ПРАКТИЧЕСКИЕ ЗАНЯТИЯ
по дисциплине
ТЕХНОЛОГИЯ КОНСТРУИРОВАНИЯ ПО

Рига 2013

Содержание

ПЗ 6 (Case Study 6). Формирование требований — создание диаграммы Use Case

В компанию по разработке ПО обратился хозяин мебельного магазина с заказом на создание программной системы учета заказов. Из-за наплыва клиентов учет заказов на бумаге и в электронных таблицах перестал быть возможен. Создаваемая программная система должна решить эту проблему.

Система должна обеспечивать возможность добавления новых заказов, изменения ранее введенных в нее заказов, учета выполнения заказов, проведения инвентаризаций на складе с составлением описей. При получении нового заказа система должна также послать сообщение бухгалтерской системе, которая выписывает счет. Любой заказ может содержать одну или более товарных позиций. Для каждой позиции заказа указывается наименование товара и его количество. Заполненный заказ получает кладовщик, который начинает сборку заказа. Если для каждой позиции товара на складе находится товар в достаточном количестве, то товар резервируется, и заказ помечается выполненным. Если требуемого товара нет на складе, то заказ может быть отменен, либо выполнение заказа задерживается до поступления товара на склад.

Требования заказчика

Формирование требований – это процесс, в ходе которого определяются задачи, поставленные перед разработчиками, и создаются модели, на основе которых планируется разработка системы. Требование фиксирует условие, которому должно удовлетворять программный продукт, или характеристику, которой он должен обладать, чтобы удовлетворить пожелание пользователя при решении определенной задачи, или соблюдении пунктов контракта, стандарта. Все требования делятся на функциональные и нефункциональные. Функциональные требования определяют действия, которые должна выполнять система, без учета ограничений, связанных с ее реализацией. Нефункциональные требования не определяют поведение системы, но описывают ее характеристики или атрибуты внешней среды. Например, нефункциональными являются требования к производительности системы и требования к аппаратным средствам.

Требования оформляются в виде ряда документов и моделей. К основным документам относятся: концепция, словарь предметной области, дополнительная спецификация.

Концепция определяет глобальные цели проекта и основные особенности разрабатываемой системы. Существенной частью концепции является постановка задачи разработки, определяющая требования к функциям системы.

Словарь предметной области определяет общую терминологию для всех моделей и описаний требований к системе.

Дополнительная спецификация содержит описание нефункциональных требований к системе, таких, как надежность, удобство использования, производительность, сопровождаемость и т.д.

Постановка задачи для системы обработки заказов

Пользователями новой системы будут продавцы и работники склада (заведующий и кладовщики).

База данных системы будет поддерживаться реляционной СУБД. Система должна обеспечивать возможность продавцам вводить новые заказы и изменять заказы, хранящиеся в системе. Заказ может быть изменен до тех пор, пока не закончились работы на складе по его сборке. Собранные (выполненные) заказы поставляются заказчикам, внесение в них изменений запрещено. Дата окончания сборки заказа хранится в системе. После нее заказ считается выполненным. Не выполненный заказ может быть отменен.

При вводе заказа важно сохранить дату, когда был принят заказ, и дату, до которой нужно осуществить сборку и доставку заказа. Каждый заказ содержит одну или несколько позиций. В любой позиции указывается наименование предмета мебели и количество штук. После ввода заказа данные передаются в бухгалтерскую систему для составления счета на оплату.

Работают несколько продавцов, поэтому необходимо обеспечить защиту данных таким образом, чтобы продавец мог работать только с собственными заказами, и не имел доступа к данным чужих заказов. Продавец может удалить данные о любом из своих заказов.

Заведующий складом использует систему, чтобы напечатать остатки — опись, в которой указывается текущее количество предметов мебели на складе. Остатки определяются системой по данным последней инвентаризации и данным о выполнении заказов. Например, если по данным инвентаризации было 10 стульев и 7 стульев отмечены как выполненные позиции заказов введенных после инвентаризации, (т. е. стулья переданы заказчикам или отложены в собираемые заказы), то текущий остаток — 3 стула. При

проведении инвентаризации для каждого предмета мебели со склада вводится текущее его количество, относительно которого будут рассчитываться остатки до следующей инвентаризации.

Кладовщики отмечают в системе ход выполнения заказов. Любой кладовщик может работать с любым заказом. Кладовщик может пометить какую-либо позицию заказа как выполненную. При этом соответствующее количество предметов мебели вычитается при расчете текущих остатков. Если выполнены все позиции заказа, он также считается выполненным. Кладовщик может пометить невыполненный заказ как отмененный, если нужной для выполнения заказа мебели на складе нет. При этом снимается резерв с выполненных позиций отмененного заказа и, соответственно, увеличиваются остатки.

Моделирование предметной области

Модель предметной области широко используется в качестве основы для разработки программных объектов и обеспечивает важную входную информацию для создания последующих артефактов. Модель предметной области отображает основные (с точки зрения моделирующего) классы понятий предметной области. Модель предметной области — это визуальное представление концептуальных классов или объектов реального мира в терминах предметной области.

На языке UML модель предметной области представляется в виде набора диаграмм классов, на которых не определены никакие операции. Например, на рис. 1 представлен фрагмент модели предметной области, из которого ясно, что с точки зрения предметной области концептуальными классами являются Payment (Платеж) и Sale (Продажа). Как видно из диаграммы, эти понятия связаны между собой и понятию Sale соответствуют определенная дата и время.

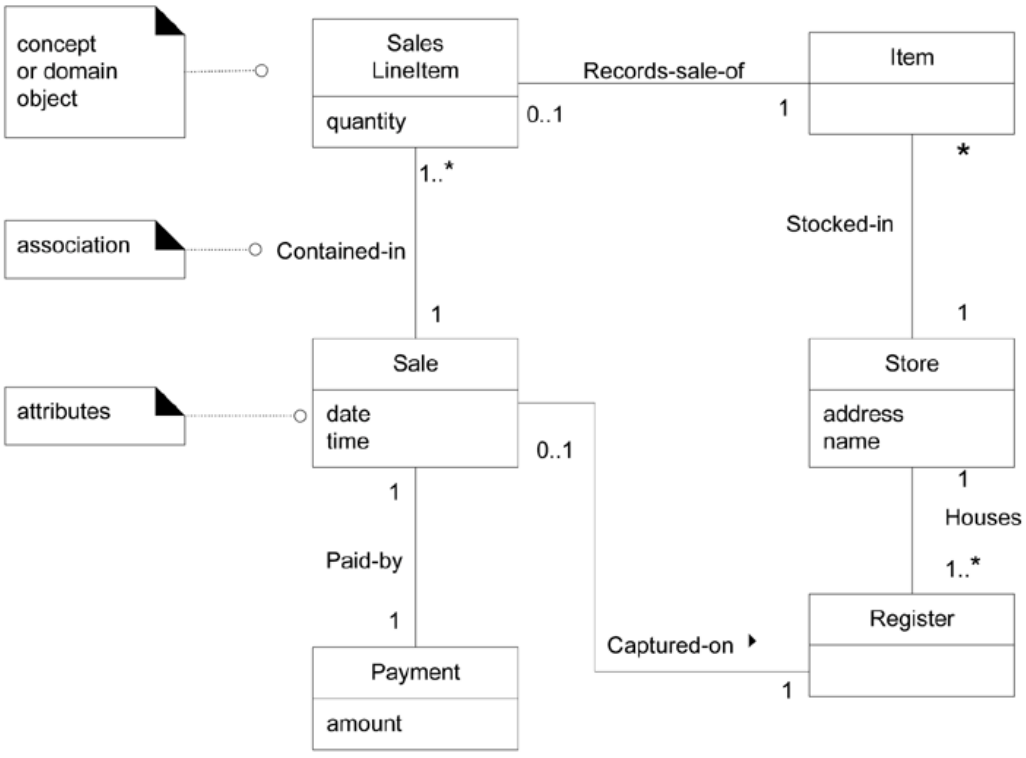


Рис. 1. Фрагмент модели предметной области — визуальный словарь

Еще раз заметим, что модель предметной области представляет классы понятий реального мира, а не программные компоненты. Это не набор диаграмм, описывающих программные классы.

Представленную на диаграмме (в системе обозначений UML) информацию можно также выразить в виде словесного описания, терминов словаря и т.д. Однако элементы и их взаимосвязи легче представить на этом визуальном языке, поскольку одним из преимуществ человеческого интеллекта является хорошая способность обработки визуальной информации.

Таким образом, модель предметной области можно рассматривать как визуальный словарь важных абстракций или словарь предметной области.

Модель предметной области — это результат визуализации понятий реального мира в терминах предметной области, а не программных элементов, таких как классы Java, C# или C++. Следовательно, в модели предметной области не используются следующие элементы:

- Артефакты программирования наподобие окон или базы данных, если только разрабатываемая система не является моделью программного средства, например моделью графического интерфейса пользователя.
- Обязанности или методы.

Модель предметной области иллюстрирует концептуальные классы или словарь предметной области. Неформально, концептуальный класс — это представление идеи или объекта.

При построении моделей предметной области применяется та же стратегия, что и при создании карт местности:

- Использовать применяемые на данной территории названия
- Исключать несущественные детали
- Не добавлять объекты, которые отсутствуют на данной территории.

Модель предметной области — это своеобразная разновидность карты понятий некоторой предметной области. Отсюда следуют аналитическая роль модели предметной области, а также справедливость следующих замечаний.

- Картографы используют названия, применяемые на данной территории. Они не изменяют названия городов на карте. С точки зрения модели предметной области это означает необходимость использования словаря предметной области при именовании концептуальных классов и атрибутов. Например, при разработке модели библиотеки в качестве понятия, означающего потребителя, следует выбрать Borrower (Читатель), т.е. использовать терминологию библиотекарей.
- Картограф не наносит на карту объекты, не имеющие отношения к основному ее назначению, например не отображает топографию или состав населения. Аналогично, в модели предметной области не должны содержаться понятия из предметной области, не имеющие отношения к требованиям. Например, из нашей модели предметной области можно исключить понятия Pen (Ручка) и PaperBag (Папка), поскольку они не имеют отношения к требованиям.
- Картограф не отображает на карте отсутствующие объекты, например, горы на карте равнинной местности. Точно так же, в модели предметной области не должны содержаться понятия, не имеющие отношения к рассматриваемой проблеме.

Возможно, наиболее типичной ошибкой при создании модели предметной области является отнесение некоторого объекта к атрибутам, в то время как он должен относиться к концептуальным классам. Чтобы избежать этой ошибки, следует придерживаться простого правила:

Если некоторый объект X в реальном мире не является числом или текстом, значит, это скорее концептуальный класс, чем атрибут.

Например, является ли store (магазин) атрибутом объекта Sale (Продажа) или отдельным концептуальным классом Store?

В реальном мире магазин не является числом или текстом, он представляет реальную сущность, организацию, занимающую некоторое место. Следовательно, Store нужно рассматривать в качестве концептуального класса.

Если у вас возникают сомнения при разграничении понятий и атрибутов, создавайте отдельный концептуальный класс. Атрибуты редко отображаются в модели предметной области.

В нашем примере ограничимся лишь начальным шагом в создании модели предметной области — созданием словаря понятий предметной области. Он составляется на основе описания системы обработки заказов, постановки задачи и концепции. Словарь предназначен для описания терминологии области, в которой будет работать программный продукт. Выделяются термины, им дается описание, рассчитанное на широкий круг читателей. Словарь составляется на русском языке. Термины сопровождаются переводом на английский на тот случай, если термин будет использован в модели системы как название класса, пакета и т. п.

Словарь понятий предметной области системы обработки заказов

Бухгалтерская система (Accounting System)	Внешняя система, в которую передаются данные обо всех введенных заказах
Заведующий складом (Warehouseman)	Пользователь системы. Имеет возможность распечатать остатки по состоянию склада на какой-либо день и провести инвентаризацию, т. е. ввести в систему данные о реальном (не рассчитанном) количестве хранимой мебели
Заказ (Order)	Непустой перечень требуемых заказчиком позиций. Дата заказа указывает момент его создания. Дата поставки заказа отмечает день, к

	которому должны быть завершены работы по сборке и поставке заказа. Дата выполнения заказа указывает день, когда была помечена выполненной последняя из невыполненных позиций заказа
Заказчик (Customer)	Покупатель мебельного магазина. Данные о покупателе включают в себя Ф., И., О., контактный телефон, адрес для доставки мебели
Инвентаризационная опись (Inventory)	Перечень, в котором для каждого предмета мебели, хранящегося на складе, указано его количество
Кладовщик (Stockman)	Пользователь системы. Работник склада, отвечающий за сборку заказов. Может помечать позиции заказов как выполненные, а невыполненные заказы — как отмененные.
Остатки (Balance)	Данные о количестве предметов мебели на складе, рассчитываемые по сведениям из последней инвентаризации и данным о выполненных позициях заказов
Позиция заказа (Order Item)	Один или более одинаковых предметов мебели, указанных в заказе. Позиция характеризуется наименованием, количеством, номером по порядку и статусом (выполнена или нет)
Предмет мебели (Article of furniture)	Предмет обстановки, хранящийся на складе. Может быть указан в позиции заказа. Характеризуется наименованием. Количество предметов мебели указывается в инвентаризационных описях
Продавец (Salesperson)	Пользователь системы. Автор произвольного количества заказов. Может вводить заказы и изменять введенные им ранее заказы

Дополнительная спецификация

Дополнительная спецификация определяет нефункциональные требования к системе, такие, как режимы работ, ограничения, особенности реализации, надежность, удобство использования, производительность, сопровождаемость и т.д. Назначение дополнительной спецификации — определить требования к системе обработки заказов, которые не отражены в других документах и моделях. Вместе они образуют полный набор требований к системе. Рассмотрим дополнительную спецификацию:

Режимы работ и ограничения

Система должна обеспечивать многопользовательский режим работы. Несколько кладовщиков и/или продавцов могут одновременно использовать систему.

Система должна обеспечивать выполнение следующих правил:

- В заказе должна быть хотя бы одна позиция. Пустые заказы не сохраняются в системе.
- Количество предметов мебели в любой позиции заказа — натуральное число.
- Невыполненный заказ может быть отменен. При этом позиции, которые были отмечены как выполненные, отменяются, остатки увеличиваются.
- Система должна поддерживать протокол обмена данных с бухгалтерской системой.

Особенности реализации

Система должна работать в операционной среде Windows.

Надежность

Система должна быть в работоспособном состоянии 24 часа в день 7 дней в неделю, время простоя — не более 10% от времени работы.

Производительность

Система должна поддерживать до 50 одновременно работающих пользователей.

Безопасность

Система должна запрещать каждому продавцу изменять заказы, которые созданы другими продавцами. Только кладовщики имеют право отмечать выполнение и отмену заказов. Только заведующий складом может распечатать остатки и провести инвентаризацию.

