состояний, приведённую на рисунке 19.
3. Сохраните результаты работы.
4. Выберите Model2.
5. Постройте диаграмму состояний по выбранной теме.
6. Сохраните результаты работы.

Содержание отчёта

1. Титульный лист
2. Цель лабораторной работы
3. Результаты выполнения пунктов 2 и 5. Диаграммы необходимо сохранить в виде рисунков (в StarUML: File → Export Diagram), которые затем вставить в отчёт.
4. Выводы по работе.

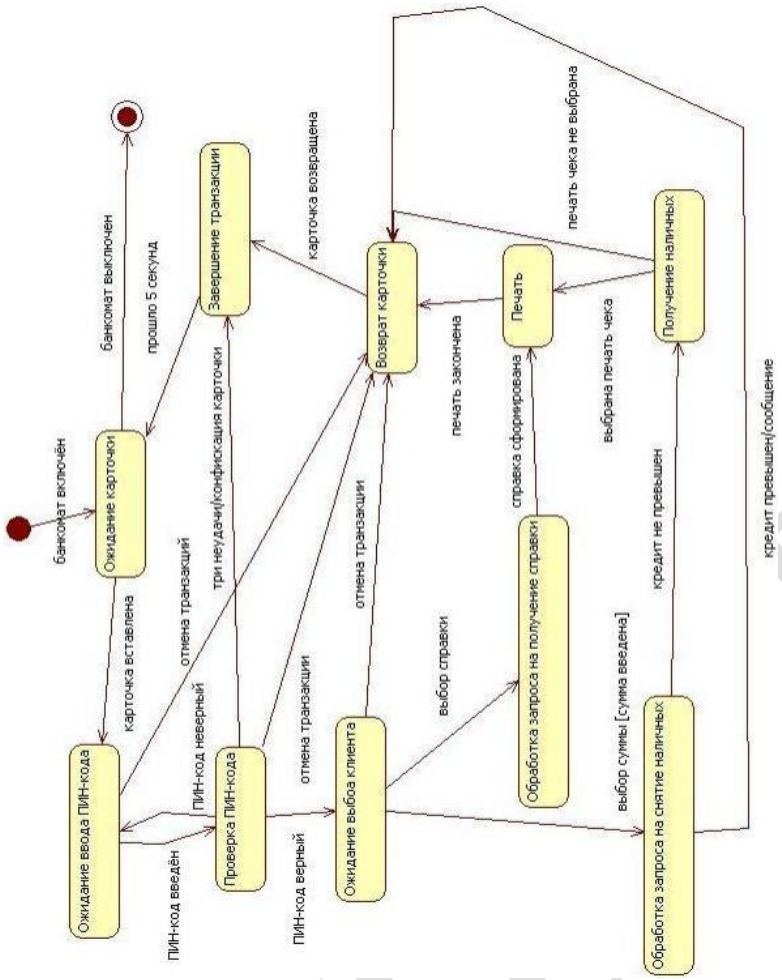

Рис. 19. Диаграмма состояний (общая)

Вышеуказанная диаграмма является учебно-тренировочной и предназначена для лучшего изучения особенностей построения диаграмм состояний.

Контрольные вопросы

1. Для чего используется диаграмма состояний?
2. Перечислите и опишите основные элементы диаграммы состояний.
3. Как изображаются основные элементы диаграммы? Как их добавить на диаграмму в StarUML?
4. Сколько может быть на диаграмме начальных состояний? Сколько конечных?
5. Какие виды состояний Вы знаете?

Список литературы

Обязательная

1. Пальмов С.В. Конспект лекций по дисциплине «Методы и средства проектирования информационных систем и технологий».

2. Руководство пользователя для StarUML.

Дополнительная

1. Леоненков А. Самоучитель UML. СПб.: БХВ-Петербург, 2006.

Лабораторная работа №6. Построение диаграммы деятельности

Цель: Научиться строить диаграммы деятельности.

