Федеральное государственное бюджетное образовательное учреждение
высшего профессионального образования
«Пермская государственная сельскохозяйственная академия
имени академика Д.Н. Прянишникова»

ПРОЕКТИРОВАНИЕ ИНФОРМАЦИОННЫХ СИСТЕМ

направление 230700 «Прикладная информатика»

ЛАБОРАТОРНОЕ ЗАНЯТИЕ № 7

Тема: **Модель проектирования: диаграммы взаимодействия**

Учебные вопросы:

1. Модель проектирования: диаграммы взаимодействия.

Вопрос 1. Диаграммы взаимодействия

Система обозначений для диаграмм взаимодействия

В этом вопросе лишь вводится система обозначений. Для грамотного объектного проектирования необходимо понимать его основные принципы. После ознакомления с системой обозначений для диаграмм взаимодействия в следующих вопросах вы ознакомитесь с этими принципами и их применением для построения диаграмм взаимодействия.

Диаграммы последовательностей и кооперации

Термин "диаграмма взаимодействия" используется в качестве общего названия для двух следующих конкретных типов диаграмм, которые могут использоваться для иллюстрации обмена сообщениями.

- Диаграммы кооперации (collaboration diagram)
- Диаграммы последовательностей (sequence diagram)

Чтобы подчеркнуть свободу разработчиков при выборе артефактов проектирования, будут использованы оба типа диаграмм.

Диаграммы кооперации (collaboration diagram) иллюстрируют взаимодействие объектов в формате графа или сети, как показано на рис. 1.1. При этом объекты могут размещаться в любом месте диаграммы.

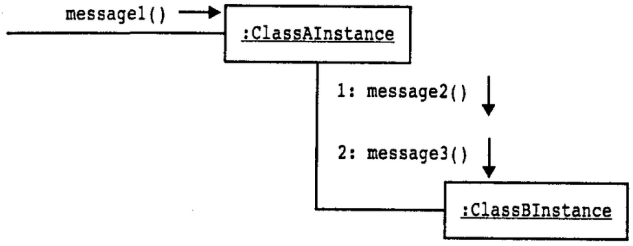

Рисунок 1.1 – Диаграмма кооперации

Диаграммы последовательностей (sequence diagram) иллюстрируют взаимодействие в форме, показанной на рис. 1.2. Здесь объекты располагаются слева направо.

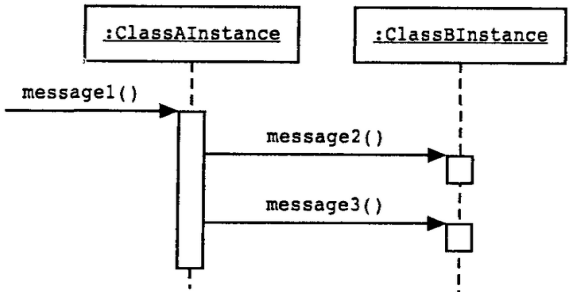

Рисунок 1.2 – Диаграмма последовательностей

Оба типа диаграмм имеют свои преимущества и недостатки.

При использовании CASE-средств разработки многие предпочитают строить диаграммы последовательностей для удобства обратного проектирования – преобразования исходного кода в диаграмму взаимодействия.

Тип диаграммы	Преимущества	Недостатки
Последовательностей	Ясно отображает последовательность и временной порядок сообщений. Простые обозначения	Расширяется вправо при добавлении новых объектов; занимает много места по горизонтали

Кооперации	Экономия пространства – возможность добавления объектов в двух направлениях. Лучше иллюстрирует сложные зависимости, итерационность и параллельные	Сложнее отследить последовательность сообщений. Более сложная система обозначений

Пример диаграммы кооперации: makePayment

Показанную на рис. 1.3 диаграмму кооперации нужно интерпретировать следующим образом.

1. Сообщение makePayment передается экземпляру объекта Register. Отправитель сообщения не определен.
2. Объект Register передает сообщение makePayment экземпляру объекта Sale.
3. Объект Sale создает экземпляр объекта Payment.