Создание диаграммы Use Case

Запустите Rational Rose и создайте модель, используя шаблон для Java. Rose добавит в пустую модель библиотечные классы Java, которые впоследствии понадобятся нам при генерации программного кода. Создайте диаграмму Use Case для системы обработки заказов. Требуемые для этого действия подробно перечислены далее.

Из постановки задачи разработки и словаря предметной области можно выделить следующий список актеров:

- Продавец — вводит заказ, изменяет заказ;
- Кладовщик — выполняет заказ, помечает заказ как отмененный;
- Заведующий складом — проводит инвентаризацию, распечатывает остатки;
- Бухгалтерская система — получает данные о введенных заказах.

Готовая диаграмма Use Case должна выглядеть как на рис. 2.

Этапы выполнения упражнения

Создать элементы Use Case и актеров

1. Дважды щелкните на Главной диаграмме Use Case (Main) в браузере, чтобы открыть ее.
2. С помощью кнопки Use Case панели инструментов поместите на диаграмму новый элемент Use Case.
3. Назовите этот новый элемент "Ввести новый заказ".
4. Повторите этапы 2 и 3, чтобы поместить на диаграмму остальные элементы Use Case: Изменить заказ, Напечатать инвентарную опись, Обновить инвентарную опись, Оформить заказ, Отклонить заказ.
5. С помощью кнопки Actor панели инструментов поместите на диаграмму нового актера.
6. Назовите его "Продавец".
7. Повторите шаги 5 и 6, поместив на диаграмму остальных актеров: Заведующий складом, Кладовщик, Бухгалтерская система.

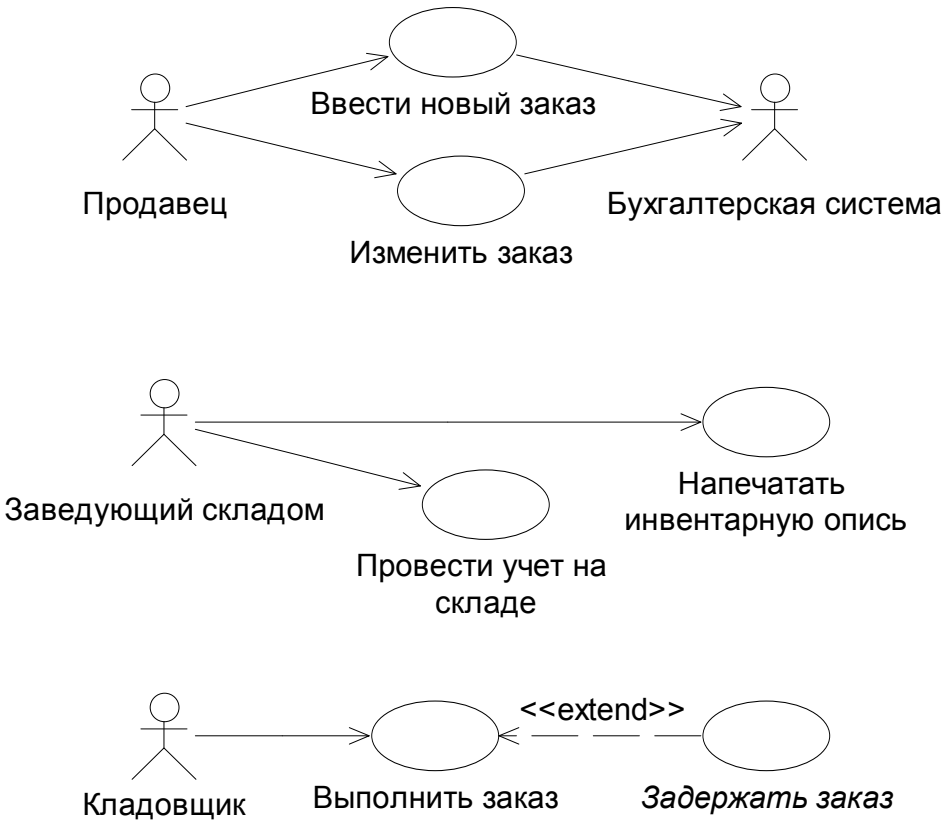


Рис. 2. Диаграмма Use Case для системы обработки заказов

Указать абстрактный элемент Use Case

1. Щелкните правой кнопкой мыши на варианте использования "Задержать заказ" на диаграмме.

2. В открывшемся меню выберите пункт Open Specification (Открыть спецификацию).
3. Пометьте контрольный переключатель Abstract (Абстрактный), чтобы сделать этот вариант использования абстрактным.

Добавить ассоциации

1. С помощью кнопки Unidirectional Association (Однонаправленная ассоциация) панели инструментов нарисуйте ассоциацию между актером Продавец и элементом "Ввести новый заказ".
2. Повторите этот этап, чтобы поместить на диаграмму остальные ассоциации.

Добавить отношение расширения

1. С помощью кнопки Dependency (Зависимость) панели инструментов нарисуйте зависимость между элементом "Задержать заказ" и элементом "Выполнить заказ". Стрелка должна протянуться от первого элемента ко второму. Отношение расширения означает, что элемент "Задержать заказ" при необходимости дополняет функциональные возможности элемента "Выполнить заказ".
2. Щелкните правой кнопкой мыши на линии новой зависимости между элементами "Задержать заказ" и "Выполнить заказ".
3. В открывшемся меню выберите пункт Open Specification (Открыть спецификацию).
4. В раскрывающемся списке стереотипов введите слово extend (расширение), затем нажмите OK.
5. Слово <<extend>> появится на линии данной зависимости.

Добавить описания к элементам Use Case

1. Выделите в браузере элемент "Ввести новый заказ".
2. В окне документации введите следующее описание к этому элементу: Этот элемент Use Case дает продавцу возможность ввести новый заказ в систему.
3. С помощью окна документации введите описания ко всем остальным элементам Use Case.

Добавить описание к актеру

1. Выделите в браузере актера Продавец.
2. В окне документации введите для этого актера следующее описание: Продавец — это служащий, который вводит в систему новый заказ, изменяет заказ.
3. С помощью окна документации введите описания к оставшимся актерам.

Прикрепление файла к элементу Use Case

1. Для описания элемента Use Case "Ввести новый заказ" создайте файл OrderFlow.doc, содержащий следующий текст:
    1. Продавец выбирает пункт "Создать новый заказ" из имеющегося меню.
    2. Система выводит форму "Подробности заказа".
    3. Продавец вводит номер заказа, заказчика и то, что заказано.
    4. Продавец сохраняет заказ.
    5. Система создает новый заказ и сохраняет его в базе данных.
2. Щелкните правой кнопкой мыши на элементе Use Case "Ввести новый заказ".
3. В открывшемся меню выберите пункт Open Specification (Открыть спецификацию)
4. Перейдите на вкладку файлов.
5. Щелкните правой кнопкой мыши на белом поле и из открывшегося меню выберите пункт Insert File (Ввести файл).
6. Укажите файл OrderFlow.doc и нажмите на кнопку Open (Открыть), чтобы прикрепить файл к элементу Use Case.

Формат описаний элементов Use Case

Конечно, желаемое поведение разрабатываемой системы, соответствующее требованиям заказчика, более подробно фиксируется в описаниях элементов Use Case. Предусматривается специальная форма, позволяющая уменьшить вероятность неверного толкования и облегчить восприятие текста. Каждое описание содержит:

1. краткую аннотацию, являющуюся сжатым обзором назначения элемента Use Case;
2. основной поток событий, описывающий такое взаимодействие системы и актеров, при котором обеспечиваются интересы основного актера;
3. альтернативные потоки, описывающие обработку ошибок и исключительных ситуаций;
4. подчиненные потоки, которые облегчают описание основного и альтернативных потоков;
5. необходимые предусловия и обеспечиваемые постусловия.

Каждый поток событий задается пронумерованным набором шагов. Используются шаги трех видов: действие системы (например, «Система запрашивает имя пользователя и пароль»); реакция актера

(«Пользователь вводит имя и пароль»); управление потоком («Выполнение переходит на начало основного потока»). Структура предложений, описывающих шаги, одинакова: существительное, глагол, снова существительное. Особую форму имеют циклические шаги и шаги ветвления.

Описание цикла начинается с условия повторения (Для каждого ... выполняется, или повторить n раз). Далее записывается тело цикла (последовательность вложенных шагов).

Ветвления обычно описывают с помощью альтернативных потоков, которые указываются для конкретных шагов процесса.

Задание на самостоятельную работу 1.

Конечно, использованное описание элемента Use Case «Ввести новый заказ» имеет существенные недостатки:

- Не предусмотрена проверка подлинности пользователя-продавца;
- Проигнорирована необходимость взаимодействия с бухгалтерской системой;
- Ввод данных заказа описан лишь в самых общих чертах;
- Не предусмотрены никакие альтернативы поведения.

Устраните эти недостатки, создав альтернативную диаграмму Use Case, где предусмотрено описание элемента «Ввести новый заказ», например, в следующем виде:

Краткое описание

Данный вариант использования позволяет продавцу ввести данные о новом заказе. Сведения о заказе включают в себя данные о заказчике, дату поставки заказа и перечень позиций в заказе. Система присваивает заказу дату создания. Новые сведения о заказе передаются системой в бухгалтерскую систему.

Основной поток событий

1. Система запрашивает имя продавца и пароль.
2. Продавец вводит имя и пароль.
3. Система подтверждает правильность имени и пароля. Система запрашивает связь с бухгалтерской системой.
4. Бухгалтерская система подтверждает, что связь установлена.
5. Система запрашивает данные о заказе.
6. Продавец вводит данные о заказчике и дате поставки заказа.
7. Для каждой позиции нового заказа выполняется:
   7.1. Продавец вводит наименование и количество.
   7.2. Система подтверждает, что наименование и количество указаны верно.
8. Продавец сообщает системе о необходимости сохранить заказ.
9. Система сохраняет данные о заказе.
10. Система передает данные о заказе бухгалтерской системе.
11. Система заканчивает сеанс связи с бухгалтерской системой.
12. Бухгалтерская система подтверждает, что сеанс закончен.

Альтернативные потоки

3А. Неправильное имя/пароль

3А.1. Система обнаруживает, что комбинация имени и пароля не верна.

3А.2. Система сообщает об ошибке и предлагает продавцу либо заново ввести имя и пароль, либо отказаться от входа в систему.

3А.3. Продавец сообщает системе свой выбор.

3А.4. В соответствии с выбором продавца либо выполнение переходит на шаг 1 основного потока, либо сеанс работы завершается.

4А. Бухгалтерская система недоступна

4А.1. Система обнаруживает, что невозможно установить связь с бухгалтерской системой.

4А.2. Система выдает сообщение об ошибке.

4А.3. Сеанс работы завершается.

7.2А. Ошибка при вводе позиции заказа

7.2А.1. Система обнаруживает, что наименование предмета мебели либо количество указаны неверно.

7.2А.2. Система выдает сообщение об ошибке.

7.2А.3. Управление передается на шаг 7.1. Ввести заказ.

Предусловия

Отсутствуют.

Постусловия

Если элемент Use Case завершится успешно, данные о новом заказе будут внесены в систему и переданы в бухгалтерскую систему.

Задание на самостоятельную работу 2.

Реализуйте описание элемента Use Case «Изменить заказ» в следующем формате:

Краткое описание

Данный вариант использования позволяет продавцу изменить данные о введенном заказе. Изменения в заказе передаются системой в бухгалтерскую систему.

Основной поток событий

1. Система запрашивает имя продавца и пароль.
2. Продавец вводит имя и пароль.
3. Система подтверждает правильность имени и пароля. Система запрашивает связь с бухгалтерской системой.
4. Бухгалтерская система подтверждает, что связь установлена.
5. Система выводит список заказов, созданных текущим продавцом.
6. Продавец выбирает заказ из списка.
7. Система выводит все данные о заказе.
8. Продавец изменяет данные в заказе и сообщает системе о необходимости сохранить заказ.
9. Система подтверждает, что измененные данные в заказе указаны корректно.
10. Система сохраняет данные о заказе.
11. Система передает данные об изменении заказа бухгалтерской системе.
12. Система заканчивает сеанс связи с бухгалтерской системой.
13. Бухгалтерская система подтверждает, что сеанс закончен.

Альтернативные потоки

3А. Неправильное имя/пароль

3А.1. Система обнаруживает, что комбинация имени и пароля не верна.

3А.2. Система сообщает об ошибке и предлагает продавцу либо заново ввести имя и пароль, либо отказаться от входа в систему.

3А.3. Продавец сообщает системе свой выбор.

3А.4. В соответствии с выбором продавца либо выполнение переходит на шаг 1 основного потока, либо сеанс работы завершается.

4А. Бухгалтерская система недоступна

4А.1. Система обнаруживает, что невозможно установить связь с бухгалтерской системой.

4А.2. Система выдает сообщение об ошибке.

4А.3. Сеанс работы завершается.

9А. Система обнаруживает, что измененные данные в заказе указаны неверно.

9А.1. Система выдает сообщение об ошибке.

9А.2. Управление передается на шаг 8. Изменить заказ.

Предусловия

Отсутствуют.

Постусловия

Если элемент Use Case завершится успешно, данные о новом заказе будут внесены в систему и переданы в бухгалтерскую систему.

Задание на самостоятельную работу 3.

Выделите из всех элементов Use Case диаграммы поток аутентификации пользователя в отдельный абстрактный элемент, подключаемый отношением включения. Модифицируйте описания всех элементов диаграммы.

Задание на самостоятельную работу 4.

Объедините элементы «Ввести новый заказ» и «Изменить заказ» в один элемент Use Case, предлагающий (через меню) три операции с заказом: ввод, изменение, удаление.

ПЗ 7. Анализ требований — создание диаграмм взаимодействия

Разработаем диаграммы последовательности и диаграммы сотрудничества (коммуникации), описывающие введение нового заказа в нашу систему обработки заказов.

Постановка задачи

Суть анализа требований состоит в формализации и уточнении тех требований, которые в текстовой форме желаемого поведения были представлены в диаграмме Use Case. Здесь используются диаграммы последовательности (реже диаграммы сотрудничества). Минимальное количество этих диаграмм равно количеству элементов Use Case.

Создание диаграмм взаимодействия

Создайте диаграмму последовательности и диаграмму сотрудничества, отражающую ввод нового заказа в систему обработки заказов.

Это только одна из диаграмм, необходимых для моделирования элемента Use Case «Ввести новый заказ». Она соответствует успешному варианту хода событий. Для описания того, что случится, если возникнет ошибка, или если пользователь выберет другие действия из предложенных, придется разработать другие диаграммы. Каждый альтернативный поток элемента Use Case может быть промоделирован с помощью своих собственных диаграмм взаимодействия.

Ранее мы использовали русские имена, начиная с этого упражнения, будем использовать английские, так как создаваемые нами элементы модели будут транслироваться в программный код.

Этапы выполнения упражнения

Настройка

1. В меню модели выберите пункт Tools > Options (Инструменты > Параметры).
2. Перейдите на вкладку диаграмм.
3. Контрольные переключатели Sequence Numbering и Collaboration Numbering должны быть помечены.
4. Нажмите ОК, чтобы выйти из окна параметров.