Введение

Краткая характеристика диаграммы деятельности

Для моделирования процесса выполнения операций в UML используются так называемые диаграммы деятельности. Применяемая в них графическая нотация во многом похожа на нотацию диаграммы состояний, поскольку на диаграммах деятельности также присутствуют обозначения состояний и переходов. Отличие заключается в семантике состояний, которые используются для представления не деятельностей, а действий, и в отсутствии на переходах сигнатуры событий. Каждое состояние на диаграмме деятельности соответствует выполнению некоторой элементарной операции, а переход в следующее состояние срабатывает только при завершении этой, операции в предыдущем состоянии. Графически диаграмма деятельности представляется в форме графа деятельности, вершинами которого являются состояния действия, а дугами — переходы от одного состояния действия к другому.

Таким образом, диаграммы деятельности можно считать частным случаем диаграмм состояний. Именно они позволяют реализовать в UML особенности процедурного и синхронного управления, обусловленного завершением внутренних деятельностей и действий. Метамодель UML предоставляет для этого необходимые термины и семантику. Основным направлением использования диаграмм деятельности является визуализация особенностей реализации операций классов, когда необходимо представить алгоритмы их выполнения. При этом каждое состояние может являться выполнением операции некоторого класса либо ее части, позволяя использовать диаграммы деятельности для описания реакций на внутренние события системы.

В контексте языка UML деятельность (activity) представляет собой некоторую совокупность отдельных вычислений, выполняемых автоматом. При этом отдельные элементарные вычисления могут приводить к некоторому результату или действию (action). На диаграмме деятельности отображается логика или последовательность перехода от одной деятельности к другой, при этом внимание фиксируется на результате деятельности. Сам же результат может привести к изменению состояния системы или возвращению некоторого значения.

Состояние действия

Состояние действия (action state) является специальным случаем состояния с некоторым входным действием и, по крайней мере, одним выходящим из состояния переходом. Этот переход неявно предполагает, что входное действие уже завершилось. Состояние действия не может иметь внутренних переходов, поскольку оно является элементарным. Обычное использование состояния дей-

ствия заключается в моделировании одного шага выполнения алгоритма (процедуры) или потока управления.

Переходы

При построении диаграммы деятельности используются только нетриггерные переходы, т. е. такие, которые срабатывают сразу после завершения деятельности или выполнения соответствующего действия. Этот переход переводит деятельность в последующее состояние сразу, как только закончится действие в предыдущем состоянии. На диаграмме такой переход изображается сплошной линией со стрелкой.

Построение диаграмм деятельности в StarUML

Диаграмма деятельности (activity diagram) добавляется аналогично диаграмме вариантов использования (см. лр.№1).

Добавление элементов диаграммы

Чтобы добавить состояние действия (ActionState) выполните следующие действия: панель Toolbox (слева вверху) → закладка Activity → ActionState. После этого щёлкните ЛКМ по рабочей области в том месте, где будет размещаться данный элемент. Имя состояния действия необходимо задать сразу после его добавления на диаграмму. Отредактировать имя – двойной щелчок ЛКМ по состоянию действия.

Чтобы добавить переход (transition) на диаграмму выполните следующие действия: панель Toolbox (слева вверху) → закладка Activity → Transition. После этого щёлкните ЛКМ по рабочей области в том месте, где будет размещаться данный элемент.

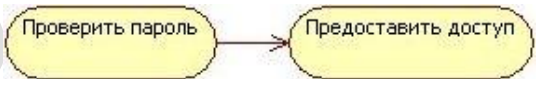

Рис. 20. Добавленные состояния действия и переход

Домашнее задание студентам для подготовки к выполнению лабораторной работы

Изучить по лекциям и учебной литературе особенности построения диаграмм деятельности.

Изучить принципы построения диаграмм деятельности в StarUML (см. руководство пользователя).

Варианты заданий

См. лр.№1.

