Лабораторная работа №7

Тема: Построение диаграммы последовательности и диаграммы кооперации в среде Rational Rose.

Цель: Научить строить диаграммы диаграммы последовательности и диаграммы кооперации с использованием пакета Rational Rose.

Литература: Вендров А.М. Проектирование программного обеспечения экономических информационных систем

Задание: Изучите методические указания и рекомендации по выполнению работы, выполните практические задания, ответьте на контрольные вопросы.

Методические указания и рекомендации по выполнению работы

Диаграммы последовательности

Как правило, поток событий описывает не одну последовательность действий, а несколько возможных, это отражается наличием главного потока событий и альтернативных потоков. Чаще всего невозможно описать прецедент с помощью только одной последовательности действий. Например, для прецедента Заказ товаров возможно оформление заказа без изменения корзины, с изменением состава корзины, или покупатель, просмотрев корзину, захочет вернуться в каталог и что-то в нее добавить, возможно, вернувшись в каталог, покупатель не станет ничего больше добавлять, а снова вернется в корзину и оформит заказ. Каждый такой вариант мы можем описать своей последовательностью действий, своим сценарием. И, таким образом, один прецедент описывает несколько последовательностей – сценариев, каждый из которых описывает один из вариантов возможного потока событий.

Сценарий (Scenario) – это некоторая последовательность действий, иллюстрирующая поведение системы .

Сценарий – это экземпляр потока событий. Он представляет собой одиночный проход по потоку событий для прецедента. Для графического отображения сценария используются диаграммы последовательностей.

Диаграмма последовательности действий отображает взаимодействие объектов, упорядоченное по времени.

Основные элементы нотации диаграмм последовательности

На диаграммах последовательности изображаются объекты, классы и последовательность сообщений, которыми обмениваются объекты в ходе выполнения сценария (рисунок 1).

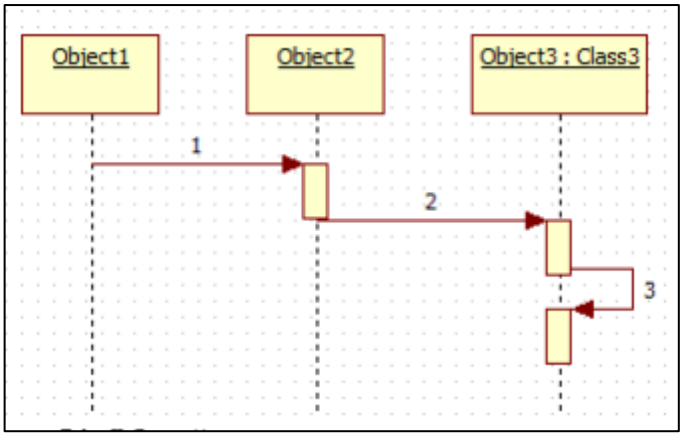

Рисунок 1 – Общий вид диаграммы последовательности

На диаграмме последовательностей могут также изображаться экземпляры действующих лиц. Для того чтобы поместить действующее лицо на диаграмму, нужно найти его в навигаторе модели справа и перетащить на поле диаграммы последовательностей (рисунок 2).

Пример.

Рисунок 2 – Экземпляр действующего лица на диаграмме последовательности

Действующие лица, присутствующие на диаграммах взаимодействия, выделяются из потока событий как сущности, запускающие процессы. На одной диаграмме их может быть несколько.

Для того чтобы поместить экземпляр уже созданного ранее на диаграмме прецедентов действующего лица на диаграмму взаимодействия, просто перетащите его с навигатора модели на рабочее поле диаграммы.

Каждый объект или действующее лицо на диаграмме последовательностей имеет свою линию жизни, которая обозначается пунктиром.

Линия жизни объекта (object lifeline) – вертикальная пунктирная линия на диаграмме последовательности, которая представляет существование объекта в течение определенного периода времени.

Фокус управления (активность, focus of control) - специальный символ на диаграмме последовательности, указывающий период времени, в течение которого объект выполняет некоторое действие, находясь в активном состоянии.