Создание диаграммы Последовательности

1. Щелкните правой кнопкой мыши на Логическом представлении браузера.
2. В открывшемся меню выберите пункт New > Sequence Diagram.
3. Назовите новую диаграмму "Ввод заказа".
4. Дважды щелкните на ней, чтобы открыть ее.

Добавление на диаграмму актера и объектов, связывание с классами

5. Перетащите актера Продавец из браузера на диаграмму. Откройте двойным щелчком спецификацию добавленного на диаграмму лица и введите имя объекта Вася.
6. На панели инструментов нажмите кнопку Object (Объект).
7. Щелкните мышью в верхней части диаграммы, чтобы поместить туда новый объект.
8. Назовите объект «Order Options Form» (Форма выбора заказа).
9. Повторите этапы 3 и 4, чтобы поместить на диаграмму все остальные объекты: Order Detail Form, Order #1234
10. Щелкните правой кнопкой мыши на объекте Order Options Form.
11. В открывшемся меню выберите пункт Open Specification.
12. В раскрывающемся списке классов выберите пункт New. Появится окно спецификации классов.
13. В поле имени введите имя OrderOptions (Выбор заказа).
14. Щелкните на кнопке ОК. Вы вернетесь к окну спецификации объекта.
15. В списке классов выберите теперь класс OrderOptions.
16. Щелкните на кнопке ОК, чтобы вернуться к диаграмме. Теперь объект называется Order Options Form : OrderOptions (Форма выбора заказа : Выбор заказа).
17. Для связывания остальных объектов с классами повторите этапы с 6 по 12: класс OrderDetail свяжите с объектом Order Detail Form; класс Order — с объектом Order #1234.

Добавление сообщений на диаграмму

1. На панели инструментов нажмите кнопку Object Message (Сообщение объекта).
2. Проведите мышью от линии жизни актера Продавец к линии жизни объекта Order Options Form.
3. Выделив сообщение, введите его имя "Create" -- Создать новый заказ.

4. Повторите этапы 2 и 3, чтобы поместить на диаграмму дополнительные сообщения:
    - Open — Открыть форму (между Order Options Form и Order Detail Form)
    - SubmitInfo — Ввести номер заказа, заказчика и число заказываемых предметов (между Продавцом и Order Detail Form)
    - Save — Сохранить заказ (между Продавцом и Order Detail Form)
    - Create — Создать пустой заказ (между Order Detail Form и Order #1234)
    - SetInfo — Ввести номер заказа, заказчика и число заказываемых предметов (между Order Detail Form и Order #1234).

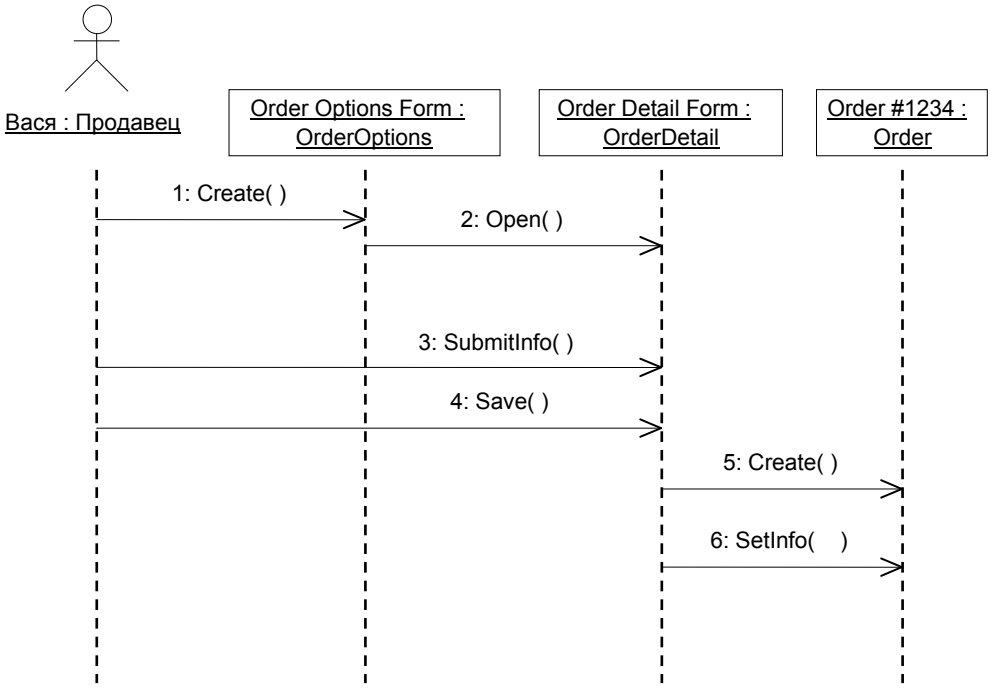


Рис. 3. Диаграмма последовательности ввода нового заказа после завершения первого этапа работы

Мы завершили первый этап работы. Готовая диаграмма последовательности представлена на рис. 3.

Она отражает желаемое поведение с точки зрения заказчика. Теперь можно углубиться в детали программной реализации, которые для заказчика неочевидны. В частности, целесообразно снять с объекта Order Detail Form менеджерские функции и возложить их на специальный объект-менеджер, который будет управлять всеми действиями при вводе нового заказа. Кроме того, целесообразно отделять бизнес-логику системы от логики взаимодействия с базой данных. Следовательно, желательно добавление менеджера транзакций с БД.

Добавление на диаграмму дополнительных объектов

1. На панели инструментов нажмите кнопку Object.
2. Щелкните мышью между объектами Order Detail Form и Order #1234, чтобы поместить туда новый объект.
3. Введите имя объекта — Order Manager.
4. На панели инструментов нажмите кнопку Object.
5. Новый объект расположите справа от Order #1234.
6. Введите его имя — Transaction Manager.

Изменение сообщений между объектами, связывание объектов с классами

1. Выделите сообщение 5 (Create).
2. Нажмите комбинацию клавиш CTRL + D, чтобы удалить это сообщение.
3. Повторите этапы 1 и 2, чтобы удалить сообщение: SetInfo.
4. Создайте классы и свяжите объекты с классами: OrderMgr — с объектом Order Manager, TransactionMgr — с объектом Transaction Manager.
5. На панели инструментов нажмите кнопку Object Message.
6. Поместите на диаграмму новое сообщение, расположив его под сообщением 4 между объектами Order Detail Form и Order Manager.
7. Назовите это сообщение SaveOrder (Сохранить заказ).
8. Повторите этапы 5 – 7, добавив сообщения с шестого по девятое и назвав их:
    - Create (Создать новый заказ) — между Order Manager и Order #1234.

- SetInfo (Вести номер заказа, заказчика и число заказываемых предметов) — между Order Manager и Order #1234.
- SaveOrder (Сохранить заказ) — между Order Manager и Transaction Manager.
- GetInfo (Получить информацию о заказе) — между Transaction Manager и Order #1234.

9. На панели инструментов нажмите кнопку Message to Self (Сообщение себе).
10. Щелкните на линии жизни объекта Transaction Manager ниже сообщения 9, добавив туда рефлексивное сообщение. Назовите его Commit (Сохранить информацию о заказе в базе данных).

Связывание сообщений с операциями

1. Щелкните **правой** кнопкой на сообщении 1:Create.
2. В открывшемся меню выберите пункт <new operation> (создать операцию). Появится окно спецификации операции.
3. В поле имени введите имя операции — Create (Создать).
4. Нажмите на кнопку ОК, чтобы закрыть окно спецификации операции и вернуться в диаграмму.
5. Повторите шаги 1–4, пока не свяжете с операциями все остальные сообщения со 2-го по 10-е.

Теперь диаграмма последовательности должна выглядеть так, как показано на рис. 4.

Задание на самостоятельную работу

Создайте диаграммы последовательности для всех остальных элементов из диаграммы Use Case.

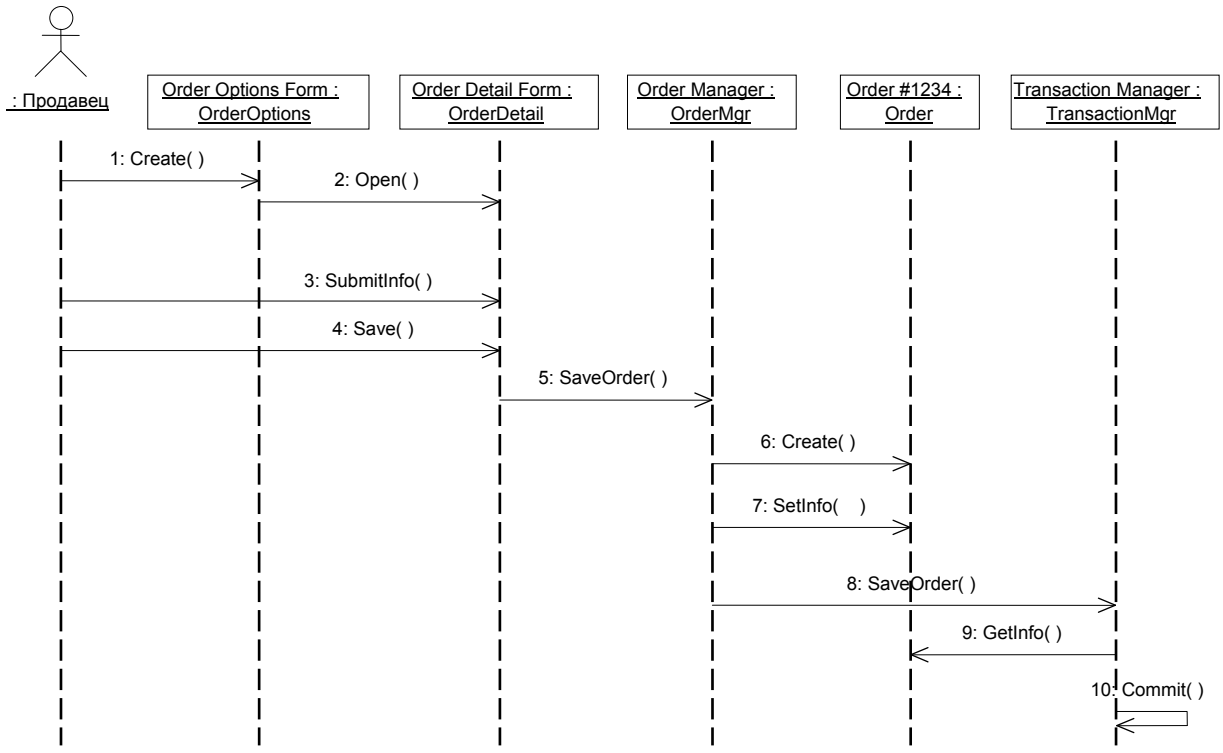


Рис. 4. Диаграмма последовательности с новыми объектами, классами и операциями

Создание диаграммы сотрудничества

Для создания диаграммы сотрудничества достаточно просто нажать клавишу F5 и расположить элементы диаграммы так, чтобы они не налезали друг на друга. Также следует удалить ненужное соединение объектов Order #1234 и Order Detail Form, вдоль которого не пересылаются никакие сообщения.

Ваша диаграмма должна выглядеть как на рис. 5.

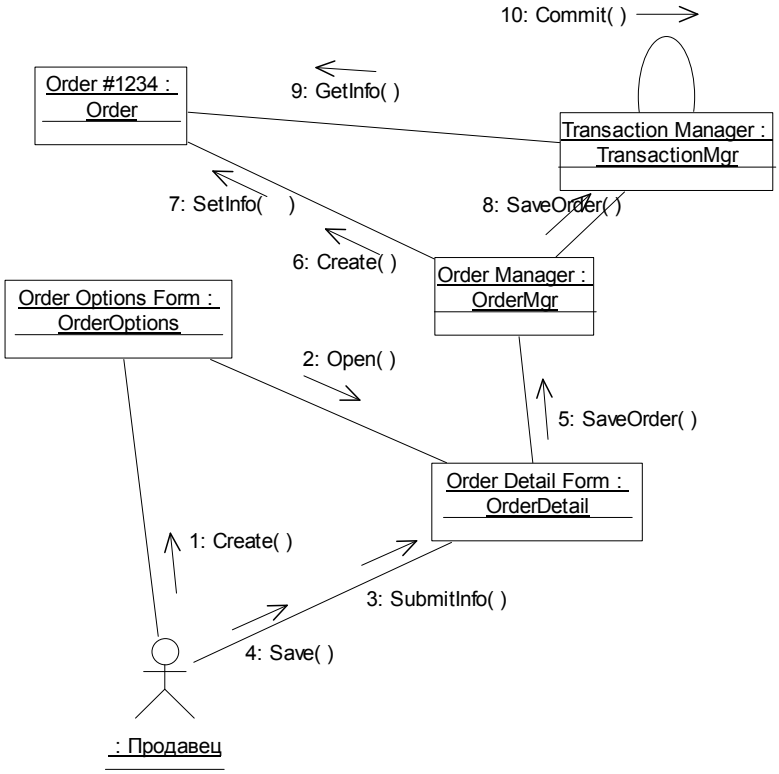


Рис. 5. Диаграмма сотрудничества с показанными на ней операциями.

ПЗ 8. Проектирование — создание диаграмм классов

В этом упражнении надо будет сгруппировать в пакеты классы, созданные во время выполнения предыдущего упражнения (это одна из задач архитектурного проектирования, формирующая подсистемы). Затем предстоит создать несколько диаграмм классов, в которых также показываются классы и пакеты системы (детальное проектирование).

Постановка задачи

Исходные данные для проектирования — набор диаграмм последовательности, в которых указаны объекты и классы анализа, а также диаграмма Use Case.

Классы анализа отражают функциональные требования к системе и задают типы объектов. Совокупность классов анализа образует модель анализа системы. Эта модель проста и позволяет сосредоточиться на реализации функциональных требований, не отвлекаясь на детали реализации, обеспечение эффективности и надежности. Для решения этих вопросов впоследствии модель анализа будет трансформирована в проектную модель. В ходе анализа содержания диаграмм последовательности выявляются классы трех типов:

- граничные классы (boundary classes), являющиеся посредниками при взаимодействии системы с актерами;
- классы-сущности (entity classes), отвечающие за хранение данных;
- управляющие классы (control classes), реализующие бизнес-логику и обеспечивающие координацию поведения объектов в системе.

Правило выделения граничных классов: для каждой связи между актером и системой создается или назначается граничный класс, отвечающий за данное взаимодействие. Правило выделения классов-сущностей: классы-сущности — это, обычно, классы, представляющие ключевые абстракции системы. Правило выделения управляющих классов: для каждой диаграммы последовательности создается ответственный за координацию ее действий класс управления.

В силу описанных причин удобно создать пакеты Entities (Сущности), Boundaries (Границы) и Control (Управление), поместив в них соответствующие классы. Затем на основе каждого пакета проектируются диаграммы классов. На главной диаграмме показываются все пакеты, а на диаграмме «Ввод нового заказа» — все классы этого сценария (диаграммы последовательности, элемента Use Case).