Порядок выполнения лабораторной работы

1. Запустите StarUML и откройте файл, который Вы создали в предыдущей работе. Выберите Model1.
2. Постройте диаграмму деятельности, приведённую на рисунке 21.
3. Сохраните результаты работы.
4. Выберите Model2.
5. Постройте диаграмму деятельности по выбранной теме.
6. Сохраните результаты работы.

Содержание отчёта

1. Титульный лист
2. Цель лабораторной работы
3. Результаты выполнения пунктов 2 и 5. Диаграммы необходимо сохранить в виде рисунков (в StarUML: File → Export Diagram), которые затем вставить в отчёт.
4. Выводы по работе.

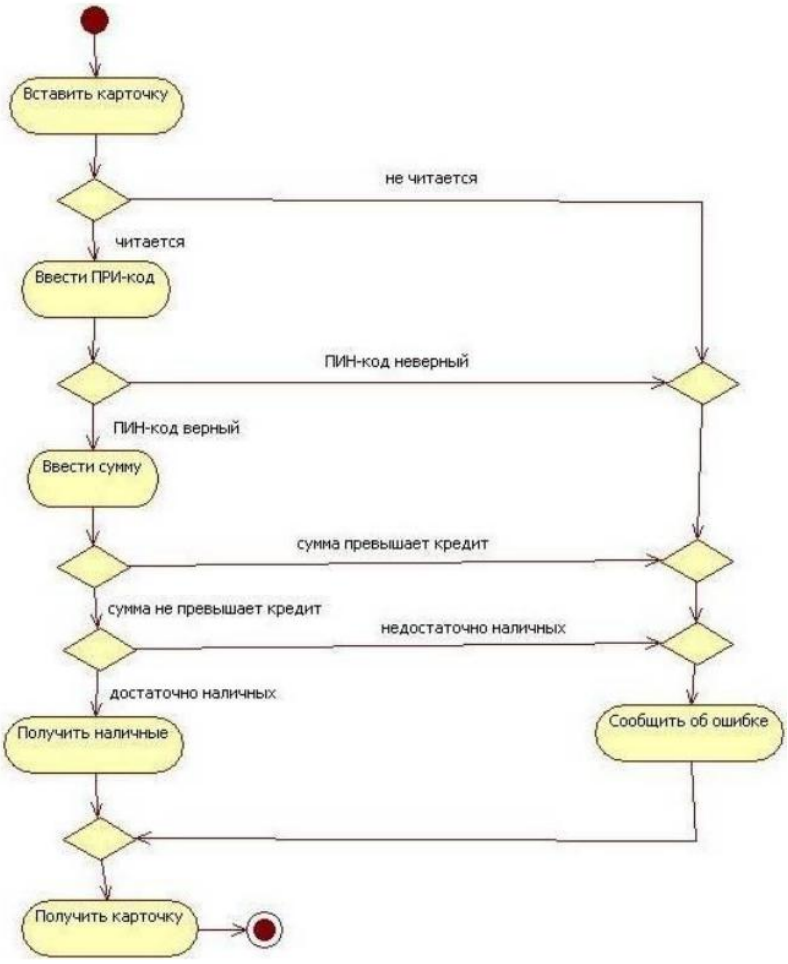

Рис. 21. Диаграмма деятельности (общая)

Вышеуказанная диаграмма является учебно-тренировочной и предназначена для лучшего изучения особенностей построения диаграмм деятельности.

Контрольные вопросы

1. Для чего используется диаграмма деятельности?
2. Перечислите и опишите основные элементы диаграммы деятельности.
3. Как изображаются основные элементы диаграммы? Как их добавить на диаграмму в StarUML?
4. Сколько может быть на диаграмме начальных состояний? Сколько конечных?
5. Какие виды состояний деятельности Вы знаете?

Список литературы

Обязательная

1. Пальмов С.В. Конспект лекций по дисциплине «Методы и средства проектирования информационных систем и технологий».
2. Руководство пользователя для StarUML.

Дополнительная

1. Леоненков А. Самоучитель UML. СПб.: БХВ-Петербург, 2006.