Фокус управления изображается тонким прямоугольником, расположенным на линии жизни (рисунок 3).

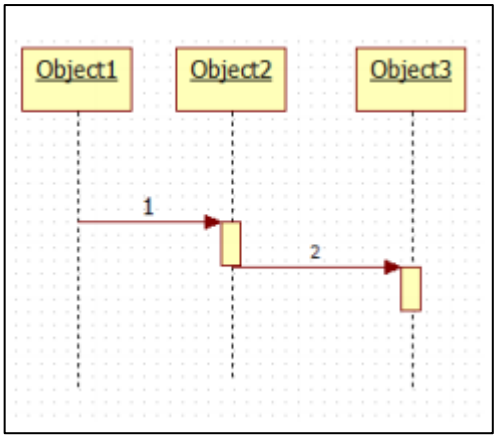

Рисунок 3 – Фокус управления

Иногда отображение фокуса активности и нумерации сообщений на диаграмме могут сделать ее трудной для чтения. Чтобы фокус управления и нумерация сообщений не отображались на диаграмме последовательности в StarUML нужно открыть редактор свойств этой диаграммы в инспекторе модели и в разделах ShowSequenceNumber и ShowActivation убрать «галочки» (рисунок 4).

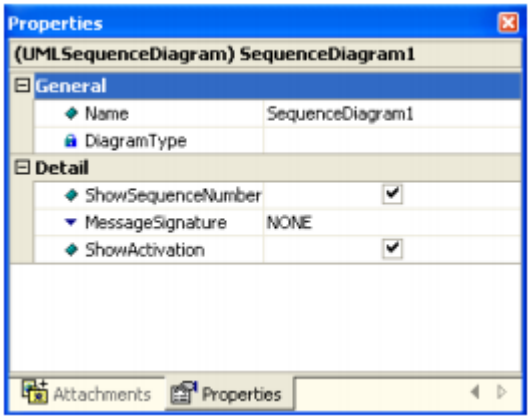

Рисунок 4 – Управление отображением фокуса управления и нумерации сообщений

Объекты и действующие лица на диаграммах последовательности обмениваются сообщениями. Сообщения обозначаются стрелками, идущими от отправителя к получателю.

Сообщение (message) — спецификация передачи информации от одного элемента модели к другому с ожиданием выполнения определенных действий со стороны принимающего элемента (рисунок 5).

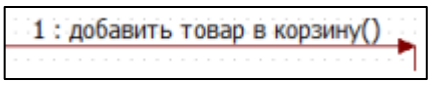

Рисунок 5 – Сообщение

Для сообщений на диаграммах последовательностей, как и для других элементов модели, доступен ряд спецификаций.

Во-первых, у каждого сообщения должно быть имя, соответствующее его цели.

Во-вторых, сообщения на диаграммах последовательностей можно соотнести с операциями, определенными для классов. Если от одного объекта к другому направлено сообщение, то это означает, что объект-источник вызывает операцию объекта-приемника. Объект не может вызвать произвольную операцию: она должна быть доступна этому объекту.

В особых случаях сообщение не становится операцией: например, ввод логина и пароля подразумевает их печать в соответствующих полях, и сообщение будет реализовано в виде поля ввода в окне программы.

Процедура создания операций из сообщений будет описана ниже.

В-третьих, мы можем для каждого сообщения установить тип синхронизации. Каждому типу соответствует его обозначение.

Вызов операции (процедуры) (call) вызывает операцию того объекта, к которому направлено. Объект может вызвать свою операцию. Тогда стрелка начинается и заканчивается на линии жизни одного и того же объекта, такое сообщение называется рефлексивным.

Синхронное сообщение обозначается стрелкой с закрашенным наконечником (рисунок 6).

Рисунок 6 – Синхронное сообщение

Асинхронное сообщение (send) посылает объекту сигнал. При этом источник не ждет отклика приемника или подтверждения получения, а продолжает свою работу. Обозначается нежирной стрелкой (рисунок 7).