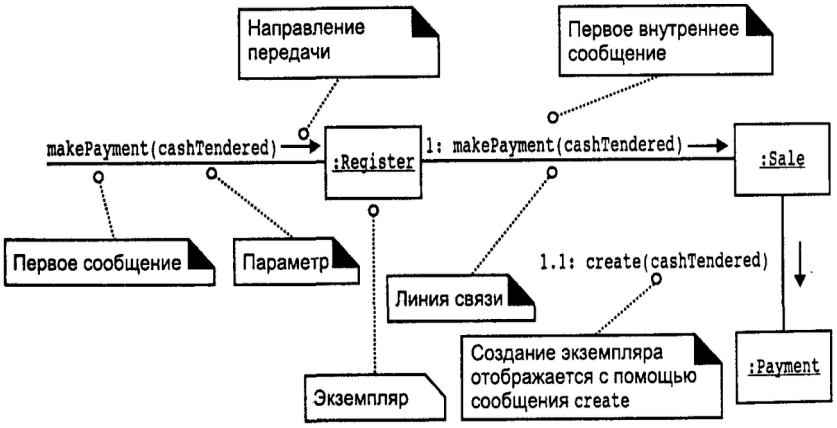

Рисунок 1.3 – Диаграмма кооперации

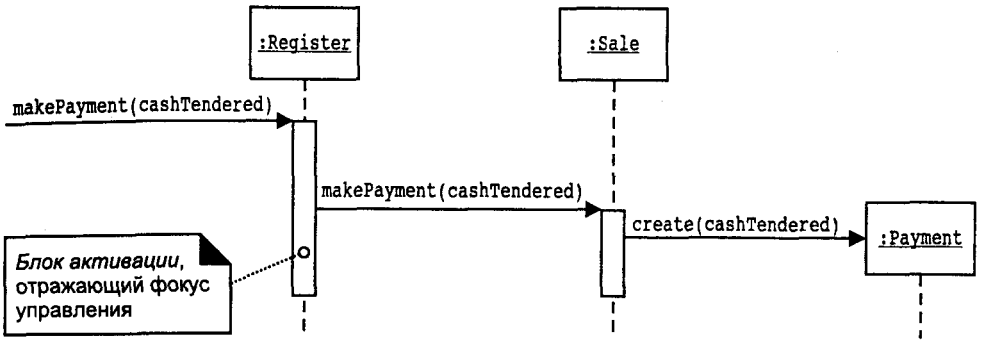

Рисунок 1.4 – Диаграмма последовательностей

Для успешного конструирования диаграмм взаимодействий принципы разработки предварительно *могут* быть систематизированы и проанализированы. Такой подход к пониманию и использованию этих принципов основывается на *шаблонах* (patterns), представляющих собой структурированные рекомендации \ принципы.

Основные обозначения для диаграмм взаимодействия: Отображение классов и экземпляров объектов

В языке UML для иллюстрации *экземпляров* объектов (а не классификаторов) используется простой и непротиворечивый подход (рис. 1.5):

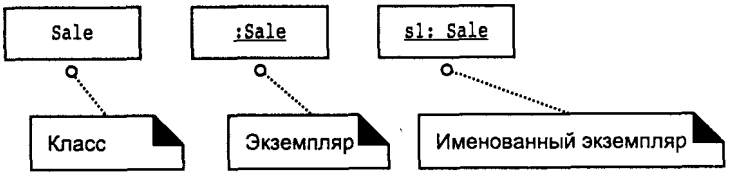

Рисунок 1.5 – Классы и экземпляры объектов

Для экземпляра любого элемента языка UML (класса, исполнителя и т.д. используется то же самое графическое обозначение, что и для типа, однако при этом соответствующая определяющая строка *подчеркивается.*

Таким образом, для отображения экземпляра класса на диаграмме взаимодействий используется обычное графическое условное обозначение класса, одна ко при этом его имя подчеркивается.

Для уникальной идентификации экземпляра класса может использоваться его имя. Если имя экземпляра не указано, на диаграмме кооперации перед именем класса ставится двоеточие.

Синтаксис для отображения сообщений

В языке UML существует стандартный синтаксис для обозначения передач сообщений.