Лабораторная работа №7. Построение диаграммы компонентов

Цель: Научиться строить диаграммы компонентов.

Введение

Краткая характеристика диаграммы компонентов

Диаграмма компонентов, в отличие от ранее рассмотренных диаграмм, описывает особенности физического представления системы. Диаграмма компонентов позволяет определить архитектуру разрабатываемой системы, установив зависимости между программными компонентами, в роли которых может выступать исходный, бинарный и исполняемый код. Во многих средах разработки модуль или компонент соответствует файлу. Пунктирные стрелки, соединяющие модули, показывают отношения взаимозависимости, аналогичные тем, которые имеют место при компиляции исходных текстов программ. Основными графическими элементами диаграммы компонентов являются компоненты, интерфейсы и зависимости между ними.

Диаграмма компонентов разрабатывается для следующих целей:

- Визуализации общей структуры исходного кода программной системы.
- Спецификации исполнимого варианта программной системы.
- Обеспечения многократного использования отдельных фрагментов программного кода.
- Представления концептуальной и физической схем баз данных.

В разработке диаграмм компонентов участвуют как системные аналитики и архитекторы, так и программисты. Диаграмма компонентов обеспечивает согласованный переход от логического представления к конкретной реализации проекта в форме программного кода. Одни компоненты могут существовать только на этапе компиляции программного кода, другие — на этапе его исполнения. Диаграмма компонентов отражает общие зависимости между компонентами, рассматривая последние в качестве классификаторов.

Компоненты

Для представления физических сущностей в языке UML применяется специальный термин — компонент (component). Компонент реализует некоторый набор интерфейсов и служит для общего обозначения элементов физического представления модели. Для графического представления компонента может использоваться специальный символ — прямоугольник со вставленными слева двумя более мелкими прямоугольниками. Внутри объемлющего прямоугольника записывается имя компонента и, возможно, некоторая дополнительная информация. Изображение этого символа может незначительно варьироваться в зависимости от характера ассоциируемой с компонентом информации.

Зависимости

В общем случае отношение зависимости также было рассмотрено ранее. Напомним, что зависимость не является ассоциацией, а служит для представления только факта наличия такой связи, когда изменение одного элемента модели оказывает влияние или приводит к изменению другого элемента модели. Отношение зависимости на диаграмме компонентов изображается пунктирной линией со стрелкой, направленной от клиента (зависимого элемента) к источнику (независимому элементу).

Построение диаграмм компонентов в StarUML

Диаграмма компонентов (component diagram) добавляется аналогично диаграмме вариантов использования (см. лр.№1).

Добавление элементов диаграммы

Чтобы добавить компонент (component) выполните следующие действия: панель Toolbox (слева вверху) → закладка Component → Component. После этого щёлкните ЛКМ по рабочей области в том месте, где будет размещаться данный элемент. Имя компонента необходимо задать сразу после его добавления на диаграмму. Отредактировать имя – двойной щелчок ЛКМ по компоненту.

Чтобы добавить отношение на диаграмму компонентов выполните следующие действия: панель Toolbox (слева вверху) → закладка Component → (нужный тип отношения) (см. рисунок 22).

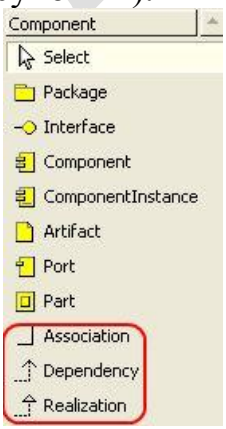

Рис. 22. Отношения на диаграмме компонентов

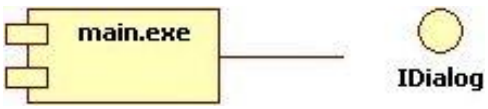

Рис. 23. Добавленный компонент, отношение и интерфейс.

Домашнее задание студентам для подготовки к выполнению лабораторной работы

Изучить по лекциям и учебной литературе особенности построения диаграмм компонентов.