Рисунок 7 – Асинхронное сообщение

Ответное сообщение (return) возвращает значение из процедуры тому объекту, к которому направлено. Обозначается пунктирной стрелкой (рисунок 8)

Рисунок 8 – Ответное сообщение

Создать объект (create) – создает новый объект. Обозначается стрелкой со стереотипом <> (рисунок 9).

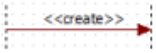

Рисунок 9 – Создание объекта

Уничтожить объект (destroy)- удаляет объект. Объект может уничтожить сам себя. Обозначается стрелкой со стереотипом <>. При уничтожении объекта на его линии жизни появляется символ разрушения, который обозначается крестом (рисунок 10).

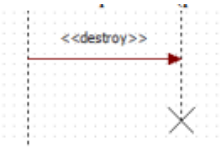

Рисунок 10 – Уничтожение объекта

Для определения типа сообщения в StarUML нужно выполнить след действия: выделите сообщение, щелкнув по соответствующей стрелке один раз левой кнопкой мыши, откройте редактор свойств, выберите на нем раздел ActionKind и в выпадающем списке выберите тот тип синхронизации, который вы хотите установить (рисунок 11).

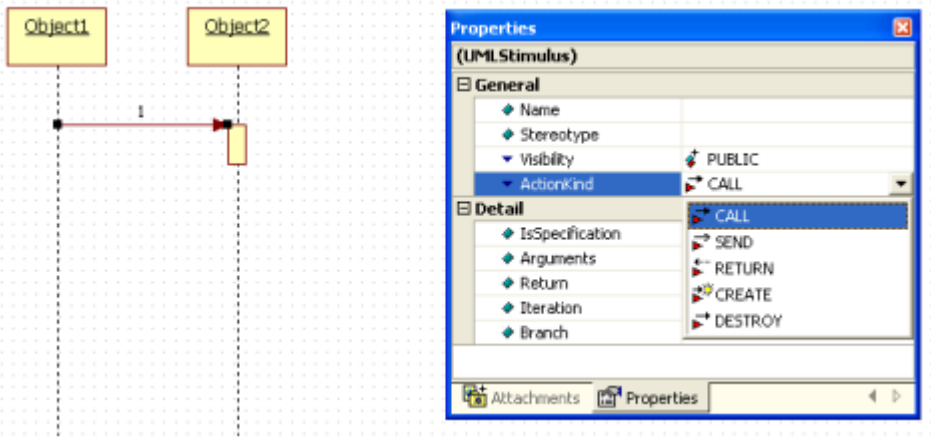

Рисунок 11 – Выбор типа сообщения

Взаимосвязь диаграмм классов и последовательности

Процесс построения модели системы является итеративным. Особенно хорошо это можно видеть при создании диаграмм классов и последовательности. Какую диаграмму создавать первой: классов или последовательности? Одни разработчики начинают с диаграмм классов, другие – наоборот, с последовательности. И в том и в другом случае, скорее всего, обе эти диаграммы, построенные для одного сценария, будут в дальнейшем подвергаться изменению. После построения диаграмм последовательности на диаграммах классов могут появиться новые классы, а на диаграммах последовательности – новые объекты, которых раньше там не

было, но они придут туда из диаграмм классов. Возможно, что некоторые объекты и классы будут, напротив, удалены.

Пример. В соответствии с нашей диаграммой последовательности на диаграмме классов сценария Оформить заказ произойдут некоторые изменения.

Не сложно видеть на диаграмме последовательностей, что покупатель участвует в данном сценарии как действующее лицо-инициатор, запускающий выполнение сценария, но не как внутренний объект системы. Поэтому класс Customer (Покупатель) с данной диаграммы классов удалим: скорее всего такой класс в нашей модели будет (и мы удалили его только с диаграммы), но классом сценария Оформление заказа он не является.

Диаграмма классов прецедента Оформление заказа изменится и будет выглядеть так (рисунок 12).