Создание диаграммы классов

Объедините обнаруженные классы в пакеты. Создайте диаграмму классов для отображения пакетов, диаграммы классов для представления классов в каждом пакете и диаграмму классов для представления всех классов сценария "Ввести новый заказ".

Этапы выполнения упражнения

Настройка

1. В меню модели выберите пункт Tools > Options (Инструменты > Параметры).
2. Перейдите на вкладку диаграмм.
3. Убедитесь, что помечен контрольный переключатель Show Stereotypes (Показать стереотипы).
4. Убедитесь, что помечены контрольные переключатели Show All Attributes (Показать все атрибуты) и Show All Operations (Показать все операции).
5. Убедитесь, что не помечены переключатели Suppress Attributes (Подавить вывод атрибутов) и Suppress Operations (Подавить вывод операций).

Создание пакетов

1. Щелкните правой кнопкой мыши на Логическом представлении браузера.
2. В открывшемся меню выберите пункт New > Package (Создать > пакет).
3. Назовите новый пакет Entities (Сущности).
4. Повторите этапы с первого по третий, создав пакеты Boundaries (границы) и Control (управление).

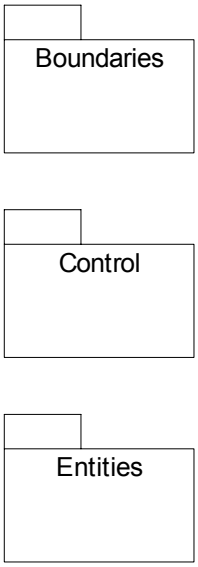


Рис. 6. Главная диаграмма классов системы обработки заказов

Создание Главной диаграммы классов

1. Дважды щелкните на Главной диаграмме классов прямо под Логическим представлением браузера, чтобы открыть ее.
2. Перетащите пакет Entities из браузера на диаграмму.
3. Перетащите пакеты Boundaries и Control из браузера на диаграмму.

Главная диаграмма классов должна выглядеть как на рис. 6. Для полноты картины проставьте на диаграмме две стрелки зависимостей: от пакета Boundaries к пакету Control и от пакета Control к пакету Entities.

Создание диаграммы классов, участвующих в реализации сценария "Ввести новый заказ"

1. Щелкните правой кнопкой мыши на Логическом представлении браузера.
2. В открывшемся меню выберите пункт New > Class Diagram (Создать > Диаграмму Классов).
3. Назовите новую диаграмму классов Add New Order.
4. Щелкните в браузере на этой диаграмме дважды, чтобы открыть ее.
5. Перетащите из браузера все классы (OrderOptions, OrderDetail, Order, OrderMgr и TransactionMgr).

Диаграмма классов должна выглядеть как на рис. 7.

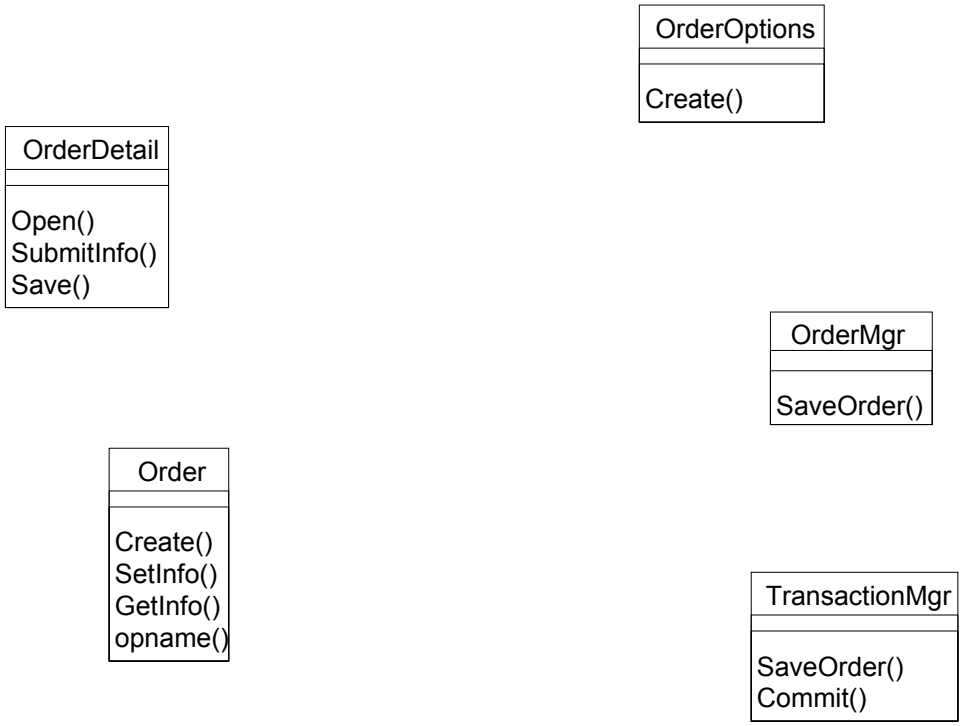


Рис. 7. Диаграмма классов Add New Order

Добавление стереотипов к классам

1. Щелкните правой кнопкой мыши на классе OrderOptions диаграммы.
2. В открывшемся меню выберите пункт Open Specification (Открыть спецификацию).
3. В поле стереотипа введите слово Boundary.
4. Нажмите на кнопку ОК.
5. Щелкните правой кнопкой мыши на классе OrderDetail диаграммы.
6. В открывшемся меню выберите пункт Open Specification (Открыть спецификацию).
7. В раскрывающемся списке в поле стереотипов теперь будет стереотип Boundary. Укажите его.
8. Нажмите на кнопку ОК.
9. Повторите этапы 1 – 4, связав классы OrderMgr и TransactionMgr со стереотипом Control, а класс Order – со стереотипом Entity.

Теперь диаграмма классов должна выглядеть как на рис. 8.

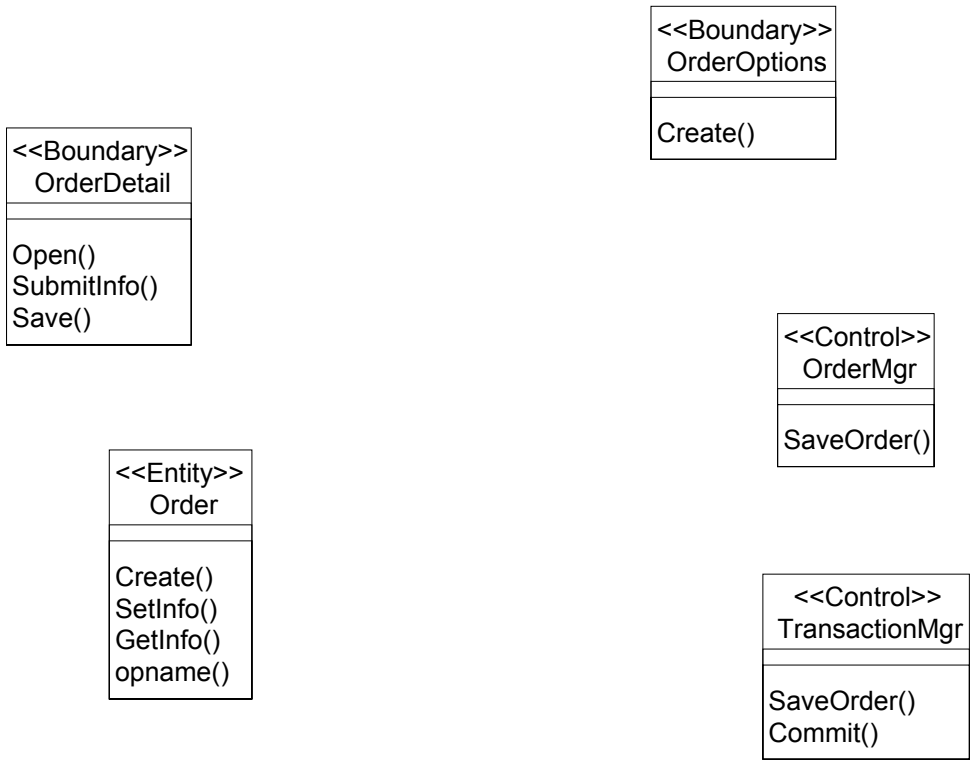


Рис. 8. Стереотипы классов для сценария Ввести новый заказ

Объединение классов в пакеты

1. Перетащите в браузере класс OrderOptions на пакет Boundaries.
2. Перетащите класс OrderDetail на пакет Boundaries.
3. Перетащите классы OrderMgr и TransactionMgr на пакет Control.
4. Перетащите класс Order на пакет Entities.

Добавление диаграмм классов к каждому пакету

1. Щелкните правой кнопкой на пакете Boundaries браузера.
2. В открывшемся меню выберите пункт New > Class Diagram (Создать > Диаграмму Классов).
3. Введите имя новой диаграммы — Main (Главная).
4. Дважды щелкните мышью на этой диаграмме, чтобы открыть ее.
5. Перетащите на нее из браузера классы OrderOptions и OrderDetail.
6. Закройте диаграмму.
7. Щелкните правой кнопкой на пакете Entities браузера.
8. В открывшемся меню выберите пункт New > Class Diagram (Создать > Диаграмму Классов).

9. Введите имя новой диаграммы — Main (Главная).
10. Дважды щелкните мышью на этой диаграмме, чтобы открыть ее.
11. Перетащите на нее из браузера класс Order.
12. Закройте диаграмму.
13. Щелкните правой кнопкой на пакете Control браузера.
14. В открывшемся меню выберите пункт New > Class Diagram (Создать > Диаграмму Классов).
15. Введите имя новой диаграммы — Main (Главная).
16. Дважды щелкните мышью на этой диаграмме, чтобы открыть ее.
17. Перетащите на нее из браузера классы OrderMgr и TransactionMgr.
18. Закройте диаграмму.

ПЗ 9-1. Создание диаграмм классов (добавление требований, классов, атрибутов, операций, детализация операций)

В упражнении 7 было создано несколько операций для классов нашей задачи. В предыдущем упражнении мы разместили в диаграмме классы. В этом упражнении к описаниям операций будут добавлены детали, включая параметры и типы возвращаемых значений. Кроме того, у классов будут определены атрибуты.

Постановка задачи

Положим, что дальнейшая детализация сценария "Ввести новый заказ" выявила необходимость дополнительного класса, отдельно описывающего каждый элемент заказа.

В силу этого пришлось обновить диаграмму последовательности, как показано на рис. 9.

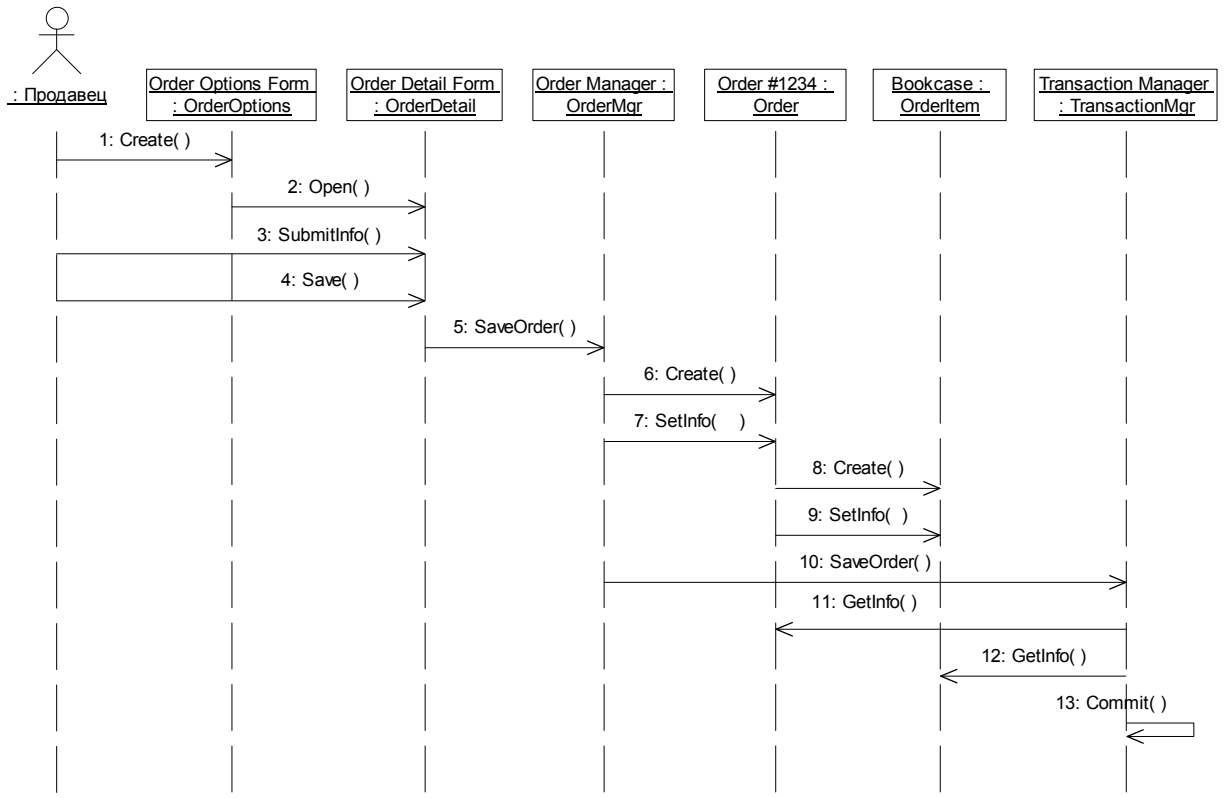


Рис. 9. Обновленная диаграмма последовательности

Кроме этого, потребовалось ввести пару новых атрибутов в класс Order.

Добавление класса, а также атрибутов и операций для существующих классов

Обновим содержание диаграммы классов "Ввести новый заказ". Для атрибутов и операций используем специфические для языка Java особенности. Установим параметры так, чтобы показывать все атрибуты, все операции и их сигнатуры. Видимость покажем с помощью нотации UML.

Этапы выполнения упражнения

Настройка

1. В меню модели выберите пункт Tools > Options.
2. Перейдите на вкладку Diagram.
3. Убедитесь, что переключатель Show Visibility помечен.
4. Убедитесь, что переключатель Show Stereotypes помечен.
5. Убедитесь, что переключатель Show Operation Signatures помечен.
6. Убедитесь, что переключатели Show All Attributes и Show All Operations помечены.
7. Убедитесь, что переключатели Suppress Attributes и Suppress Operations не помечены.

8. Перейдите на вкладку Notation.
9. Убедитесь, что переключатель Visibility as Icons не помечен.

Добавление нового класса

1. Найдите в браузере диаграмму классов "Add New Order".
2. Щелкните на ней дважды, чтобы ее открыть.
3. Нажмите кнопку Class панели инструментов.
4. Щелкните мышью внутри диаграммы, чтобы поместить там новый класс.
5. Назовите его OrderItem (ПозицияЗаказа).
6. Назначьте этому классу стереотип Entity.
7. В браузере перетащите класс в пакет Entities.

