Тема: Диаграммы последовательности
Цель: Изучить принципы построения диаграмм последовательности и построить диаграмму последовательности для собственной модели.

Теоретические сведения
Для моделирования взаимодействия объектов в языке UML используются соответствующие диаграммы *взаимодействия.* Взаимодействия объектов можно рассматривать во времени, и тогда для представления временных особенностей передачи и приема сообщений между объектами используется диаграмма последовательности. Временной аспект поведения может иметь существенное значение при моделировании синхронных процессов, описывающих взаимодействия объектов. Именно для этой цели в языке UML используются диаграммы последовательности.

Объекты

На диаграмме последовательности изображаются исключительно те объекты, которые непосредственно участвуют во взаимодействии и не показываются возможные статические ассоциации с другими объектами. Для диаграммы последовательности ключевым моментом является именно динамика взаимодействия объектов во времени. При этом диаграмма последовательности имеет как бы два измерения. Одно — слева направо в виде вертикальных линий, каждая из которых изображает линию жизни отдельного объекта, участвующего во взаимодействии. Графически каждый объект изображается прямоугольником и располагается в верхней части своей линии жизни. Внутри прямоугольника записываются имя объекта и имя класса, разделенные двоеточием. При этом вся запись подчеркивается, что является признаком объекта, который, как известно, представляет собой экземпляр класса.

> **Примечание**
> Не исключается ситуация, когда имя объекта может отсутствовать на диаграмме последовательности. В этом случае указывается только имя класса, а сам объект считается анонимным

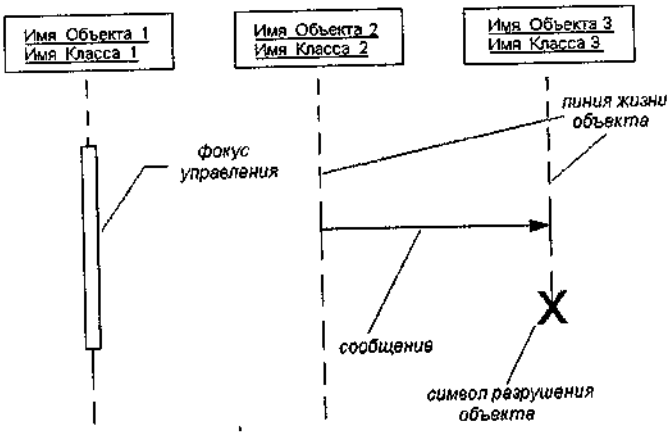

Рис. 1. Различные графические примитивы диаграммы последовательности

Крайним слева на диаграмме изображается объект, который является инициатором взаимодействия (объект 1 на рис. 1). Правее изображается другой объект, который непосредственно взаимодействует с первым. Таким образом, все объекты на диаграмме последовательности образуют некоторый порядок, определяемый степенью активности этих объектов при взаимодействии друг с другом.

Второе измерение диаграммы последовательности — вертикальная временная OCL, направленная сверху вниз. Начальному моменту времени соответствует самая верхняя часть

диаграммы. При этом взаимодействия объектов реализуются посредством сообщений, которые посылаются одними объектами другим. Сообщения изображаются в виде горизонтальных стрелок с именем сообщения и также образуют порядок по времени своего возникновения. Другими словами, сообщения, расположенные на диаграмме последовательности выше, инициируются раньше тех, которые расположены ниже. При этом масштаб на оси времени не указывается, поскольку диаграмма последовательности моделирует лишь временную упорядоченность взаимодействий типа "раньше-позже".

Линия жизни объекта

Линия жизни объекта (object lifeline) изображается пунктирной вертикальной линией, ассоциированной с единственным объектом на диаграмме последовательности. Линия жизни служит для обозначения периода времени, в течение которого объект существует в системе и, следовательно, может потенциально участвовать во всех ее взаимодействиях. Если объект существует в системе постоянно, то и его линия жизни должна продолжаться по всей плоскости диаграммы последовательности от самой верхней ее части до самой нижней (объекты 1 и 2 на рис. 1).

Отдельные объекты, выполнив свою роль в системе, могут быть уничтожены (разрушены), чтобы освободить занимаемые ими ресурсы. Для таких объектов линия жизни обрывается в момент его уничтожения. Для обозначения момента уничтожения объекта в языке UML используется специальный символ в форме латинской буквы "X" (объект 3 на рис. 1). Ниже этого символа пунктирная линия не изображается, поскольку соответствующего объекта в системе уже нет, и этот объект должен быть исключен из всех последующих взаимодействий.