получатель := ***сообщение (параметр : типПараметра) : типПолучателя***

Информация о типах может исключаться в силу своей очевидности или не значительности. Например:

```
spec := getProductSpect(id)
spec := getProductSpect(id:ItemID)
spec := getProductSpect(id:ItemID) : ProductSpecification
```

Основные обозначения диаграммы кооперации: Отображение связей

Связь (link) является соединением между двумя экземплярами классов, определяющим некоторую форму перемещения и видимости между ними (рис. 1.6). Более строго, можно сказать, что связь является экземпляром ассоциации. Например, имеется связь, или маршрут перемещения, от объекта Register к объекту Sale, в соответствии с которым могут передаваться сообщения, такие как makePayment.

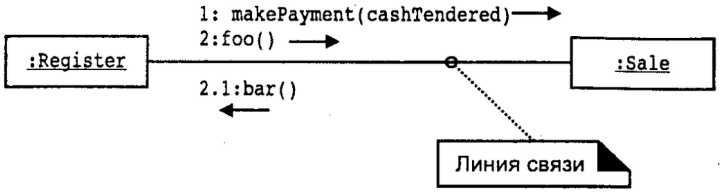

Рисунок 1.6 – Линии связей

Обратите внимание, что по одной и той же линии связи могут передаваться несколько сообщений в обоих направлениях.

Отображение сообщений

Передаваемые между объектами сообщения представляются в виде имен этих сообщений над линиями связей, помеченных стрелками. Над одной линией связи может быть указано любое количество сообщений (рис. 1.7). Для отображения порядка следования сообщений в текущем потоке управления рядом с сообщением приводится порядковый номер.

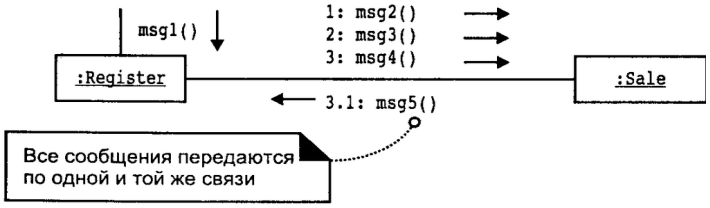

Рисунок 1.7 – Сообщения

Сообщения, передаваемые самому объекту

Сообщение может передаваться объектом самому себе (рис. 1.8). Такой случай отображается с помощью связи объекта с самим собой, когда сообщения передаются в направлении этой связи.

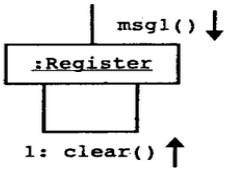

Рисунок 1.8 – Сообщения, передаваемые самому объекту

Создание экземпляров объектов

Для создания экземпляра объекта можно использовать любые сообщения. Однако в языке UML принято использовать для этой цели сообщение create (создать). При использовании другого имени (возможно, менее очевидного) сообщение следует снабдить специальным свойством, получившим название стереотипа UML, в частности «create». Это сообщение передается создаваемому экземпляру объекта (рис. 1.9).

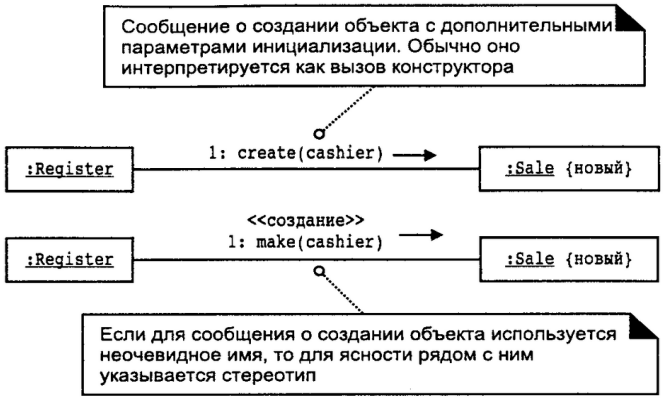

Рисунок 1.9 – Создание экземпляра объекта

Сообщение create может иметь параметры, задающие начальные значения свойств создаваемого экземпляра. Такая форма записи может означать, например, вызов конструктора с параметрами на языке Java.