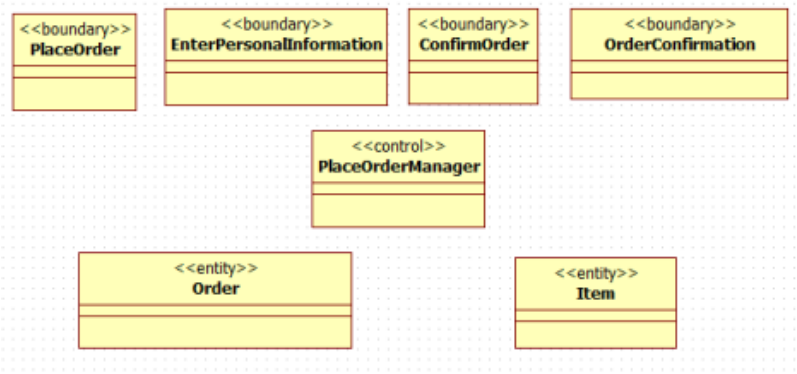

Рисунок 12 – Измененная диаграмма классов сценария Оформление заказа

Замечание. Для создания диаграммы последовательностей, мы могли каждый объект этой диаграммы не создавать заново, а воспользоваться методом перетаскивания. Если перетащить класс с навигатора модели на диаграмму последовательности, то будет создан анонимный объект этого класса (рисунок 13).

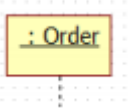

Рисунок 13 – Анонимный объект класса Order

Можно изменить имя объекта, присвоив ему имя (рисунок 14).

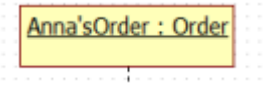

Рисунок 14 – Именованный объект класса Order

Кооперативные диаграммы

Диаграмма кооперации – это альтернативный способ изображения сценария варианта использования. Этот тип диаграмм заостряет внимание на связях между объектами, отображая обмен данными в системе. А диаграммы последовательности отображают взаимодействие объектов во времени, поэтому ее следует читать сверху вниз и слева направо.

Диаграммы кооперации содержат все те же элементы, что и диаграммы последовательности: объекты, действующие лица, связи между ними и сообщения, которыми они обмениваются, но они уже не упорядочены во времени.

Пример выполнения задания:

Добавление диаграммы последовательности в модель

Для создания новой диаграммы последовательности нужно выполнить следующие шаги: щелкнуть правой кнопкой мыши по папке представления Logical View в навигаторе модели, в контекстном меню выбрать пункт Add Diagram, в списке выбрать диаграмму последовательности Sequence Diagram (рисунок 15).

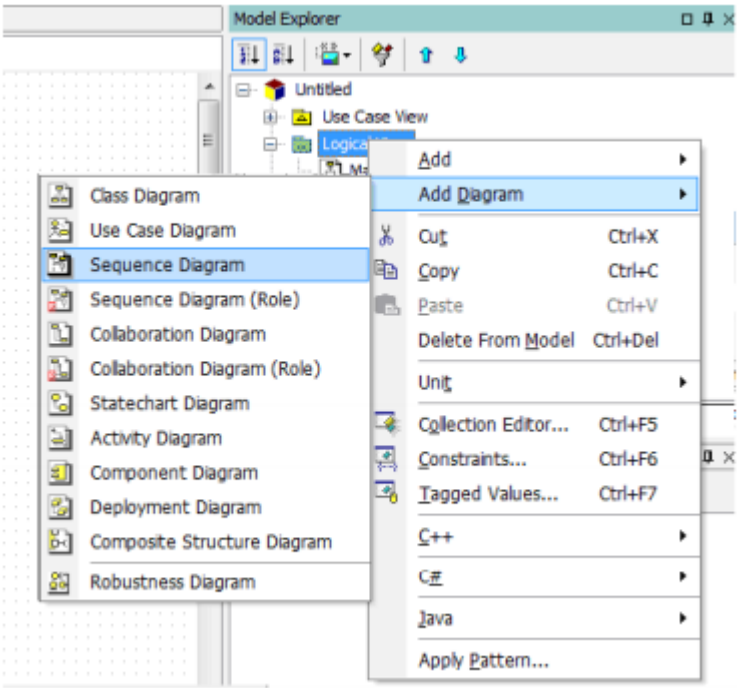

Рисунок 15 – Добавление диаграммы последовательности

Мы также можем использовать диаграмму последовательности для детализации прецедента. Для этого нужно связать диаграмму с прецедентом: для создания диаграммы щелкните правой кнопкой мыши по прецеденту, а не по папке Logical View. Однако, если мы строим диаграмму последовательности для анализа системы, то лучше все-таки помещать ее в Logical View.