Изучить принципы построения диаграмм компонентов в StarUML (см. руководство пользователя).

Варианты заданий

См. лр.№1.

Порядок выполнения лабораторной работы

1. Запустите StarUML и откройте файл, который Вы создали в предыдущей работе. Выберите Model1.
2. Постройте диаграмму компонентов, приведённую на рисунке 24.
3. Сохраните результаты работы.
4. Выберите Model2.
5. Постройте диаграмму компонентов по выбранной теме.
6. Сохраните результаты работы.

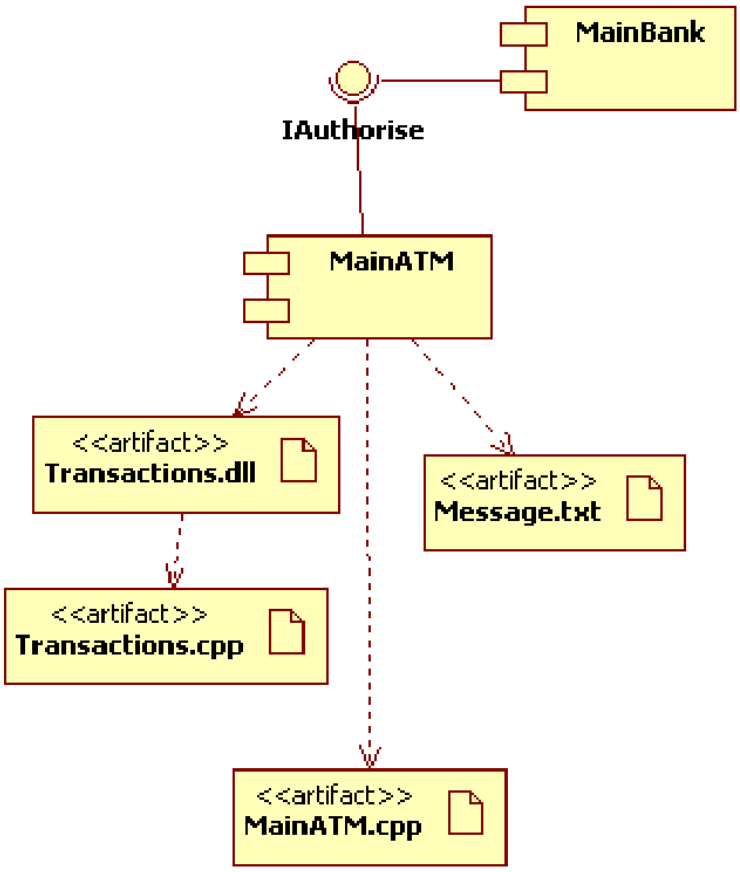

Рис. 24. Диаграмма компонентов (общая)

Вышеуказанная диаграмма является учебно-тренировочной и предназначена для лучшего изучения особенностей построения диаграмм компонентов.

Содержание отчёта

1. Титульный лист
2. Цель лабораторной работы
3. Результаты выполнения пунктов 2 и 5. Диаграммы необходимо сохранить в виде рисунков (в StarUML: File → Export Diagram), которые затем вставить в отчёт.
4. Выводы по работе.

Контрольные вопросы

1. Для чего используется диаграмма компонентов?
2. Перечислите и опишите основные элементы диаграммы компонентов.
3. Как изображаются основные элементы диаграммы? Как их добавить на диаграмму в StarUML?
4. Приведите примеры расширений файлов? Расшифруйте их.
5. Какие виды компонентов Вы знаете?

Список литературы

Обязательная

1. Пальмов С.В. Конспект лекций по дисциплине «Методы и средства проектирования информационных систем и технологий».
2. Руководство пользователя для StarUML.

Дополнительная

1. Леоненков А. Самоучитель UML. СПб.: БХВ-Петербург, 2006.

Лабораторная работа №8. Построение диаграммы развёртывания

Цель: Научиться строить диаграммы развёртывания.