Более того, чтобы подчеркнуть факт создания экземпляра, новому экземпляру объекта можно дополнительно присвоить свойство {новый}.

Представление порядка передачи сообщений

Порядок передачи сообщений иллюстрируется с помощью *порядковых номеров* (sequence number) (рис. 1.10). При этом используется следующая схема нумерации:

1. Первое сообщение не нумеруется. Таким образом, сообщение msgl() является ненумерованным.

2. Порядок и вложенность последующих сообщений отображается в соответствии с принятой схемой нумерации. При ее использовании к вложенным сообщениям добавляется номер. Вложенность означает, что к номеру исходящего сообщения добавляется номер входящего сообщения.

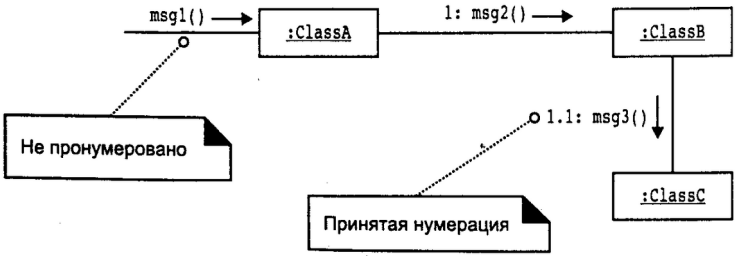

Рисунок 1.10 – Нумерация последовательности сообщений

На рис. 1.11 показан более сложный случай.

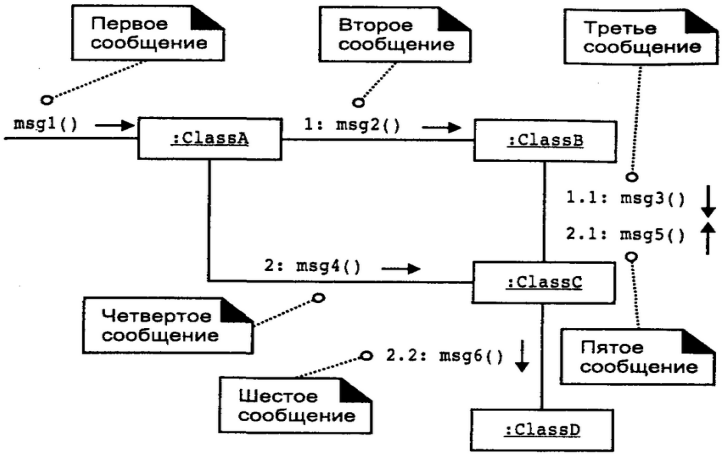

Рисунок 1.11 – Сложная нумерация последовательности сообщений

Представление условных сообщений

Условное сообщение (рис. 1.12) изображается с помощью его номера, за которым в квадратных скобках указывается условное выражение, аналогичное условию цикла. Сообщение передается только в том случае, когда оператором возвращается значение true.

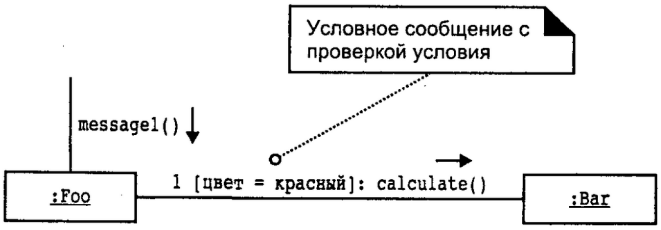

Рисунок 1.12 – Условное сообщение

Представление взаимоисключающих условных маршрутов

Приведенный на рис. 1.13 пример иллюстрирует порядковые номера сообщений при использовании взаимоисключающих условных маршрутов.

В этом случае требуется модифицировать схему нумерации, воспользовавшись символом условного маршрута. В соответствии с принятым соглашением, первым таким символом является буква a. Фрагмент диаграммы, приведенной на рис. 1.13, означает, что после передачи сообщения msgl() будет передано либо сообщение 1a, либо сообщение 1b. Наличие номера 1 означает, что оба сообщения относятся к первому внутреннему сообщению.