Добавление атрибутов

1. Щелкните правой кнопкой мыши на классе Order (Заказ).
2. В открывшемся меню выберите пункт New Attribute (Создать атрибут).
3. Введите новый атрибут OrderNumber : Integer (НомерЗаказа)
4. Нажмите клавишу Enter.
5. Введите следующий атрибут CustomerName : String (НаименованиеЗаказчика).
6. Повторите этапы 4 и 5, добавив атрибуты OrderDate : java.util.Date (ДатаЗаказа) и OrderFillDate : java.util.Date (ДатаЗаполненияЗаказа).
7. Щелкните правой кнопкой мыши на классе OrderItem.
8. В открывшемся меню выберите пункт New Attribute (Создать атрибут).
9. Введите новый атрибут ItemID : Integer (ИдентификаторПредмета).
10. Нажмите клавишу Enter.
11. Введите следующий атрибут ItemDescription : String (ОписаниеПредмета).

Добавление операций к классу OrderItem

1. Щелкните правой кнопкой мыши на классе OrderItem.
2. В открывшемся меню выберите пункт New Operation (Создать операцию).
3. Введите новую операцию Create.
4. Нажмите клавишу Enter.
5. Введите следующую операцию SetInfo
6. Нажмите клавишу Enter.
7. Введите следующую операцию GetInfo.

Подробное описание операций с помощью диаграммы Классов

1. Щелкните мышью на классе Order, выделив его таким способом.
2. Щелкните на этом классе еще один раз, чтобы переместить курсор внутрь.
3. Отредактируйте операцию Create(), чтобы она выглядела следующим образом: Create() : Boolean
4. Отредактируйте операцию SetInfo(), чтобы она выглядела следующим образом: SetInfo(OrderNum : Integer, Customer : String, OrderDate : java.util.Date, FillDate : java.util.Date) : Boolean
5. Отредактируйте операцию GetInfo(), чтобы она выглядела следующим образом: GetInfo() : String

Подробное описание операций с помощью браузера

1. Найдите в браузере класс OrderItem.
2. Чтобы раскрыть этот класс, щелкните на значке "+" рядом с ним. В браузере появятся его атрибуты и операции.
3. Дважды щелкните на операции GetIInfo(), чтобы открыть окно ее спецификации.
4. В раскрывающемся списке Return class (возвращаемый класс) укажите String.
5. Щелкните на кнопке OK, закрыв окно спецификации операции.
6. Дважды щелкните в браузере на операции SetInfo класса OrderItem, чтобы открыть окно ее спецификации.
7. В раскрывающемся списке Return class укажите Boolean.
8. Перейдите на вкладку Detail (Подробно).
9. Щелкните правой кнопкой мыши на белом поле в области аргументов, чтобы добавить туда новый параметр.
10. В открывшемся меню выберите пункт Insert. Rose добавит туда аргумент под названием argname.
11. Щелкните один раз на этом слове, чтобы выделить его, и измените имя аргумента на ID.
12. Щелкните на колонке Type, открыв раскрывающийся список типов. В нем выберите тип Integer.
13. Щелкните на колонке Default, чтобы добавить значение аргумента по умолчанию. Введите туда число 0.
14. Нажмите на кнопку OK, закрыв окно спецификации операции.

15. Дважды щелкните на операции Create() класса OrderItem, чтобы открыть окно ее спецификации.
16. В раскрывающемся списке Return class укажите Boolean.
17. Нажмите на кнопку ОК, закрыв окно спецификации операции.

Подробное описание операций с помощью любого из описанных методов

1. Используя браузер или диаграмму классов, введите следующую сигнатуру операций класса OrderDetail:
   Open() : Boolean
   SubmitInfo() : Boolean
   Save() : Boolean
2. Используя браузер или диаграмму классов, введите следующую сигнатуру операций класса OrderOptions:
   Create() : Boolean
3. Используя браузер или диаграмму классов, введите следующую сигнатуру операций класса OrderMgr:
   SaveOrder(OrderID : Integer) : Boolean
4. Используя браузер или диаграмму классов, введите следующую сигнатуру операций класса TransactionMgr:
   SaveOrder(OrderID : Integer) : Boolean
   Commit() : Integer

ПЗ 9-2. Создание диаграмм классов (добавление ассоциаций между классами)

В этом упражнении будут определены линии ассоциации между классами диаграммы "Ввести новый заказ".

Постановка задачи

Чтобы найти ассоциации, следует изучить диаграммы последовательности. Все обозначенные там связи между объектами, обеспечивающие пересылку сообщений, превращаются в линии ассоциаций между соответствующими классами.

Этапы выполнения упражнения

Настройка

1. Найдите в браузере диаграмму классов "Add New Order"
2. Дважды щелкните на ней, чтобы открыть ее.
3. Проверьте, имеется ли на панели инструментов диаграммы кнопка Unidirectional Association. Если ее нет, продолжайте настройку, выполнив этапы 4 и 5. Если есть, приступайте к выполнению самого упражнения.
4. Щелкните правой кнопкой мыши на панели инструментов диаграммы и в открывшемся меню выберите пункт Customize.
5. Добавьте на панель кнопку, называющуюся Create A Unidirectional Association.

Добавление ассоциаций

1. Нажмите кнопку панели инструментов Unidirectional Association.
2. Нарисуйте ассоциацию от класса OrderOptions к классу OrderDetail.
3. Повторите этапы 1 и 2, создав еще ассоциации:
    - От класса OrderDetail к классу OrderMgr
    - От класса OrderMgr к классу Order
    - От класса OrderMgr к классу TransactionMgr
    - От класса TransactionMgr к классу Order
    - От класса TransactionMgr к классу OrderItem
    - От класса Order к классу OrderItem
4. Щелкните правой кнопкой мыши на однонаправленной ассоциации между классами OrderOptions и OrderDetail, со стороны класса OrderOptions.
5. В открывшемся меню выберите пункт Multiplicity > Zero or One.
6. Щелкните правой кнопкой мыши на другом конце однонаправленной ассоциации.
7. В открывшемся меню выберите пункт Multiplicity > Zero or One.
8. Повторите этапы 4 – 7, добавив на диаграмму значения множественности для остальных ассоциаций, как показано на рис. 10.

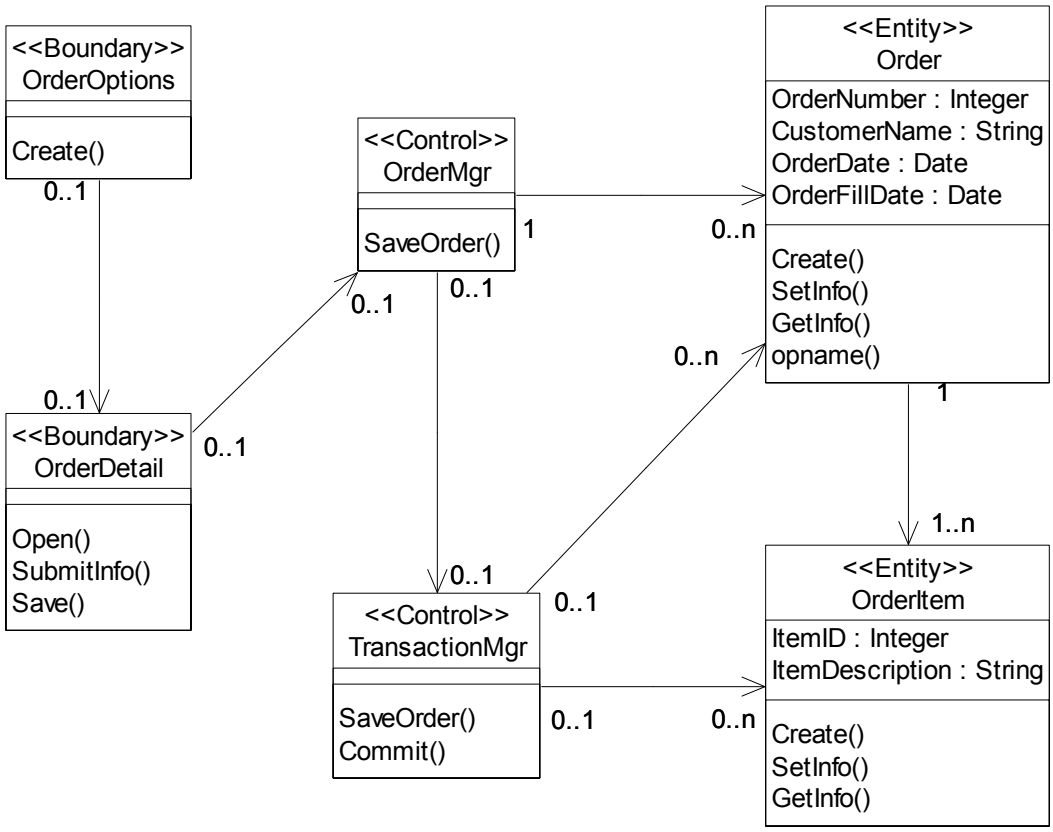


Рис. 10. Ассоциации в диаграмме классов "Add New Order"

ПЗ 10. Создание диаграммы конечного автомата

В этом упражнении будет создана диаграмма конечного автомата для класса Order.

Постановка задачи

При проектировании класса Order стало ясно, что он имеет сложное поведение. Многие требования к классу значительно менялись при изменении состояния его экземпляра. Например, заказы, выполнение которых было приостановлено, вели себя не так, как выполненные заказы, а те, в свою очередь, не так, как отмененные заказы.

Чтобы убедиться, что проектное решение удовлетворяет всем требованиям, целесообразно создание диаграммы конечного автомата для класса Order. С помощью этой диаграммы программисты смогут разобраться, как надо писать программный код для класса Order.

Разработайте диаграмму конечного автомата для класса Order, показанную на рис. 11.

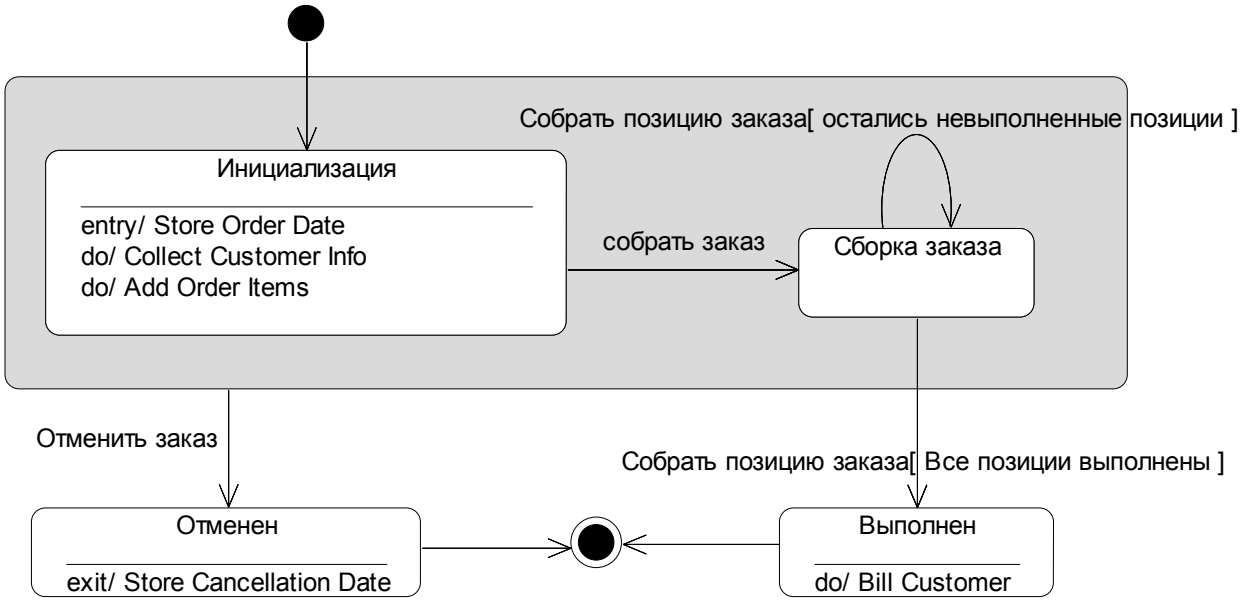


Рис. 11. Диаграмма конечного автомата для класса Order

Этапы выполнения упражнения

Создание диаграммы

1. Найдите в браузере класс Order.
2. Щелкните на классе правой кнопкой мыши и в открывшемся меню укажите пункт New, а в нем пункт Statechart Diagram. Назовите диаграмму State machine Order.

Добавление начального и конечного состояний

1. На панели инструментов нажмите кнопку Start State (Начальное состояние).
2. Поместите это состояние на диаграмму.
3. На панели инструментов нажмите кнопку End State (Конечное состояние).
4. Поместите это состояние на диаграмму.

Добавление композитного состояния

1. На панели инструментов нажмите кнопку State.
2. Поместите это состояние на диаграмму.

Добавление оставшихся состояний

1. На панели инструментов нажмите кнопку State.
2. Поместите это состояние на диаграмму.
3. Назовите состояние Cancelled (Отменен).
4. На панели инструментов нажмите кнопку State.
5. Поместите это состояние на диаграмму.
6. Назовите состояние Filled (Выполнен).
7. На панели инструментов нажмите кнопку State.

8. Поместите это состояние на диаграмму внутрь композитного состояния.
9. Назовите состояние Initialization (Инициализация).
10. На панели инструментов нажмите кнопку State.
11. Поместите это состояние на диаграмму внутрь композитного состояния.
12. Назовите состояние Pending (В ожидании, сборка заказа).

Подробное описание состояний

1. Дважды щелкните на состоянии Initialization (Инициализация).
2. Перейдите на страницу Actions.
3. Щелкните правой кнопкой мыши в окне страницы.
4. В открывшемся меню выберите пункт Insert (Вставить).
5. Дважды щелкните мышью на новом действии.
6. Назовите его Store Order Date (Сохранить дату заказа).
7. Убедитесь, что в окне When (Когда) указан пункт On Entry. Нажмите на кнопку ОК.
8. Повторите этапы 3 – 7, добавив следующие действия:
   Collect Customer Info (Собрать клиентскую информацию), в окне When указать пункт Do.
   Add Order Items (Добавить к заказу новые графы), в окне When указать Do.
9. Нажмите на кнопку ОК, чтобы закрыть спецификацию.
10. Дважды щелкните на состоянии Cancelled (Отменен).
11. Повторите этапы 2 – 7, добавив действие Store Cancellation Data (Сохранить дату отмены), указать пункт On Exit (на выходе)
12. Нажмите на кнопку ОК, чтобы закрыть спецификацию.
13. Дважды щелкните на состоянии Filled (Выполнен).
14. Повторите этапы 2 – 7, добавив действие Bill Customer (Счет заказчику), указать пункт Do.
15. Нажмите на кнопку ОК, чтобы закрыть спецификацию.