Введение

Краткая характеристика диаграммы развёртывания

Как было отмечено ранее, первой из диаграмм физического представления является диаграмма компонентов. Второй формой физического представления программной системы является диаграмма развертывания (синоним — диаграмма размещения). Она применяется для представления общей конфигурации и топологии распределенной программной системы и содержит распределение компонентов по отдельным узлам системы. Кроме того, диаграмма развертывания показывает наличие физических соединений — маршрутов передачи информации между аппаратными устройствами, задействованными в реализации системы.

Диаграмма развертывания предназначена для визуализации элементов и компонентов программы, существующих лишь на этапе ее исполнения (runtime). При этом представляются только компоненты-экземпляры программы, являющиеся исполнимыми файлами или динамическими библиотеками. Те компоненты, которые не используются на этапе исполнения, на диаграмме развертывания не показываются. Так, компоненты с исходными текстами программ могут присутствовать только на диаграмме компонентов. На диаграмме развертывания они не указываются.

Диаграмма развертывания содержит графические изображения процессоров, устройств, процессов и связей между ними. В отличие от диаграмм логического представления, диаграмма развертывания является единой для системы в целом, поскольку должна всецело отражать особенности ее реализации. Эта диаграмма, по сути, завершает процесс ООАП для конкретной программной системы и ее разработка, как правило, является последним этапом спецификации модели.

Итак, перечислим цели, преследуемые при разработке диаграммы развертывания:

- Определить распределение компонентов системы по ее физическим узлам.
- Показать физические связи между всеми узлами реализации системы на этапе ее исполнения.
- Выявить узкие места системы и реконфигурировать ее топологию для достижения требуемой производительности.

Для обеспечения этих требований диаграмма развертывания разрабатывается совместно системными аналитиками, сетевыми инженерами и системотех-

никами. Далее рассмотрим отдельные элементы, из которых состоят диаграммы развертывания.

Узел

Узел (node) представляет собой некоторый физически существующий элемент системы, обладающий некоторым вычислительным ресурсом. В качестве вычислительного ресурса узла может рассматриваться наличие по меньшей мере некоторого объема электронной или магнитооптической памяти и/или процессора. В последней версии UML понятие узла расширено и может включать в себя не только вычислительные устройства (процессоры), но и другие механические или электронные устройства, такие как датчики, принтеры, модемы, цифровые камеры, сканеры и манипуляторы.

Соединения

Кроме собственно изображений узлов на диаграмме развертывания указываются отношения между ними. В качестве отношений выступают физические соединения между узлами и зависимости между узлами и компонентами, изображения которых тоже могут присутствовать на диаграммах развертывания.

Соединения являются разновидностью ассоциации и изображаются отрезками линий без стрелок. Наличие такой линии указывает на необходимость организации физического канала для обмена информацией между соответствующими узлами. Характер соединения может быть дополнительно специфицирован примечанием, помеченным значением или ограничением.

Построение диаграмм развёртывания в StarUML

Диаграмма развёртывания (deployment diagram) добавляется аналогично диаграмме вариантов использования (см. лр.№1).

Добавление элементов диаграммы

Чтобы добавить узел (node) выполните следующие действия: панель Toolbox (слева вверху) → закладка Deployment → Node. После этого щёлкните ЛКМ по рабочей области в том месте, где будет размещаться данный элемент. Имя узла необходимо задать сразу после его добавления на диаграмму. Отредактировать имя – двойной щелчок ЛКМ по узлу.

Отношения на диаграмму развёртывания добавляются так же, как и на диаграмме компонентов.

Чтобы добавить соединение (link) выполните следующие действия: панель Toolbox (слева вверху) → закладка Deployment → Link. После этого щёлкните ЛКМ по рабочей области в том месте, где будет размещаться данный элемент.

Домашнее задание студентам для подготовки к выполнению лабораторной работы

Изучить по лекциям и учебной литературе особенности построения диаграмм развёртывания.

Изучить принципы построения диаграмм развёртывания в StarUML (см. руководство пользователя).