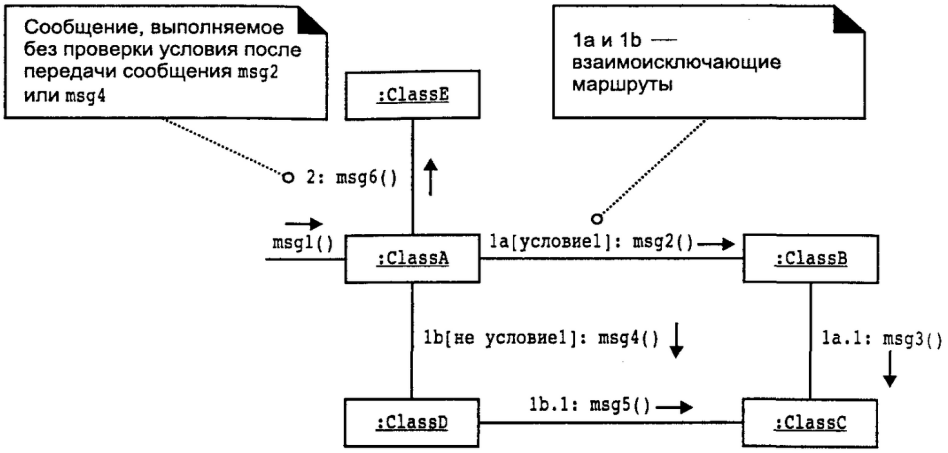

Рисунок 1.13 – Взаимоисключающие сообщения

Обратите внимание, что последующие вложенные сообщения по-прежнему нумеруются согласно соответствующим внешним сообщениям. Таким образом, 1b.1 является вложенным сообщением сообщения 1b.

Представление итерационного процесса или циклов

Обозначения для итерационного процесса показаны на рис. 1.14. Итерационный процесс можно отобразить, указав за порядковым номером сообщения символ *.

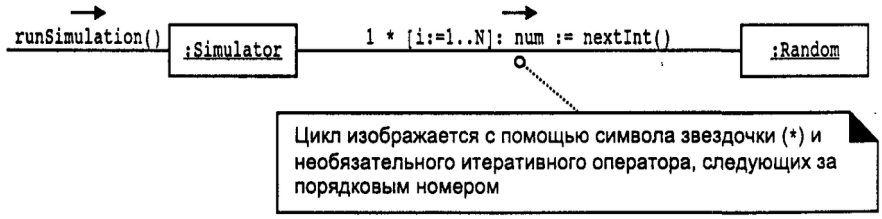

Рисунок 1.14 – Итерационный процесс

Итерационный процесс для коллекций объектов

Зачастую сообщение передается каждому элементу коллекции (например, списку или карте). Для этого можно использовать некий вид объекта-итератора, например, в Java реализацию java.util.Iterator или итератор из стандартной библиотеки C++. В UML для обозначения набора экземпляров или коллекции применяется термин *сложный объект* или *мультиобъект* (multiobject). На диаграмме кооперации сложный объект отображается с использованием условного обозначения, показанного на рис. 1.15.

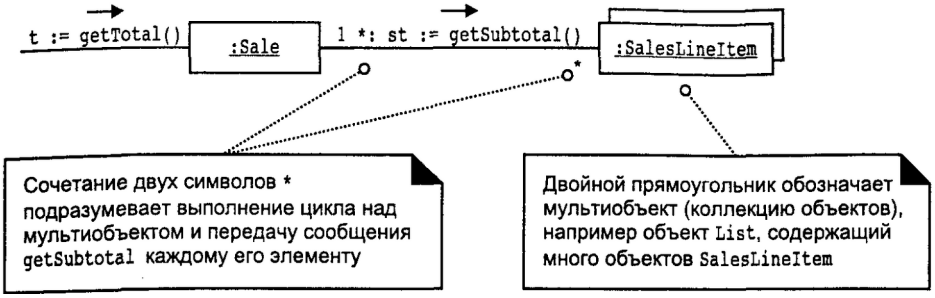

Рисунок 1.15 – Итерационный процесс для сложного объекта

Сообщения, передаваемые классу

Сообщения могут передаваться самому классу, а не его экземплярам. Это Может понадобиться, например, для вызова статических методов класса. При этом сообщение изображается как обычно, однако в условном обозначении класса его имя не подчеркнуто. Тем самым указывается, что это сообщение передается самому классу, а не его экземпляру (рис. 1.16).