Добавление переходов

1. На панели инструментов нажмите кнопку Transition (Переход).
2. Щелкните мышью на начальном состоянии.
3. Проведите линию перехода к состоянию Initialization.
4. Повторите этапы с первого по третий, создав следующие переходы:
   От состояния Initialization к состоянию Pending.
   От состояния Pending к состоянию Filled.
   От композитного состояния к состоянию Cancelled.
   От состояния Cancelled к конечному состоянию.
   От состояния Filled к конечному состоянию.
5. На панели инструментов нажмите кнопку Transition to Self (Переход к себе).
6. Щелкните на состоянии Pending.

Подробное описание переходов

1. Дважды щелкните на переходе от состояния Initialization (Инициализация) к состоянию Pending (Сборка заказа), открыв окно его спецификации.
2. В поле Event (Событие) введите фразу Finalize order (Собрать заказ).
3. Щелкните на кнопке ОК, закрыв окно спецификации.
4. Повторите этапы с первого по третий, добавив событие Cancel order (Отменить заказ) к переходу между композитным состоянием и состоянием Cancelled (Отменен).
5. Дважды щелкните на переходе от состояния Pending (Сборка заказа) к состоянию Filled (Выполнен), открыв окно его спецификации.
6. В поле Event (Событие) введите фразу Add order item (Собрать позицию заказа).
7. Перейдите на страницу Detail (Подробно).
8. В поле Guard Condition (Условие) введите No unfilled items remaining (Все позиции выполнены).
9. Щелкните на кнопке ОК, закрыв окно спецификации.
10. Дважды щелкните мышью на рефлексивном переходе (Transition to Self) состояния Pending (Выполнение заказа приостановлено).
11. В поле Event (Событие) введите фразу Add order item (Собрать позицию заказа).
12. Перейдите на страницу Detail (Подробно).
13. В поле Guard Condition (Условие) введите Unfilled items remaining (Остались невыполненные позиции).
14. Щелкните на кнопке ОК, закрыв окно спецификации.

ПЗ 11-1. Создание диаграммы компонентов

В этом упражнении будет создана диаграмма компонентов для системы обработки заказов. На данный момент уже определены все классы, требуемые элемента Use Case "Ввести новый заказ". По мере реализации других элементов Use Case в диаграмму будут добавлять новые компоненты.

Постановка задачи

Завершив анализ и проектирование системы, переходим к разработке диаграммы компонентов. Выберем в качестве языка программирования Java и для каждого класса создадим соответствующие компоненты.

На рис. 12 показана главная диаграмма пакетов для компонентов.

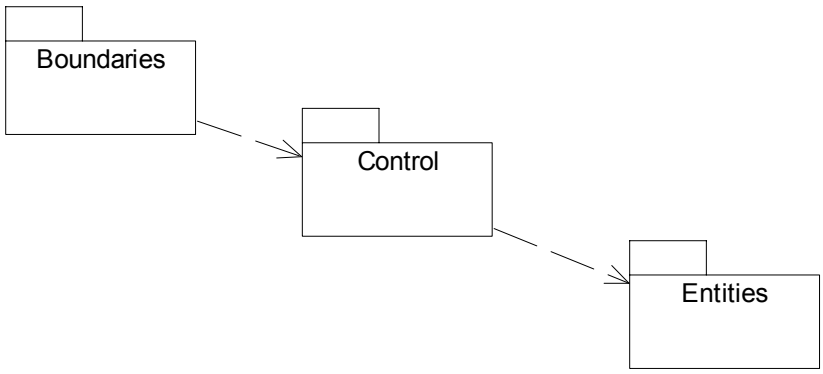


Рис. 12. Главная диаграмма пакетов для компонентов

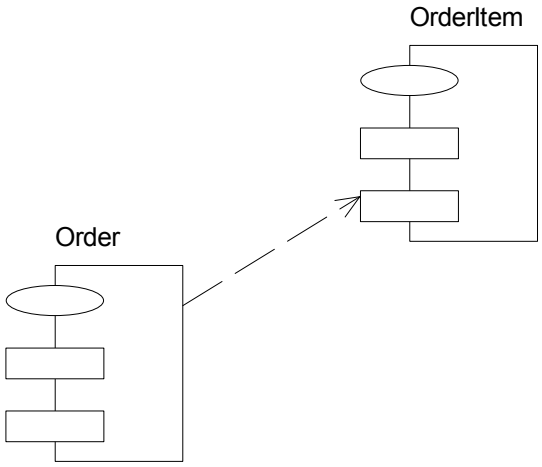


Рис. 13. Диаграмма компонентов пакета Entities

На рис. 13 показаны все компоненты пакета Entities. Эти компоненты содержат классы пакета Entities из Логического представления системы.

На рис. 14 показаны компоненты пакета Control. Они содержат классы пакета Control из Логического представления системы.

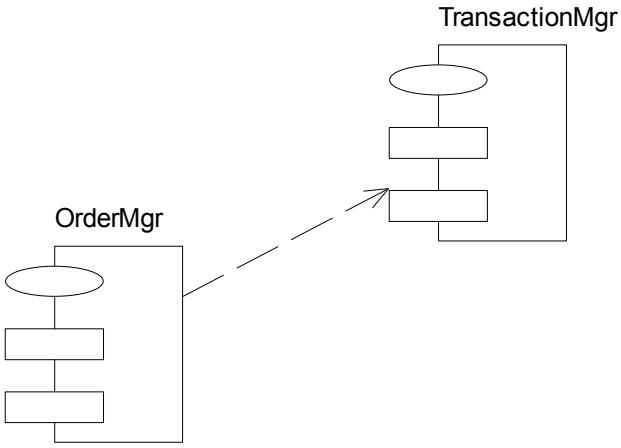


Рис. 14. Диаграмма Компонентов пакета Control

Наконец, на рис. 15 показаны компоненты пакета Boundaries. Они также соответствуют классам одноименного пакета из Логического представления системы.

На рис. 16 показаны все компоненты и поэтому мы назвали эту диаграмму диаграммой компонентов системы. На ней видны все зависимости между всеми компонентами проектируемой системы.

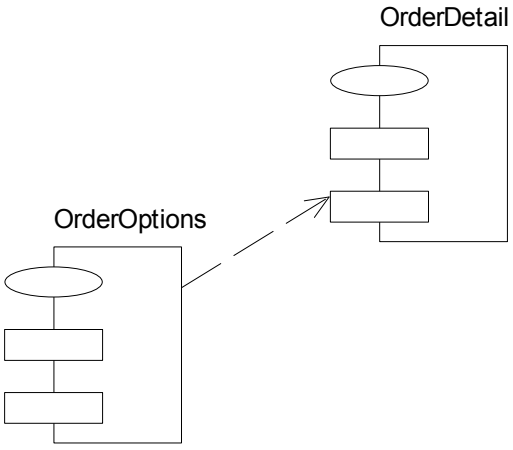


Рис. 15. Диаграмма Компонентов пакета Boundaries

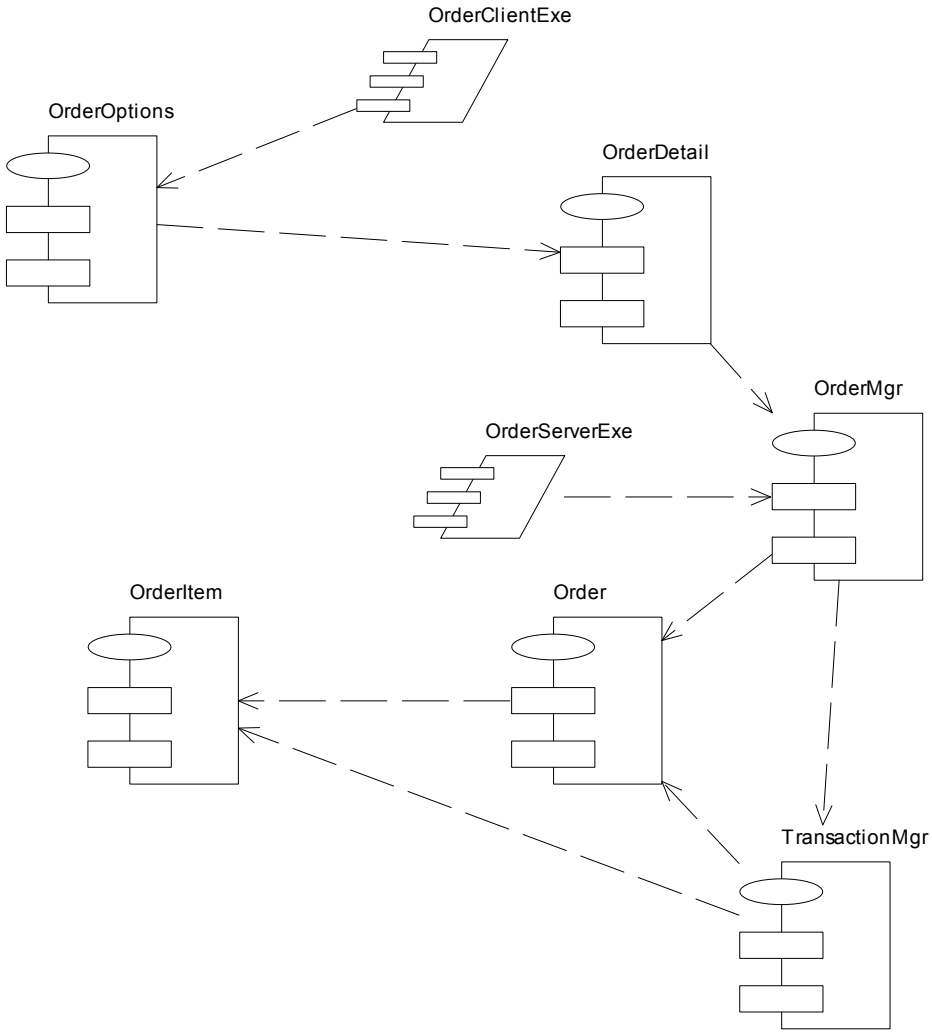


Рис. 16. Диаграмма компонентов системы

Этапы выполнения упражнения

Создание пакетов компонентов

1. Щелкните правой кнопкой мыши на представлении компонентов в браузере.
2. В открывшемся меню выберите пункт New > Package (Создать > пакет).
3. Назовите этот пакет Entities (Сущности).
4. Повторите этапы с первого по третий, создав пакеты Boundaries (Границы) и Control (Управление).

Добавление пакетов на Главную диаграмму пакетов для компонентов

1. Откройте Главную диаграмму пакетов для компонентов, дважды щелкнув на ней.
2. Перетащите пакеты Entities, Boundary и Control из браузера на Главную диаграмму.

Рисование зависимостей между пакетами

1. На панели инструментов нажмите кнопку Dependency (Зависимость).
2. Щелкните мышью на пакете Boundaries Главной диаграммы пакетов для компонентов.
3. Проведите линию зависимости до пакета Control.
4. Повторите этапы 1 – 3, проведя еще зависимость от пакета Control до пакета Entities.

Добавление компонентов к пакетам и рисование зависимостей

1. Дважды щелкните мышью на пакете Entities Главной диаграммы пакетов для компонентов, открыв Главную диаграмму компонентов этого пакета.
2. На панели инструментов нажмите кнопку Package Specification (Спецификация пакета).
3. Поместите спецификацию пакета на диаграмму.
4. Введите имя спецификации пакета OrderItem.
5. Повторите этапы 2 – 4, добавив спецификацию пакета Order.
6. На панели инструментов нажмите кнопку Dependency (Зависимость).
7. Щелкните мышью на спецификации пакета Order.
8. Проведите линию зависимости от него к спецификации пакета OrderItem.
9. С помощью описанного метода создайте следующие компоненты и зависимости:

Для пакета Boundaries:

Спецификацию пакета OrderOptions
Спецификацию пакета OrderDetail

Зависимости в пакете Boundaries:

От спецификации пакета OrderOptions до спецификации пакета OrderDetail.

Для пакета Control:

Спецификацию пакета OrderMgr
Спецификацию пакета TransactionMgr

Зависимости в пакете Control:

От спецификации пакета OrderMgr до спецификации пакета TransactionMgr

Создание диаграммы компонентов системы

1. Щелкните правой кнопкой мыши на представлении компонентов в броузере.
2. В открывшемся меню выберите пункт New > Component Diagram
3. Назовите новую диаграмму System.
4. Дважды щелкните на этой диаграмме.

Размещение компонентов на диаграмме компонентов системы

1. Если это еще не было сделано, разверните в броузере пакет компонентов Entities, чтобы открыть его.
2. Щелкните мышью на спецификации пакета Order в пакете компонентов Entities.
3. Перетащите эту спецификацию на диаграмму.
4. Повторите этапы 2 и 3, поместив на диаграмму спецификацию пакета OrderItem.
5. С помощью этого метода поместите на диаграмму следующие компоненты:

Из пакета компонентов Boundaries:

Спецификацию пакета OrderOptions
Спецификацию пакета OrderDetail

Из пакета компонентов Control:

Спецификацию пакета OrderMgr
Спецификацию пакета TransactionMgr

6. На панели инструментов нажмите кнопку Task Specification (Спецификация задачи).
7. Поместите спецификацию задачи на диаграмму и назовите ее OrderClientExe.

8. Повторите этапы 6 и 7 для спецификации задачи OrderServerExe.

Добавление оставшихся зависимостей на диаграмму компонентов системы

Уже существующие зависимости будут автоматически показаны на диаграмме компонентов системы после добавления туда соответствующих компонентов. Теперь надо добавить остальные зависимости.

1. На панели инструментов нажмите кнопку Dependency (Зависимость).
2. Щелкните на спецификации пакета OrderDetail.
3. Проведите линию зависимости к спецификации пакета OrderMgr.
4. Повторите этапы 1 – 3, создав следующие зависимости:
   - От спецификации пакета OrderMgr к спецификации пакета Order
   - От спецификации пакета TransactionMgr к спецификации пакета OrderItem
   - От спецификации пакета TransactionMgr к спецификации пакета Order
   - От спецификации задачи OrderClientExe к спецификации пакета OrderOptions
   - От спецификации задачи OrderServerExe к спецификации пакета OrderMgr

Связывание классов с компонентами

1. В Логическом представлении браузера найдите класс Order пакета Entities.
2. Перетащите этот класс на спецификацию пакета компонента Order в представлении Компонентов браузера. В результате класс Order будет соотнесен со спецификацией пакета компонента Order.
3. Повторите этапы 1 – 3, соотнеся с классами следующие компоненты:
   - Класс OrderItem со спецификацией пакета OrderItem
   - Класс OrderOptions со спецификацией пакета OrderOptions
   - Класс OrderDetail со спецификацией пакета OrderDetail
   - Класс OrderMgr со спецификацией пакета OrderMgr
   - Класс TransactionMgr со спецификацией пакета TransactionMgr

ПЗ 11-2. Создание диаграммы размещения

В этом упражнении будет создана диаграмма размещения для системы обработки заказов.