Пример. Мы уже определили классы сценария Оформление заказа, теперь с помощью диаграммы последовательности покажем, как взаимодействуют объекты этих классов во времени.

Составим диаграмму последовательности для случая, когда покупатель успешно оформляет заказ (рисунок 16).

Покупатель выбирает опцию «Оформить заказ» (place order), при этом вызывается некоторый объект PlaceOrder (забегая вперед скажем, что это будет граничный объект, принадлежащий соответствующему граничному классу). Далее открывается форма ввода личных данных покупателя и его кредитной карты (EnterPersonalInformation), на ней покупатель вводит свое имя, адрес, телефон, адрес электронной почты (enter personal information) и кредитные данные. Информация принимается и открывается форма подтверждения заказа (ConfirmOrder), покупатель подтверждает, что согласен с реквизитами заказа (confirm order), детали заказа сохраняются для дальнейшего использования (save the details). Фокус управления передается некоторому управляющему объекту (PlaceOrderManager), который обращается к внешней кредитной системе (Credit System) для проведения платежа. Если платеж прошел успешно (а именно такой сценарий мы сейчас и рассматриваем), то PlaceOrderManager посылает сообщение (create order) создать объект Заказ (Order), затем вызывает форму подтверждения заказа (OrderConfirmation). Объект Заказ (Order) обращается к объектам Товар (Item) для того, чтобы получить информацию о товарах и создает заказ. Процесс завершается.

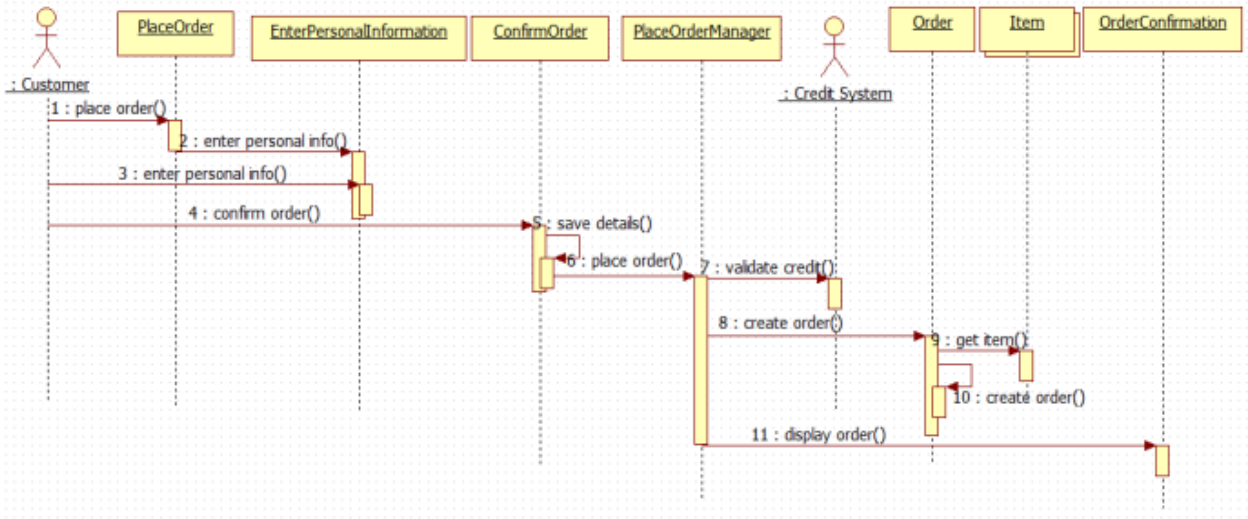

Рисунок 16 – Диаграмма последовательности сценария Оформление заказа

Замечание. Обратите внимание, что символ объекта Товар (Item) на диаграмме последовательности отличается от символов других объектов. Дело в том, что мы задали множественный экземпляр класса. Действительно, заказ может состоять из нескольких товаров, значит объекту Заказ (Order) требуется получить информацию о нескольких объектах Товар (Item). Вместо того, чтобы представлять каждый товар отдельно мы используем нотацию UML для

множественного экземпляра класса, представляя одним значком несколько объектов.