Вовсе не обязательно создавать все объекты в начальный момент времени. Отдельные объекты в системе могут создаваться по мере необходимости, существенно экономя ресурсы системы и повышая ее производительность. В этом случае прямоугольник такого объекта изображается не в верхней части диаграммы последовательности, а в той ее части, которая соответствует моменту создания объекта (объект 6 на рис. 2). При этом прямоугольник объекта вертикально располагается в том месте диаграммы, которое по оси времени совпадает с моментом его возникновения в системе. Очевидно, объект обязательно создается со своей линией жизни и, возможно, с фокусом управления.

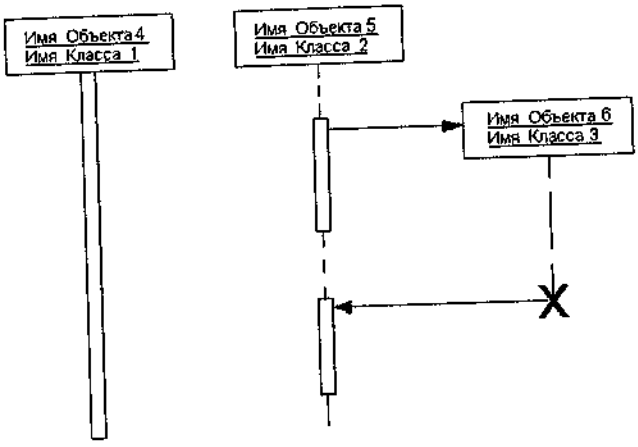

Рис.2. Графическое изображение различных вариантов линий жизни и фокусов управления объектов

Фокус управления

В процессе функционирования объектно-ориентированных систем одни объекты могут находиться в активном состоянии, непосредственно выполняя определенные действия или в состоянии пассивного ожидания сообщений от других объектов. Чтобы явно выделить подобную активность объектов, в языке UML применяется специальное понятие, получившее название *фокуса управления* (focus of control). Фокус управления изображается в форме вытянутого узкого прямоугольника (см. объект 1 на рис. 1), верхняя сторона которого обозначает начало получения фокуса управления объекта (начало активности), а ее нижняя сторона — окончание фокуса управления (окончание активности). Этот прямоугольник располагается ниже обозначения соответствующего объекта и может заменять его линию жизни (объект 4 на рис. 2), если на всем ее протяжении он является активным. С другой стороны, периоды активности объекта могут чередоваться с периодами его пассивности или ожидания. В этом случае у такого объекта имеются несколько фокусов управления (объект 5 на рис. 2) Важно понимать, что получить фокус управления может только существующий объект, у которого в этот момент имеется линия жизни. Если же некоторый объект был уничтожен, то вновь возникнуть в системе он уже не может. Вместо него лишь может быть создан другой экземпляр этого же класса, который, строго говоря, будет являться другим объектом.

В отдельных случаях инициатором взаимодействия в системе может быть актор или внешний пользователь. В этом случае актор изображается на диаграмме последовательности самым первым объектом слева со своим фокусом управления (рис. 3). Чаще всего актор и его фокус управления будут существовать в системе постоянно, отмечая характерную для пользователя активность в инициировании взаимодействий с системой. При этом сам актор может иметь собственное имя либо оставаться анонимным.

Иногда некоторый объект может инициировать рекурсивное взаимодействие с самим собой. Речь идет о том, что наличие во многих языках программирования специальных средств построения рекурсивных процедур требует визуализации соответствующих понятий *в* форме графических примитивов. На диаграмме последовательности *рекурсия* обозначается небольшим прямоугольником, присоединенным к правой стороне фокуса управления того объекта, для которого изображается это рекурсивное взаимодействие (объект 7 на рис. 3).

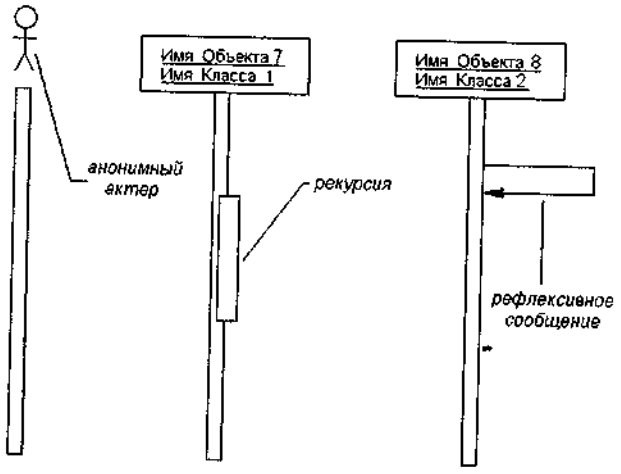

Рис. 3. Графическое изображение актера, рекурсии и рефлексивного сообщения на диаграмме последовательности