Постановка задачи

К данному моменту завершен весь предшествующий анализ и проектирование системы. Элементы Use Case, взаимодействия между объектами и компоненты описаны достаточно точно. Осталось определить, на каких компьютерах будут размещаться различные компоненты системы. В связи с этим пришлось разработать диаграмму размещения для системы обработки заказов.

Создание диаграммы размещения

Разработайте диаграмму размещения для системы обработки заказов. Готовая диаграмма должна выглядеть как на рис. 17.

Этапы выполнения упражнения

Добавление узлов к диаграмме размещения

1. Дважды щелкните мышью на представлении Размещения в браузере, чтобы открыть диаграмму размещения.
2. На панели инструментов нажмите кнопку Processor (Процессор).
3. Щелкните на диаграмме, поместив туда процессор.
4. Введите имя процессора "Сервер базы данных".
5. Повторите этапы 2 – 4, добавив следующие процессоры:
   Сервер приложения
   Клиентская рабочая станция №1
   Клиентская рабочая станция №2
6. На панели инструментов нажмите кнопку Device (Устройство).
7. Щелкните на диаграмме, поместив на нее устройство.
8. Назовите его "Принтер".

Добавление соединений

1. На панели инструментов нажмите кнопку Connection (Соединение).
2. Щелкните на процессоре "Сервер базы данных".
3. Проведите линию связи к процессору "Сервер приложения".
4. Повторите этапы 1 – 3, добавив следующие соединения:
   От процессора "Сервер приложения" к процессору "Клиентская рабочая станция №1"
   От процессора "Сервер приложения" к процессору "Клиентская рабочая станция №2"
   От процессора "Сервер приложения" к устройству "Принтер"

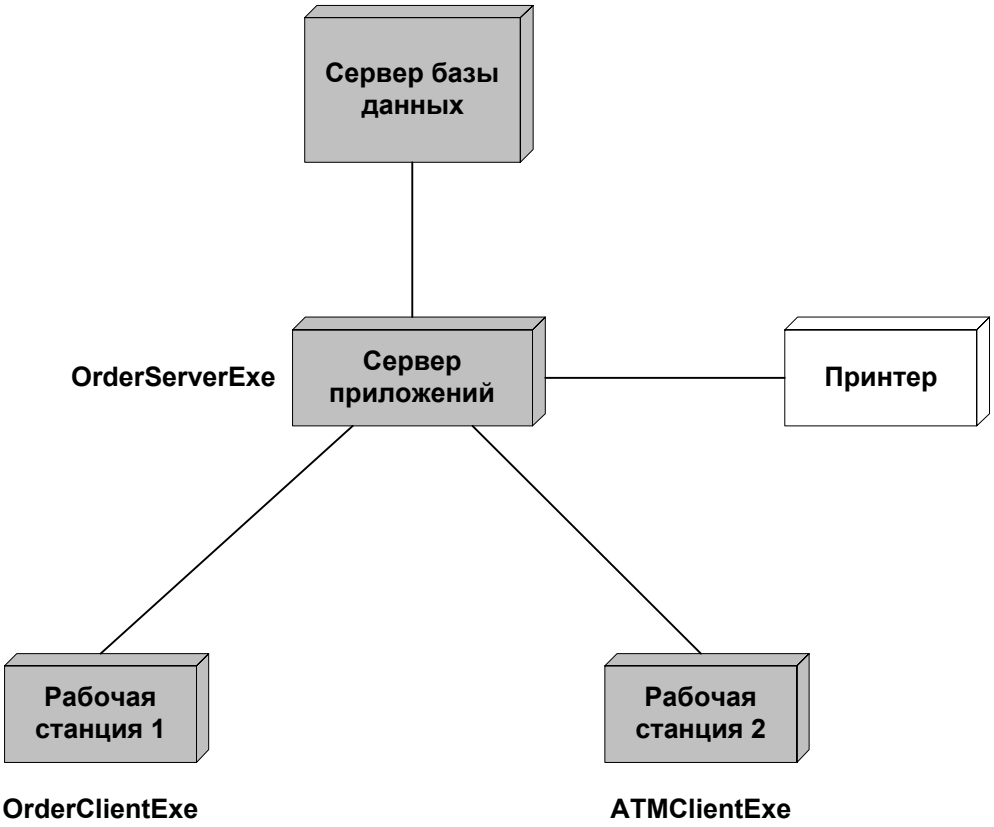


Рис. 17. Диаграмма Размещения для системы обработки заказов

Добавление процессов

1. Щелкните правой кнопкой мыши на процессоре "Сервер приложения" в браузере.
2. В открывшемся меню выберите пункт New > Process (Создать > Процесс).
3. Введите имя процесса OrderServerExe.
4. Повторите этапы 1 – 3, добавив еще процессы:
   На процессоре "Клиентская рабочая станция №1" — процесс OrderClientExe
   На процессоре "Клиентская рабочая станция №2" — процесс ATMClientEXE

Отображение процессов на диаграмме

1. Щелкните правой кнопкой мыши на процессоре "Сервер приложения".
2. В открывшемся меню выберите пункт Show Processes (Показать процессы).
3. Повторите этапы 1 и 2, показав процессы на следующих процессорах:
   Клиентская рабочая станция №1
   Клиентская рабочая станция №2.

ПЗ 12-1. Генерация Java-кода

Ранее была создана модель для системы обработки заказов (Order Entry). Теперь сгенерируем программный код Java для этой системы. При этом воспользуемся диаграммой компонентов системы, ранее представленной на рис. 16.

Для генерации программного кода необходимо выполнить описанные ниже шаги.

Этапы выполнения упражнения

Генерация программного кода Java

1. Откройте диаграмму компонентов системы.
2. Выберите все объекты на диаграмме.
3. Выберите Tools > Java/J2EE > Generate Java в меню.
4. В появившемся окне надо указать classpath для каждого из трех пакетов. Чтобы это сделать, сначала создайте classpath, указав, например, папку C:\Documents and Settings\student\Рабочий стол. Затем, выберите созданный classpath и выделите все пакеты (Select All), нажмите Assign, чтобы связать их.
5. Нажатие на OK запускает генерацию кода. Полученные заготовки исходных файлов ищите на рабочем столе.

ПЗ 12-2. Моделирование схемы базы данных

Проектирование реляционных баз данных выполняется с использованием средства Data Modeler. Его работа основана на известном механизме отображения объектной модели в реляционную. Результатом является построение диаграммы «сущность-связь» и последующая генерация описания БД на SQL. Перед выполнением упражнения модифицируйте диаграмму классов системы — укажите, что отношение между классами Order и OrderItem является ***ненаправленной композицией***.

Если среди проектных классов есть устойчивые, чьи экземпляры должны сохраняться в периодах между запусками системы, следует обеспечить сохранение их в базе данных и создать схему базы данных. Фактически, нужно отобразить объектную модель в реляционную. Одна из стратегий заключается в том, что для каждого устойчивого класса создается собственная таблица. Атрибуты класса переводятся в столбцы таблицы. Атрибут-идентификатор становится первичным ключом. Ассоциации моделируются с помощью связей между таблицами (связывающими значения первичного ключа записей одной таблицы со значениями внешнего ключа другой таблицы). Связи между таблицами всегда *двунаправленные*, по записям любой из связанных таблиц можно найти соответствующие записи другой таблицы. Связи между таблицами могут быть *идентифицирующими* и *не идентифицирующими*. Идентифицирующая связь указывает, что внешний ключ включает в себя часть первичного ключа. Связь отображается как композиция, если требуется указать на зависимость по существованию. В некоторых случаях для ассоциации (например, многие-ко-многим) требуется создавать таблицу, хранящую соединения между объектами. Для отображения обобщений используются разные способы. Один из них — *отдельная таблица для каждого класса*. В этом случае у всех сформированных таблиц будет один и тот же первичный ключ, который в таблицах подклассов будет также внешним ключом. В таблицах моделируются ограничения в виде «операций». Т. е. ограничения первичного ключа, ограничения внешнего ключа, ограничения индекса, ограничения уникальности указываются в разделе, предназначенном для операций.

Проектирование БД состоит из указанных ниже шагов.

Этапы выполнения упражнения

Создание нового компонента – базы данных

1. Щелкните правой кнопкой мыши на представлении компонентов.
2. В открывшемся меню выберите пункт Data Modeler > New > Database.
3. Откройте окно спецификации вновь созданного компонента DB_0 и в списке Target выберите Oracle 9.x.

Определение устойчивых (persistent) классов

1. Откройте окно спецификации класса Order в пакете Entities.
2. Перейдите на вкладку Detail.
3. Установите значение переключателя Persistence в Persistent.
4. Проделайте такие же действия для класса OrderItem.
5. Откройте класс Order в браузере, нажав “+”.
6. Щелкните правой кнопкой мыши на атрибуте orderNumber.
7. В открывшемся меню выберите пункт Data Modeler > Part of Object Identity (указание атрибута в качестве части первичного ключа).

Создание схемы БД

1. Щелкните правой кнопкой мыши на пакете Entities.
2. В открывшемся меню выберите пункт Data Modeler > Transform to Data Model.
3. В появившемся окне в списке Target Database укажите DB_0 и нажмите ОК. В результате в логическом представлении появится новый пакет Schemas.
4. Откройте пакет Schemas и щелкните правой кнопкой мыши на пакете <<Schema>> S_0.
5. В открывшемся меню выберите пункт Data Modeler > New > Data Model Diagram.
6. Откройте пакет, затем откройте вновь созданную диаграмму «сущность-связь» NewDiagram и перенесите на нее все классы-таблицы, находящиеся в пакете <<Schema>> S_0.

Получившаяся диаграмма показана на рис. 18.

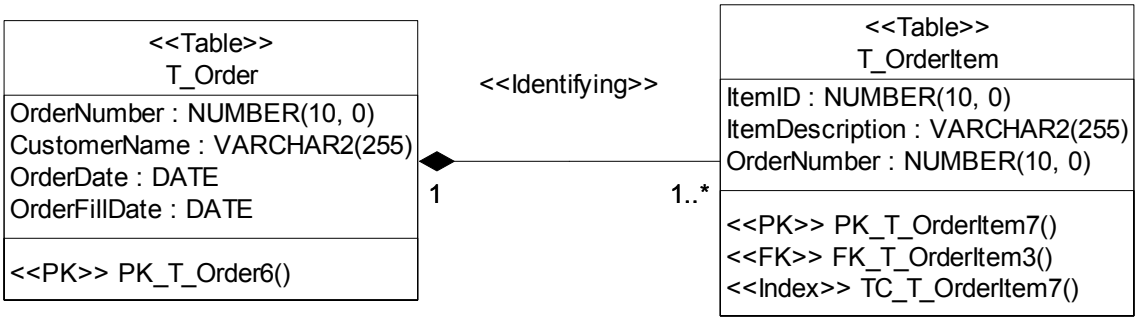


Рис. 18. Диаграмма «сущность-связь»

Замечание. Если сгенерирована не идентифицирующая связь между таблицами, ее надо преобразовать в идентифицирующую. [Напомним: идентифицирующая связь указывает, что внешний ключ включает в себя часть первичного ключа. Связь отображается как композиция.] Для этого на левом полюсе связи определяется включение по величине, а в поле «Type» спецификации связи выбирается значение «Identifying».

Генерация описания БД на SQL

1. Щелкните правой кнопкой мыши на пакете <<Schema>> S_0.
2. В открывшемся меню выберите пункт Data Modeler > Forward Engineer.
3. В открывшемся окне мастера Forward Engineering Wizard нажмите Next.
4. Отметьте все флажки генерации DDL и нажмите Next.
5. Укажите имя и расположение текстового файла с результатами генерации и нажмите Next.
6. После завершения генерации откройте созданный текстовый файл и просмотрите результаты.

ПРИЛОЖЕНИЯ

Банкомат

Постановка задачи для банкомата

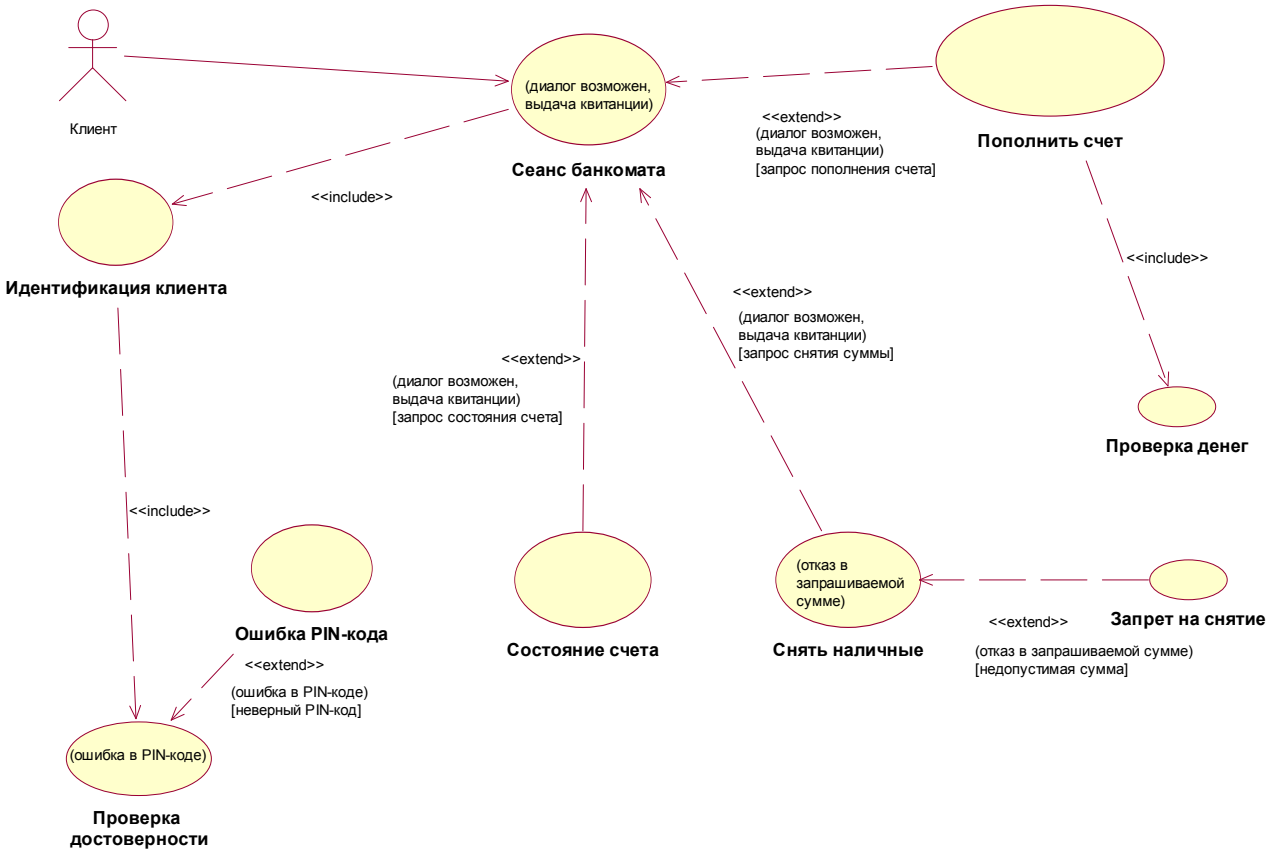
Банкомат — это автомат для выдачи / приема наличных денег по кредитным пластиковым карточкам. В его состав входят следующие устройства: дисплей, панель управления с кнопками, приемник кредитных карт, хранилище денег, лоток для приема денег, лоток для выдачи денег, хранилище конфискованных кредитных карт, хранилище конфискованных банкнот, принтер для печати справок.
Банкомат подключен к линии связи для обмена данных с банковским компьютером, хранящим сведения о счетах клиентов.
Обслуживание клиента начинается с момента помещения пластиковой карточки в банкомат. После распознавания типа пластиковой карточки, банкомат выдает на дисплей приглашение ввести PIN-код. PIN-код представляет собой четырехзначное число. Затем банкомат проверяет правильность введенного кода. Если код указан неверно, пользователю предоставляются еще две попытки для ввода правильного кода. В случае повторных неудач карта перемещается в хранилище карт, и сеанс обслуживания заканчивается. После ввода правильного кода банкомат предлагает пользователю выбрать операцию. Клиент может либо снять наличные со счета, либо пополнить счет, либо узнать остаток на его счету.
При снятии наличных со счета банкомат предлагает указать сумму (10, 50, 100, 200, 500 EUR). После выбора клиентом суммы банкомат запрашивает, нужно ли печатать справку по операции. Затем банкомат посылает запрос на снятие выбранной суммы центральному компьютеру банка. В случае получения разрешения на операцию, банкомат проверяет, имеется ли требуемая сумма в его хранилище денег. Если он может выдать деньги, банкомат выдает указанную сумму в лоток выдачи. Банкомат печатает справку по произведенной операции, если она была затребована клиентом.
При пополнении счета клиент помещает банкноты в лоток для приема денег, банкомат проверяет достоверность купюр. Если проверка прошла успешно, счет пополняется, на дисплей или печать выводится квитанция о приеме денег. Если проверка не прошла успешно, фальшивые купюры изымаются, а остальные возвращаются. На дисплей выводится информационное сообщение.
Если клиент хочет узнать остаток на счету, то банкомат посылает запрос центральному компьютеру банка и выводит сумму на дисплей. По требованию клиента печатается и выдается соответствующая справка.