Чтобы сделать объект множественным в StarUML выделите объект, щелкнув по нему мышью один раз, в открывшемся редакторе свойств поставьте флажок в разделе IsMultiInstance (рисунок 17).

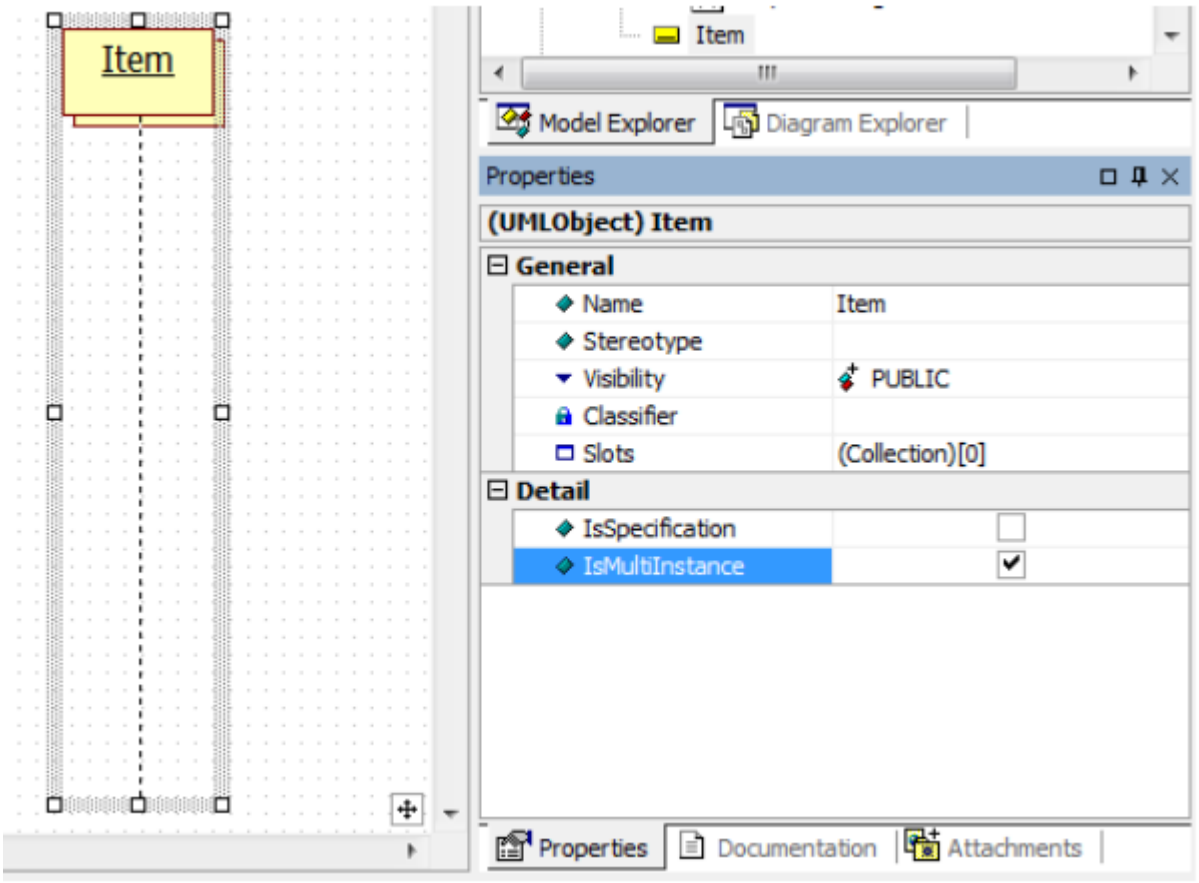

Рисунок 17 – Создание множественного объекта

Ветвление потока управления

Обычно для основного потока событий большинства прецедентов строится одна диаграмма последовательности, для альтернативных потоков - дополнительные диаграммы, описывающие все остальные сценарии. Так поступают потому, что на диаграмме последовательности действий сложно отобразить логику ЕСЛИ-ТО-ИНАЧЕ. Однако если это необходимо и не загромождает диаграмму, то это можно сделать с помощью условий. Приведем пример.