Варианты заданий

См. лр.№1.

Порядок выполнения лабораторной работы

1. Запустите StarUML и откройте файл, который Вы создали в предыдущей работе. Выберите Model1.
2. Постройте диаграмму развёртывания, приведённую на рисунке 25.
3. Сохраните результаты работы.
4. Выберите Model2.
5. Постройте диаграмму развёртывания по выбранной теме.
6. Сохраните результаты работы.

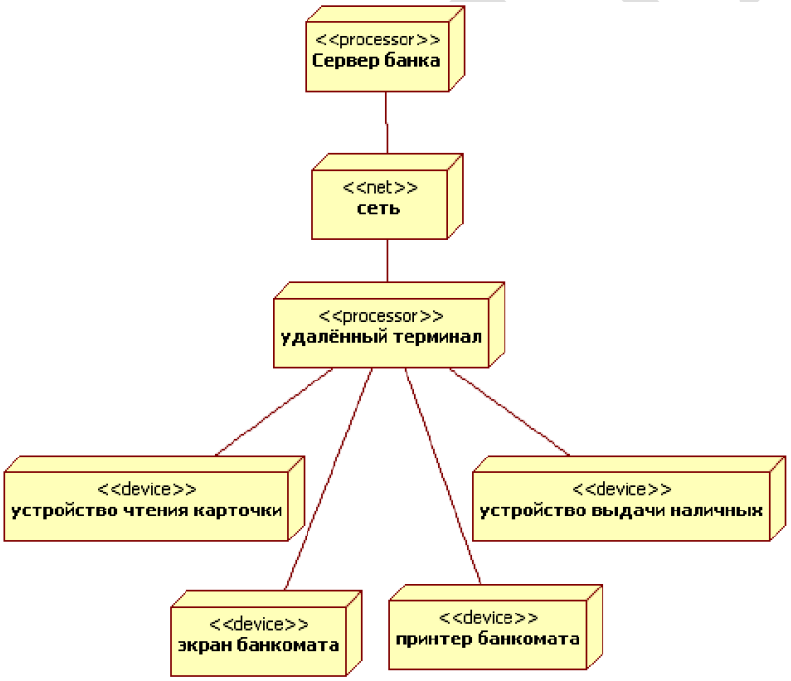

Рис. 25. Диаграмма развёртывания (общая)

Вышеуказанная диаграмма является учебно-тренировочной и предназначена для лучшего изучения особенностей построения диаграмм развёртывания.

Содержание отчёта

1. Титульный лист
2. Цель лабораторной работы
3. Результаты выполнения пунктов 2 и 5. Диаграммы необходимо сохранить в виде рисунков (в StarUML: File → Export Diagram), которые затем вставить в отчёт.
4. Выводы по работе.

Контрольные вопросы

1. Для чего используется диаграмма развёртывания?
2. Перечислите и опишите основные элементы диаграммы развёртывания.
3. Как изображаются основные элементы диаграммы? Как их добавить на диаграмму в StarUML?
4. Приведите примеры узлов.
5. Перечислите рекомендации по построению диаграмм развёртывания?

Список литературы

Обязательная

1. Пальмов С.В. Конспект лекций по дисциплине «Методы и средства проектирования информационных систем и технологий».
2. Руководство пользователя для StarUML.

Дополнительная

1. Леоненков А. Самоучитель UML. СПб.: БХВ-Петербург, 2006.

Примерный перечень тем

1. Информационная система торгового предприятия.
2. Информационная система аптеки.
3. Информационная система отдела кадров.
4. Программное обеспечение мобильного телефона.
5. Программное обеспечение роутера.
6. Программное обеспечение принтера.
7. Программное обеспечение МФУ.
8. Программное обеспечение навигатора.
9. Программное обеспечение видеорегистратора.
10. Программное обеспечение мультимедийной клавиатуры.
11. Текстовый редактор.
12. Мультимедийный плеер.
13. Поисковая система.
14. Информационная система.
15. Интернет-магазин.