Сообщения

Как было отмечено выше, цель взаимодействия в контексте языка UML заключается в том, чтобы специфицировать коммуникацию между множеством взаимодействующих объектов. Каждое взаимодействие описывается совокупностью сообщений, которыми участвующие в нем

объекты обмениваются между собой. В этом смысле *сообщение* (message) представляет собой законченный фрагмент информации, который отправляется одним объектом другому. При этом прием сообщения инициирует выполнение определенных действий, направленных на решение отдельной задачи тем объектом, которому это сообщение отправлено.

Таким образом, сообщения не только передают некоторую информацию, но и требуют или предполагают от принимающего объекта выполнения ожидаемых действий. Сообщения могут инициировать выполнение операций объектом соответствующего класса, а параметры этих операций передаются вместе с сообщением. На диаграмме последовательности все сообщения упорядочены по времени своего возникновения в моделируемой системе.

В таком контексте каждое сообщение имеет направление от объекта, который инициирует и отправляет сообщение, к объекту, который его получает. Иногда отправителя сообщения называют клиентом, а получателя — сервером. При этом сообщение от клиента имеет форму запроса некоторого сервиса, а реакция сервера на запрос после получения сообщения может быть связана с выполнением определенных действий или передачи клиенту необходимой информации тоже в форме сообщения.

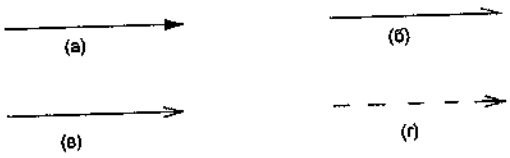

Рис. 4. Графическое изображение различных видов сообщений между объектами на диаграмме последовательности.

В языке UML могут встречаться несколько разновидностей сообщений, каждое из которых имеет свое графическое изображение (рис. 4).

Первая разновидность сообщения (рис. 4, а) является наиболее распространенной и используется для вызова процедур, выполнения операций или обозначения отдельных вложенных потоков управления Начало этой стрелки всегда соприкасается с фокусом управления или линией жизни того объекта-клиента, который инициирует это сообщение. Конец стрелки соприкасается с линией жизни того объекта, который принимает это сообщение и выполняет в ответ определенные действия. При этом принимающий объект зачастую получает и фокус управления, становясь активным.

Вторая разновидность сообщения (рис. 4, б) используется для обозначения простого (не вложенного) потока управления. Каждая такая стрелка указывает на прогресс одного шага потока. При этом соответствующие сообщения обычно являются асинхронными, т. е могут возникать в произвольные моменты времени. Передача такого сообщения обычно сопровождается получением фокуса управления объектом, его принявшим.

Третья разновидность (рис 4, в) явно обозначает асинхронное сообщение между двумя объектами в некоторой процедурной последовательности. Примером такого сообщения может служить прерывание операции при возникновении исключительной ситуации. В этом случае информация о такой ситуации передается вызывающему объекту для продолжения процесса дальнейшего взаимодействия.

Наконец, последняя разновидность сообщения (рис. 4, г) используется для возврата из вызова процедуры. Примером может служить простое сообщение о завершении некоторых вычислений без предоставления результата расчетов объекту-клиенту. В процедурных потоках управления эта стрелка может быть опущена, поскольку ее наличие неявно предполагается в конце активизации объекта. В то же время считается, что каждый вызов процедуры имеет свою пару — возврат вызова. Для непроцедурных потоков управления, включая параллельные и асинхронные сообщения, стрелка возврата должна указываться явным образом.

Обычно сообщения изображаются горизонтальными стрелками, соединяющими линии жизни или фокусы управления двух объектов на диаграмме последовательности. При этом неявно предполагается, что время передачи сообщения достаточно мало по сравнению с процессами

выполнения действий объектами. Считается также, что за время передачи сообщения с соответствующими объектами не может произойти никаких событий Другими словами, состояния объектов остаются без изменения. Если же это предположение не может быть признано справедливым, то стрелка сообщения изображается под некоторым наклоном, так чтобы конец стрелки располагался ниже ее начала.