Пример. В процессе оформления покупателем заказа возможны несколько альтернатив. Например, на втором шаге оформления заказа покупатель может подтвердить свой заказ, а может и не согласиться с его реквизитами (см. пример выше). На диаграмме последовательности — это можно изобразить так, как это показано на рисунке 18.

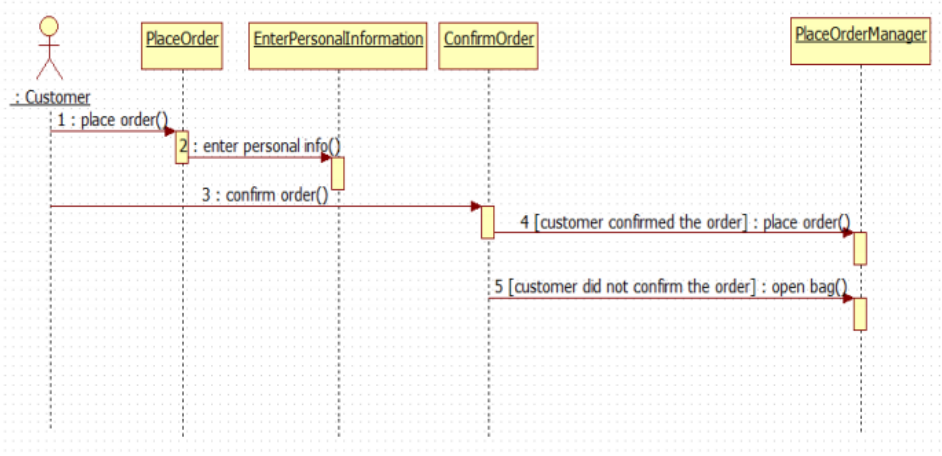

Рисунок 18 – Ветвление потока управления

Если покупатель подтверждает свой заказ на втором шаге (customer confirmed the order), то процесс переходит оплате заказа. Если покупатель не подтверждает заказ (customer did not confirm the order), то открывается корзина покупателя. Условие, как это принято в нотации UML, записывается в квадратных скобках []. Обратите внимание, что мы упростили предыдущую диаграмму описания оформления заказа, иначе добавление ветвей процесса сделало бы ее громоздкой и трудно понимаемой. На практике лучше изображать диаграмму последовательности отдельно для каждого сценария потока событий.

Добавление диаграммы кооперации в модель

Для того чтобы добавить диаграмму кооперации в представление Logical View, щелкните правой кнопкой мыши по папке, содержащей диаграмму последовательности (если вы ее не переименовывали, то она носит имя CollaborationInstanceSet1), в контекстном меню выберите пункт Add Diagram, в списке выберите диаграмму кооперации Collaboration diagram (рисунок 19).

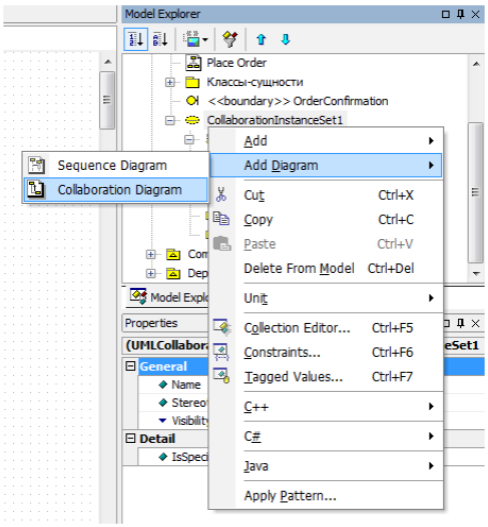

Рисунок 19 – Добавление кооперативной диаграммы

Пример. Для сценария Оформление заказа, для которого мы уже составили диаграмму последовательности. На диаграмму кооперации поместим все те же объекты, перетащив их с навигатора модели (рисунок 20).