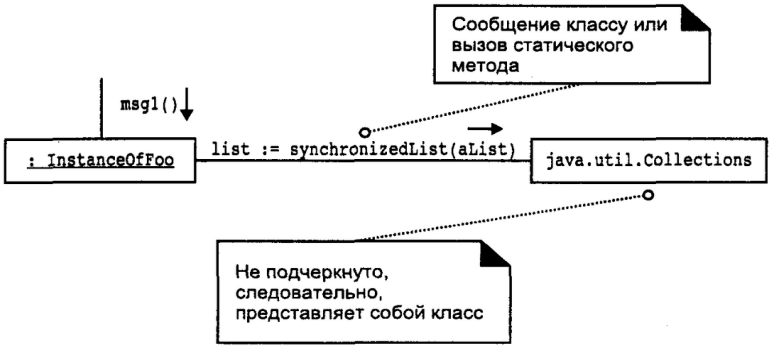

Рисунок 1.16 – Сообщения, передаваемые объекту класса (вызов статических методов)

Очень важно, чтобы там, где это нужно, имена экземпляров были подчеркнуты. В противном случае передаваемые сообщения могут быть неверно интерпретированы.

Основные обозначения диаграммы последовательностей

Связи

В отличие от диаграмм кооперации, на диаграмме последовательностей связи не отображаются.

Сообщения

Сообщения между объектами изображаются в виде соединяющих объекты линий со стрелками на конце, над которыми указывается имя сообщения. Порядок передачи сообщений определяется их расположением сверху вниз.

Фокус управления и блоки активации

Как видно из рис. 1.17, на диаграммах последовательностей можно отображать фокус управления с использованием **блока активации** (activation box). Блоки активации указывать необязательно, но специалисты по UML их обычно используют.

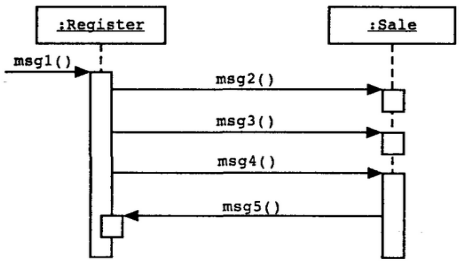

Рисунок 1.17 – Сообщения и фокус управления

Возвращаемые значения

На диаграммах последовательностей при желании можно отражать возврат значения при передаче сообщения. Для этого используются штриховые линии со стрелками на конце, исходящие из блока активации (рис. 1.18). Многие опытные специалисты их не применяют. Однако иногда с помощью этих линий изображают возвращаемые значения (если таковые имеются).

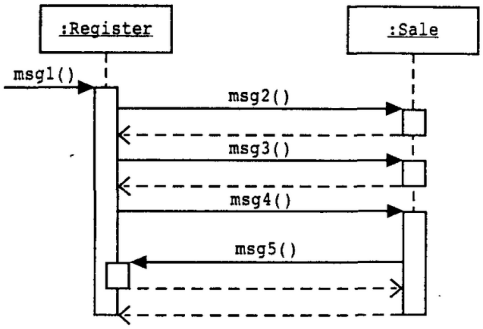

Рисунок 1.18 – Отображение возврата значения

Сообщения, передаваемые самому объекту

Передача сообщения объектом самому себе отображается с использованием вложенных активационных блоков (рис. 1.19).

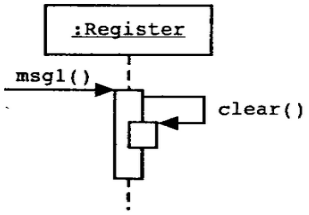

Рисунок 1.19 – Сообщения, передаваемые самому объекту

Создание экземпляров объектов

Обозначения, иллюстрирующие создание экземпляра, показаны на рис. 1.20.