В отдельных случаях объект может посылать сообщения самому себе, инициируя так называемые *рефлексивные* сообщения (объект 8 на рис. 3) Такие сообщения изображаются прямоугольником со стрелкой, начало и конец которой совпадают.

Таким образом, в языке UML каждое сообщение ассоциируется с некоторым действием, которое должно быть выполнено принявшим его объектом. При этом действие может иметь некоторые аргументы или параметры, в зависимости от конкретных значений которых может быть получен различный результат. Соответствующие параметры будет иметь и вызывающее это действие сообщение. Более того, значения параметров отдельных сообщений могут содержать условные выражения, образуя ветвление или альтернативные пути основного потока управления.

Ветвление потока управления

Для изображения ветвления рисуются две или более стрелки, выходящие из одной точки фокуса управления объекта (фокус управления объекта 1 на рис 5) При этом соответствующие условия должны быть явно указаны рядом с каждой из стрелок в форме сторожевого условия. Как нетрудно представить, если условие записано в форме булевского выражения, то ветвление будет содержать только две ветви. В любом случае условия должны взаимно исключать одновременную передачу альтернативных сообщений. С помощью ветвления можно изобразить и более сложную логику взаимодействия объектов между собой (фокус управления объекта 1 на рис 6) Если условий более двух, то для каждого из них необходимо предусмотреть ситуацию единственного выполнения.

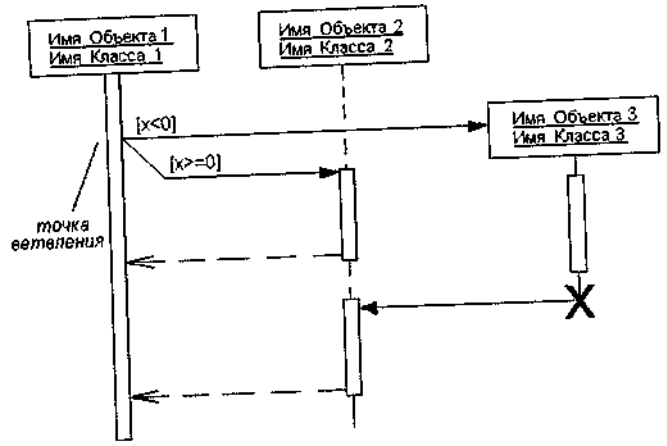

Рис. 8.5. Графическое изображение бинарного ветвления потока управления на диаграмме последовательности

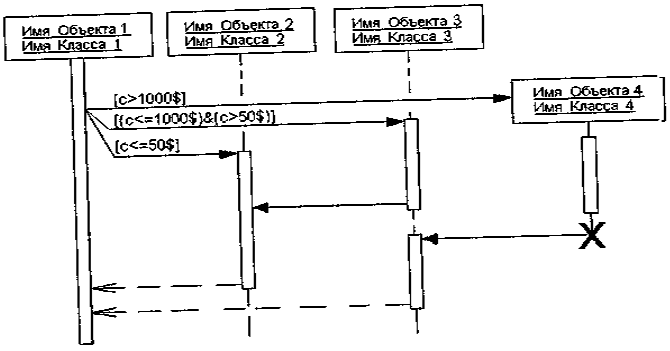

Рис. 8.6. Графическое изображение тернарного ветвления потока управления на диаграмме последовательности

Стереотипы сообщений

В языке UML предусмотрены некоторые стандартные действия, выполняемые в ответ на получение соответствующего сообщения. Они могут быть явно указаны на диаграмме последовательности в форме стереотипа рядом с сообщением, к которому они относятся. В этом случае они записываются в кавычках. Используются следующие обозначения для моделирования действий:

"call" (вызвать) — сообщение, требующее вызова операции или процедуры принимающего объекта. Если сообщение с этим стереотипом рефлексивное, то оно инициирует локальный вызов операции у самого пославшего это сообщение объекта;

"return" (возвратить) — сообщение, возвращающее значение выполненной операции или процедуры вызвавшему ее объекту. Значение результата может инициировать ветвление потока управления; "create" (создать) — сообщение, требующее создания другого объекта для выполнения определенных действий. Созданный объект может получить фокус управления, а может и не получить его;

"destroy" (уничтожить) — сообщение с явным требованием уничтожить соответствующий объект. Посылается в том случае, когда необходимо прекратить нежелательные действия со стороны существующего в системе объекта, либо когда объект больше не нужен и должен освободить задействованные им системные ресурсы;

"send" (послать) — обозначает посылку другому объекту некоторого сигнала, который асинхронно инициируется одним объектом и принимается (перехватывается) другим. Отличие сигнала от сообщения заключается в том, что сигнал должен быть явно описан в том классе, объект которого инициирует его передачу.