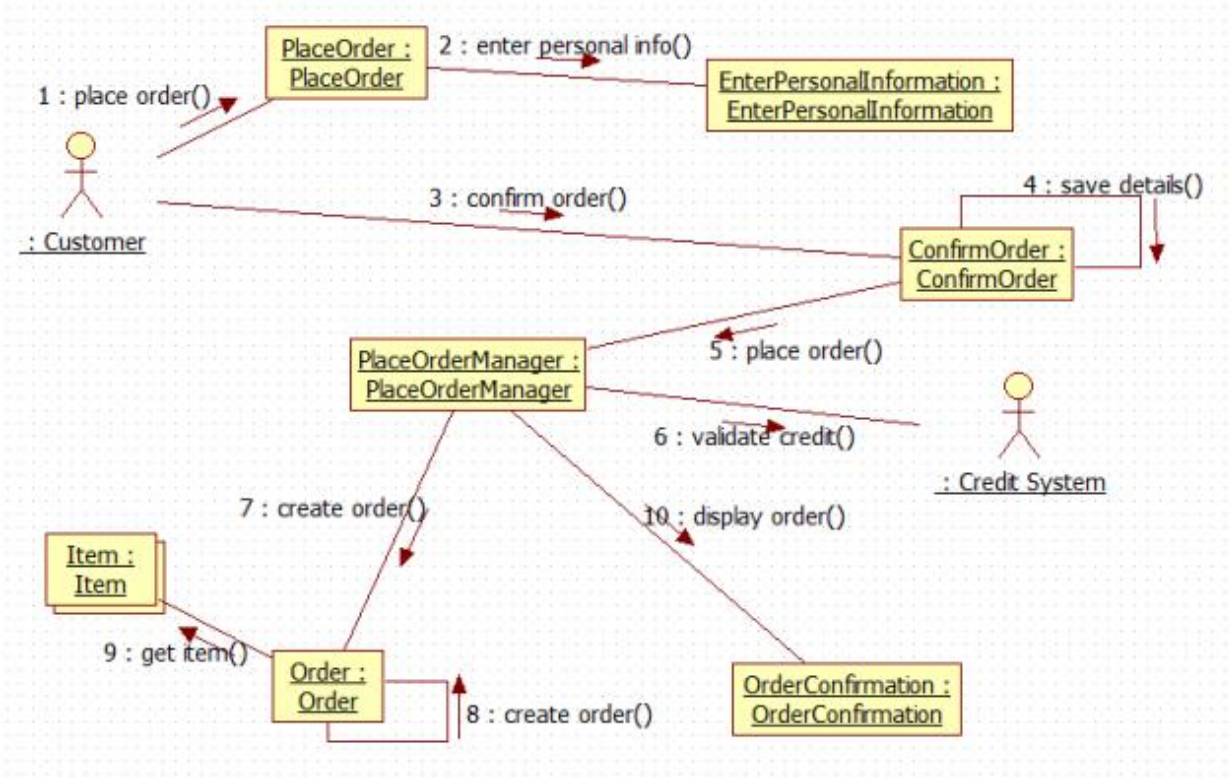

Рисунок 20 – Кооперативная диаграмма сценария Оформление заказа

Задания для самостоятельного выполнения:

Задание 1.

Построить диаграммы последовательности и диаграммы кооперации в StarUML.

Предметную область разрабатываемого программного средства возьмите в соответствии с 1 лабораторной работой

Шкала оценивания результатов деятельности

Задания	Количество баллов
1	1-9

Вопросы для самоконтроля:

1. В чем заключается основная задача диаграммы последовательности?
2. В чем заключается основная задача диаграммы кооперации?
3. Какие действия можно выполнять с помощью данного вида диаграмм?

Примечание: Отчёт по лабораторной работе принимается в электронном виде.