Моделирование предметной области

Словарь понятий предметной области банкомата

Дополнительная спецификация

Создание диаграммы Use Case



Система регистрации для института

Постановка задачи

Система должна обеспечить регистрацию студентов на курсы, а также просмотр их табелей успеваемости из локальной сети института. Профессора должны иметь доступ к онлайновой системе, чтобы указывать курсы, которые они будут читать, и проставлять оценки успеваемости. Профессора должны иметь возможность просматривать список студентов, записавшихся на их курсы (также в режиме онлайн).
База данных курсов (их каталог) поддерживается реляционной СУБД.
В начале каждого семестра студенты могут запросить каталог курсов, содержащий список курсов, предлагаемых в данном семестре. Информация о каждом курсе должна включать имя профессора, наименование кафедры и требования к предварительному уровню подготовки (предварительно прослушанным курсам).
Система должна позволять студентам выбирать по 4 курса в следующем семестре. Дополнительно каждый студент может указать 2 альтернативных курса (на тот случай, если какой-либо из

выбранных им курсов окажется уже заполненным или отмененным). На каждый курс может записаться не более 20 и не менее 10 студентов (если менее 10, то курс отменяется). В каждом семестре объявляется период времени, когда студенты могут изменить свои планы. В это время студенты должны иметь доступ к системе, чтобы добавить или удалить выбранные курсы. После того, как процесс регистрации некоторого студента завершен, система регистрации направляет информацию в расчетную систему, чтобы студент мог внести плату за семестр. Если курс окажется заполненным в процессе регистрации, студент должен быть извещен об этом до окончательного формирования его личного учебного плана.

В конце семестра студенты должны иметь доступ к системе для просмотра своих электронных табелей успеваемости. Поскольку эта информация конфиденциальная, система должна обеспечивать ее защиту от несанкционированного доступа.

Моделирование предметной области

Словарь понятий предметной области системы регистрации

Курс (Course)	Учебный курс по некоторому предмету, который может быть прочитан в университете. Курсы различаются названиями, длительностью. У курса могут быть требования к предварительно прослушанным курсам
Предлагаемый курс (Course Offering)	Запись о курсе, предлагаемом для чтения в конкретном семестре (один и тот же курс может вестись в нескольких разных семестрах). Включает день недели и номер пары, когда будут проходить лекции
Каталог курсов (Course Catalog)	Внешняя система, у которой можно запросить перечень всех курсов института
Расчетная система (Billing System)	Внешняя система, в которую передаются сведения для формирования счетов за обучение
Оценка (Mark)	Количество баллов (от 2 до 10), полученных студентом за конкретный курс
Профессор (Professor)	Пользователь системы регистрации. Лектор произвольного количества курсов в течение семестра. Отмечает в системе читаемые им курсы, ставит оценки
Табель успеваемости (Report Card)	Все оценки за все курсы, полученные студентом в данном семестре
Список курса (Roster)	Список всех студентов, записавшихся на конкретный курс
Студент (Student)	Пользователь системы регистрации. Учащийся, который выбирает перечень курсов для обучения в течение семестра. Основные сведения о студенте — имя и почтовый адрес (для отправки счетов за обучение)
План-график (Schedule)	Набор предлагаемых курсов, выбранных студентом в некотором семестре. План-график включает в себя 4 основных и 2 альтернативных курса

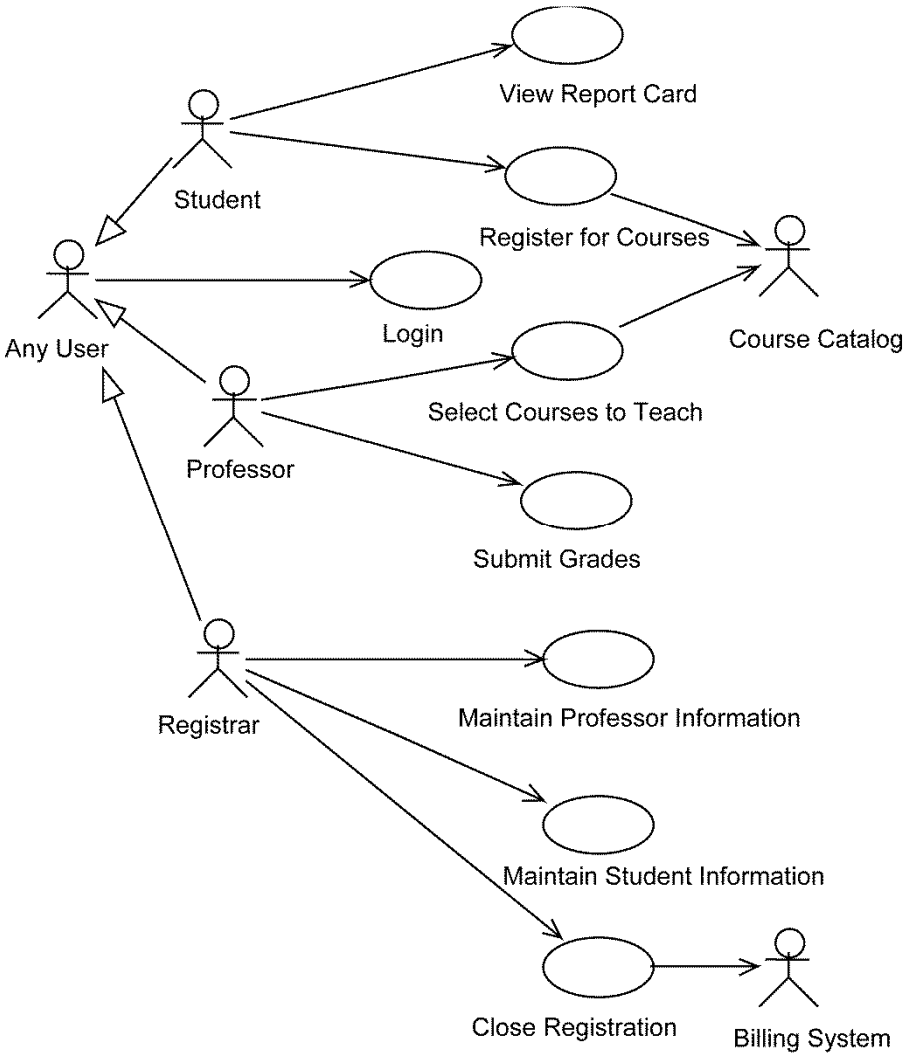
Актеры:

- Student (Студент) – записывается на курсы;
- Professor (Профессор) – выбирает курсы для преподавания;
- Registrar (Регистратор) – формирует учебный план и каталог курсов, ведет все данные о курсах, профессорах и студентах;
- Billing System (Расчетная система) – получает от данной системы информацию по оплате за курсы;
- Course Catalog (Каталог курсов) – передает в систему информацию из каталога курсов, предлагаемых университетом.

Элементы Use Case:

- Login (Войти в систему);
- Register for Courses (Зарегистрироваться на курсы);
- View Report Card (Просмотреть табель успеваемости);
- Select Courses to Teach (Выбрать курсы для преподавания);
- Submit Grades (Проставить оценки);
- Maintain Professor Information (Вести информацию о профессорах);
- Maintain Student Information (Вести информацию о студентах);
- Close Registration (Закрыть регистрацию).

Создание диаграммы Use Case



Элемент Use Case Register for Courses

Краткое описание

Данный элемент Use Case позволяет студенту зарегистрироваться на конкретные курсы в текущем семестре. Студент может изменить свой выбор (обновить или удалить курсы), если изменение выполняется в установленное время в начале семестра. Система каталога курсов предоставляет список всех конкретных курсов текущего семестра.

Основной поток событий

1. Студент сообщает о желании зарегистрироваться на курсы.
2. Система подтверждает, что регистрация на курсы в текущем семестре открыта.
3. Система запрашивает связь с каталогом курсов.
4. Каталог курсов подтверждает, что связь установлена.
5. Система запрашивает требуемое действие (создать график, обновить график, удалить график, утвердить график).
6. Студент сообщает системе свой выбор.
7. Система подтверждает, что требуемое действие можно выполнить.
8. Согласно выбору студента выполняется один из подчиненных потоков (создать, обновить, удалить или утвердить график).
9. Система заканчивает сеанс связи с каталогом курсов.
10. Каталог курсов подтверждает, что сеанс закончен.

Подчиненные потоки событий:

8А. Создать график

1. Система запрашивает у каталога курсов список доступных конкретных курсов.
2. Каталог курсов передает системе запрашиваемый список.
3. Система выводит список курсов и пустой план-график.
4. Студент выбирает из списка 4 основных курса и 2 альтернативных курса.
5. Система создает график студента и заносит в него выбранные курсы, помечая их как незафиксированные.
6. Система сообщает, что создание графика завершено.
7. Выполнение переходит на шаг 9 основного потока.

8Б. Обновить график

1. Система выводит текущий график студента.
2. Система запрашивает у каталога курсов список доступных конкретных курсов.
3. Каталог курсов передает системе запрашиваемый список.
4. Система выводит список курсов.
5. Студент обновляет свой выбор курсов, удаляя или добавляя конкретные курсы.
6. Система обновляет график в соответствии с пожеланиями студента. Для каждого зафиксированного курса, удаленного из графика, система удаляет студента из списка студентов, записавшихся на курс. Каждый добавленный курс система помечает как незафиксированный.
7. Система сообщает, что обновление графика завершено.
8. Выполнение переходит на шаг 9 основного потока.

8В. Удалить график

1. Система выводит текущий график студента.
2. Система запрашивает у студента подтверждения удаления графика.
3. Студент подтверждает удаление.
4. Система удаляет график. Для каждого зафиксированного курса из удаляемого графика, система удаляет студента из списка студентов, записавшихся на курс.
5. Выполнение переходит на шаг 9 основного потока.

8Г. Утвердить график

1. Для каждого незафиксированного курса в графике выполняется:
    1.1.1. Система подтверждает выполнение студентом предварительных требований (прохождение определенных курсов), и подтверждает, что курс открыт для регистрации, и отсутствуют конфликты (в графике не должно быть зафиксированного курса, читаемого в тот же день и на той же паре, что и проверяемый курс).
    1.1.2. Система добавляет студента в список записавшихся на курс.
    1.1.3. Система помечает курс в графике как зафиксированный.
    1.1.4. Система помечает курс как закрытый, если в списке студентов содержится 20 записей.
2. Система сообщает студенту результаты утверждения графика.
3. Выполнение переходит на шаг 9 основного потока.

Альтернативные потоки

2А. Регистрация на курсы закрыта

1. Система обнаруживает, что регистрация на курсы в текущем семестре закрыта.
2. Система выдает сообщение об ошибке.
3. Вариант использования завершается.

3А. Каталог курсов недоступен

1. Система обнаруживает, что невозможно установить связь с каталогом курсов.
2. Система выдает сообщение об ошибке.
3. Вариант использования завершается.

8Г.1.1А. Не выполнены предварительные требования, курс заполнен, или имеют место конфликты графика

1. Система обнаруживает, что студент не выполнил необходимые предварительные требования, или выбранный им конкретный курс заполнен, или имеют место конфликты графика.
2. Система выдает сообщение об ошибке.
3. Система переходит к следующему незафиксированному курсу и продолжает выполнение потока «Утвердить график»

7А. График не найден

1. Система обнаруживает, что требуемое действие (обновить, удалить или утвердить график) нельзя выполнить, так как график студента на текущий семестр отсутствует.
2. Система выдает сообщение об ошибке.
3. Выполнение переходит на шаг 5 основного потока.

7Б. График найден

1. Система обнаруживает, что требуемое действие (создать график) нельзя выполнить, так как график студента на текущий семестр создан ранее.
2. Система выдает сообщение об ошибке.
3. Выполнение переходит на шаг 5 основного потока.

8В.3А. Удаление отменено

1. Студент отменяет удаление графика.
2. Выполнение переходит на шаг 5 основного потока.

Предусловия

Перед началом выполнения данного элемента Use Case студент должен войти в систему.

Постусловия

Если элемент Use Case завершится успешно, система создаст, обновит, удалит или утвердит график студента в соответствии с выбором пользователя. В противном

случае гарантируется что: при закрытой регистрации изменения в графики студентов не производятся; при недоступном каталоге курсов изменения в графики студентов не производятся; при утверждении графика студента игнорируются курсы, для которых не выполнены предварительные требования, или которые закрыты, или которые вызывают конфликты в графике; при отсутствии графика на текущий семестр его обновление, удаление или утверждение не производятся; при наличии графика на текущий семестр добавление еще одного графика не производится.