Кроме стереотипов, сообщения могут иметь собственное обозначение операции, вызов которой они инициируют у принимающего объекта. В этом случае рядом со стрелкой записывается имя операции с круглыми скобками, в которых могут указываться параметры или аргументы соответствующей операции. Если параметры отсутствуют, то скобки все равно должны присутствовать после имени операции.

Примечание

Согласно принятой в языке UML системе обозначений такие имена операций записываются на английском с малой буквы и одним словом, возможно, состоящим из нескольких сокращенных слов, написанных без пробела и без кавычек. Если нет никаких дополнительных ограничений со стороны инструментальных средств визуализации канонических диаграмм, то дело вкуса отечественного разработчика, какие обозначения ему использовать в русскоязычной транслитерации.

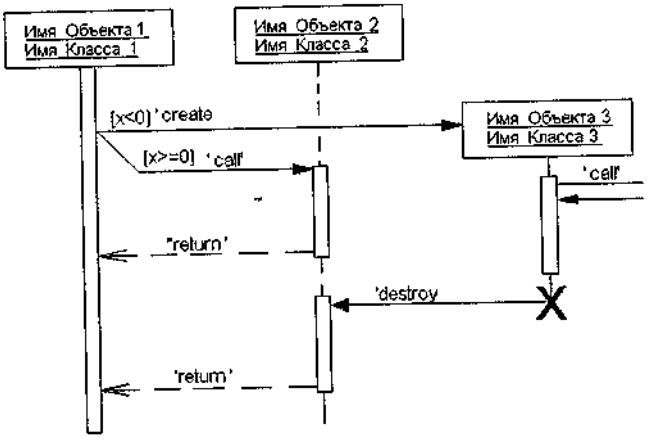

Рис. 8.7. Диаграмма последовательности со стереотипными значениями сообщений

Временные ограничения на диаграммах последовательности

В отдельных случаях выполнение тех или иных действий на диаграмме последовательности может потребовать явной спецификации *временных ограничений,* накладываемых на сам интервал выполнения операций или передачу сообщений. В языке UML для записи временных ограничений используются фигурные скобки. Временные ограничения могут относиться как к выполнению определенных действий объектами, так и к самим сообщениям, явно специфицируя условия их передачи или приема. Важно понимать, что в отличие от условий ветвления, которые должны выполняться альтернативно, временные ограничения имеют обязательный или директивный характер для ассоциированных с ними объектов.

Временные ограничения могут записываться рядом с началом стрелки соответствующего сообщения. Но наиболее часто они записываются слева от этой стрелки на одном уровне с ней. Если временная характеристика относится к конкретному объекту, то имя этого объекта записывается перед именем характеристики и отделяется от нее точкой.

Примерами таких ограничений на диаграмме последовательности могут служить ситуации, когда необходимо явно специфицировать время, в течение которого допускается передача сообщения от клиента к серверу или обработка запроса клиента сервером.

Заключительные рекомендации по построению диаграмм последовательности

Как уже отмечалось, построение диаграммы последовательности целесообразно начинать с выделения из всей совокупности тех и только тех классов, объекты которых участвуют в моделируемом взаимодействии. После этого все объекты наносятся на диаграмму с соблюдением некоторого порядка инициализации сообщений. Здесь необходимо установить, какие объекты будут существовать постоянно, а какие временно — только на период выполнения ими требуемых действий.

Когда объекты визуализированы, можно приступать к спецификации сообщений. При этом следует учитывать те роли, которые играют сообщения в системе. При необходимости уточнения этих ролей надо использовать их разновидности и стереотипы. Для уничтожения объектов, которые создаются на время выполнения своих действий, нужно предусмотреть явное сообщение.

Наиболее простые случаи ветвления процесса взаимодействия можно изобразить на одной диаграмме с использованием соответствующих графических примитивов. Однако следует помнить, что каждый альтернативный поток управления может существенно затруднить понимание построенной модели. Поэтому общим правилом является визуализация каждого потока управления на отдельной диаграмме последовательности. В этой ситуации такие отдельные диаграммы должны рассматриваться совместно как одна модель взаимодействия.

Дальнейшая детализация диаграммы последовательности связана с введением временных ограничений на выполнение отдельных действий в системе. Для простых асинхронных сообщений временные ограничения могут отсутствовать. Однако необходимость синхронизировать сложные потоки управления, как правило, требуют введение в модель таких ограничений.