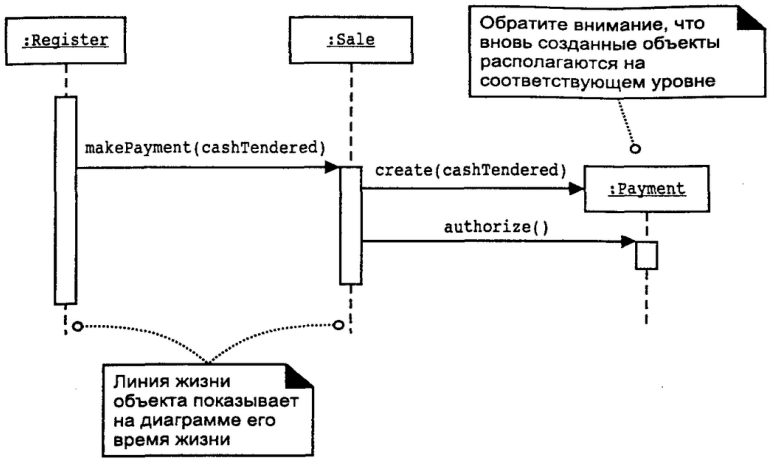

Рисунок 1.20 – Создание экземпляра объекта и линии жизни

Линии жизни объектов и уничтожение объектов

На рис. 1.20 показаны также **линии жизни объектов** (object lifelines) – вертикальные штриховые линии, расположенные под соответствующими объектами. Иногда желательно отобразить на диаграмме факт уничтожения объекта (например, в языке C++ отсутствует механизм сборки мусора). Этот факт можно отобразить в терминах UML с помощью специального символа на линии жизни объекта (рис. 1.21).

Представление условных сообщений

Условное сообщение показано на рис. 1.22.

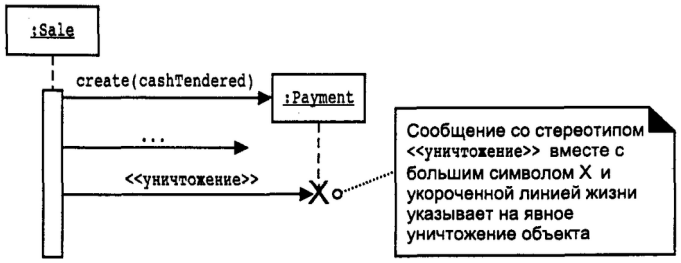

Рисунок 1.21 – Уничтожение объекта

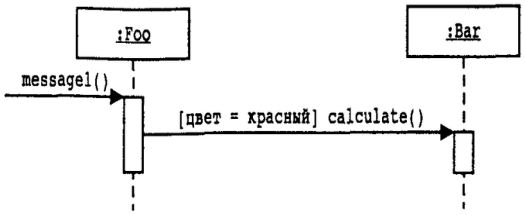

Рисунок 1.22 – Условное сообщение

Представление взаимоисключающих условных маршрутов

Для этого случая используются линии сообщений, исходящие из одной точки под разными углами, как показано на рис. 1.23.

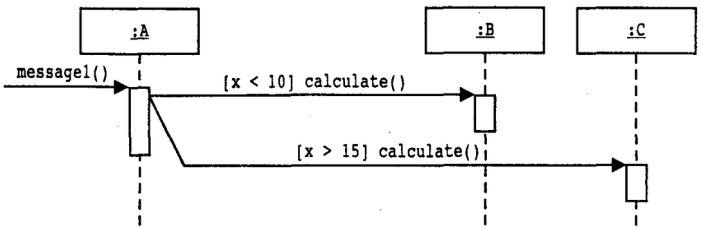

Рисунок 1.23 – Взаимоисключающие сообщения

Представление итерационного процесса для одного сообщения

Обозначения итерационного процесса для одного сообщения показаны на рис. 1.24.

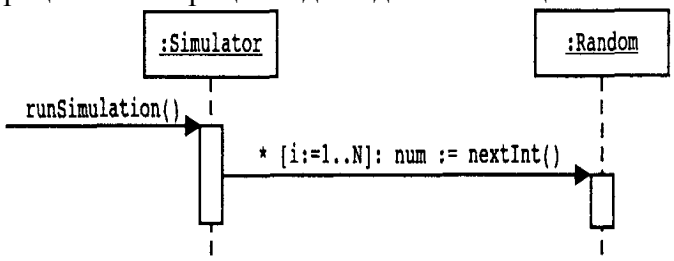

Рисунок 1.24 – Итерационный процесс для одного сообщения

Итерационный процесс для последовательности сообщений

Обозначения итерационного процесса для последовательности сообщений показаны на рис. 1.25.

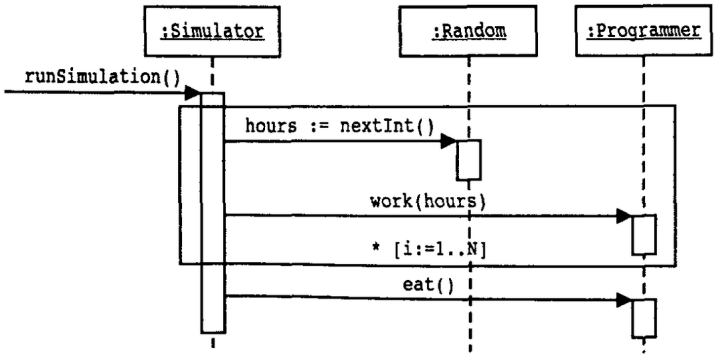

Рисунок 1.25 – Итерационный процесс для последовательности сообщений

Итерационный процесс для коллекции (сложного объекта)

Обозначения итерационного процесса для коллекции объектов показаны на рис. 1.26.

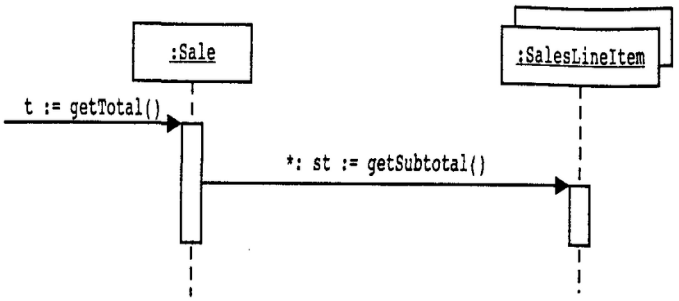

Рисунок 1.26 – Итерационный процесс для сложного объекта

На диаграмме кооперации для иллюстрации передачи сообщения каждому элементу, а не самой коллекции, в конце каждой роли указывается символ кратности "*". Для диаграммы последовательностей аналогичное обозначение в языке UML отсутствует.

Сообщения, передаваемые классу

Как и на диаграмме кооперации, при вызове статических методов или методов класса имя классификатора не подчеркивается. Тем самым указывается, что это сообщение передается самому классу, а не его экземпляру (рис. 1.27).

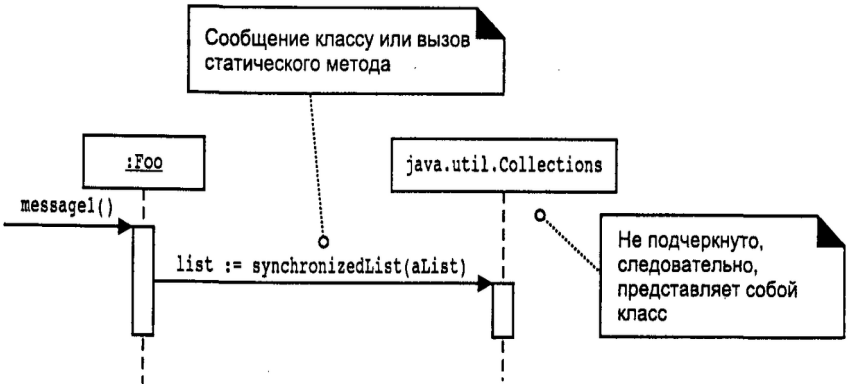

Рисунок 1.27 – Сообщения, передаваемые объекту класса (вызов статических методов)

Задание на самостоятельную работу (для выбранной темы индивидуального проекта):

1. построить диаграммы кооперации и диаграммы последовательностей для выделенного прецедента.