ПРОГРАММНАЯ ИНЖЕНЕРИЯ

РЕАЛИЗАЦИЯ ПРЕЦЕДЕНТОВ

АНАЛИЗ ПРЕЦЕДЕНТА

- Аналитическая модель классов – это статическая структура системы, а **реализация прецедентов** показывает, как взаимодействуют экземпляры классов анализа для осуществления функциональности системы.
- Цели разработчика:
  - Выяснить, взаимодействие каких классов анализа обеспечивает поведение прецедента;
  - Выяснить, какими сообщениями и в какой последовательности должны обмениваться экземпляры данных классов

РЕАЛИЗАЦИЯ ПРЕЦЕДЕНТОВ

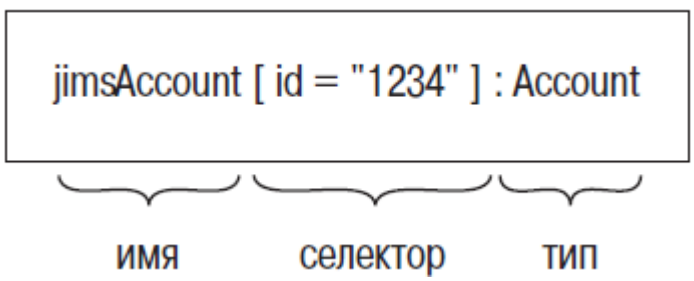
- Реализации прецедентов показывают, как взаимодействуют классы, чтобы реализовать функциональность системы.

Элемент	Назначение
Диаграммы классов анализа	Показывают классы анализа, взаимодействующие для реализации прецедента.
Диаграммы взаимодействий	Показывают взаимодействия определенных экземпляров, реализующих прецедент; это «моментальные снимки» работающей системы.
Особые требования	Процесс реализации прецедентов может выявить новые характерные для прецедента требования; они должны быть зафиксированы.
Уточнение прецедентов	Во время реализации может быть обнаружена новая информация, т. е. происходит обновление исходного прецедента.

ДИАГРАММЫ ПОСЛЕДОВАТЕЛЬНОСТЕЙ

ЛИНИЯ ЖИЗНИ

- **Линия жизни (lifeline)** представляет одного участника взаимодействия, т. е. она представляет, как экземпляр классификатора (объекта, класса, актера и др.) участвует во взаимодействии.



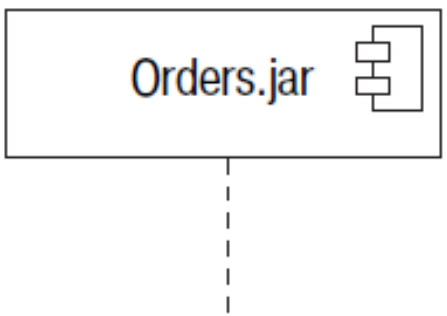
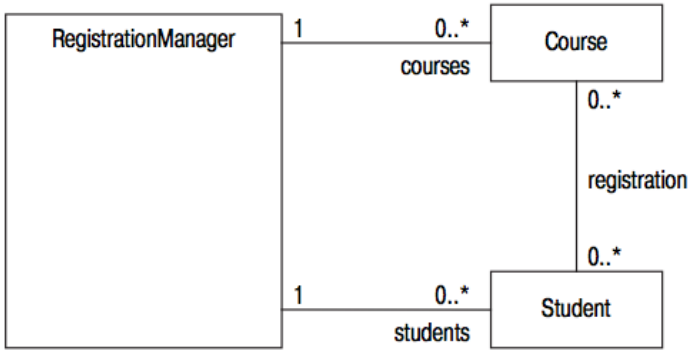
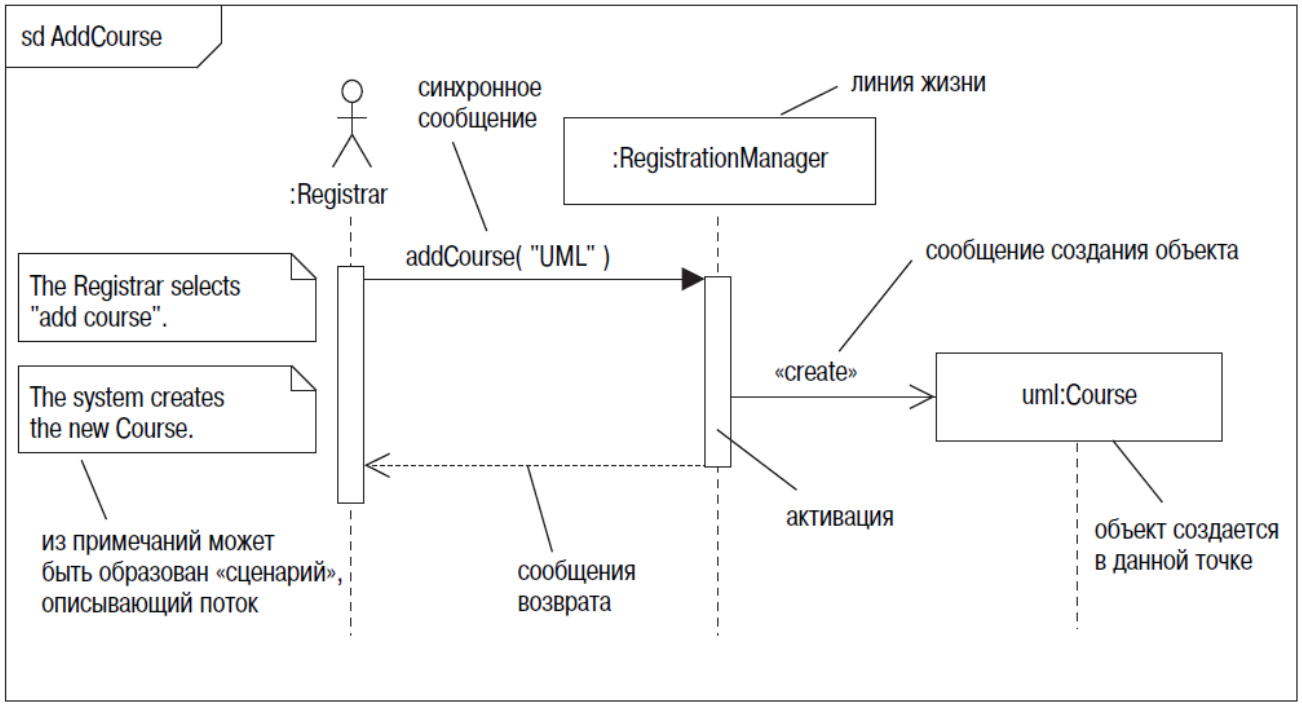


ДИАГРАММА ПОСЛЕДОВАТЕЛЬНОСТИ

- Диаграммы последовательностей представляют взаимодействия между линиями жизни как упорядоченную последовательность событий.

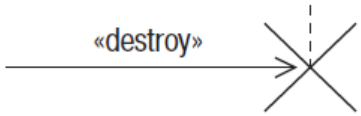
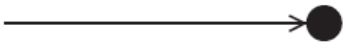




СООБЩЕНИЯ

- Сообщение – это особый вид коммуникации между линиями жизни:
  - вызов операции – сообщение вызова;
  - создание или уничтожение экземпляра – сообщение создания или уничтожения;
  - отправку сигнала.
- Сообщения отображаются в виде стрелок между линиями жизни.

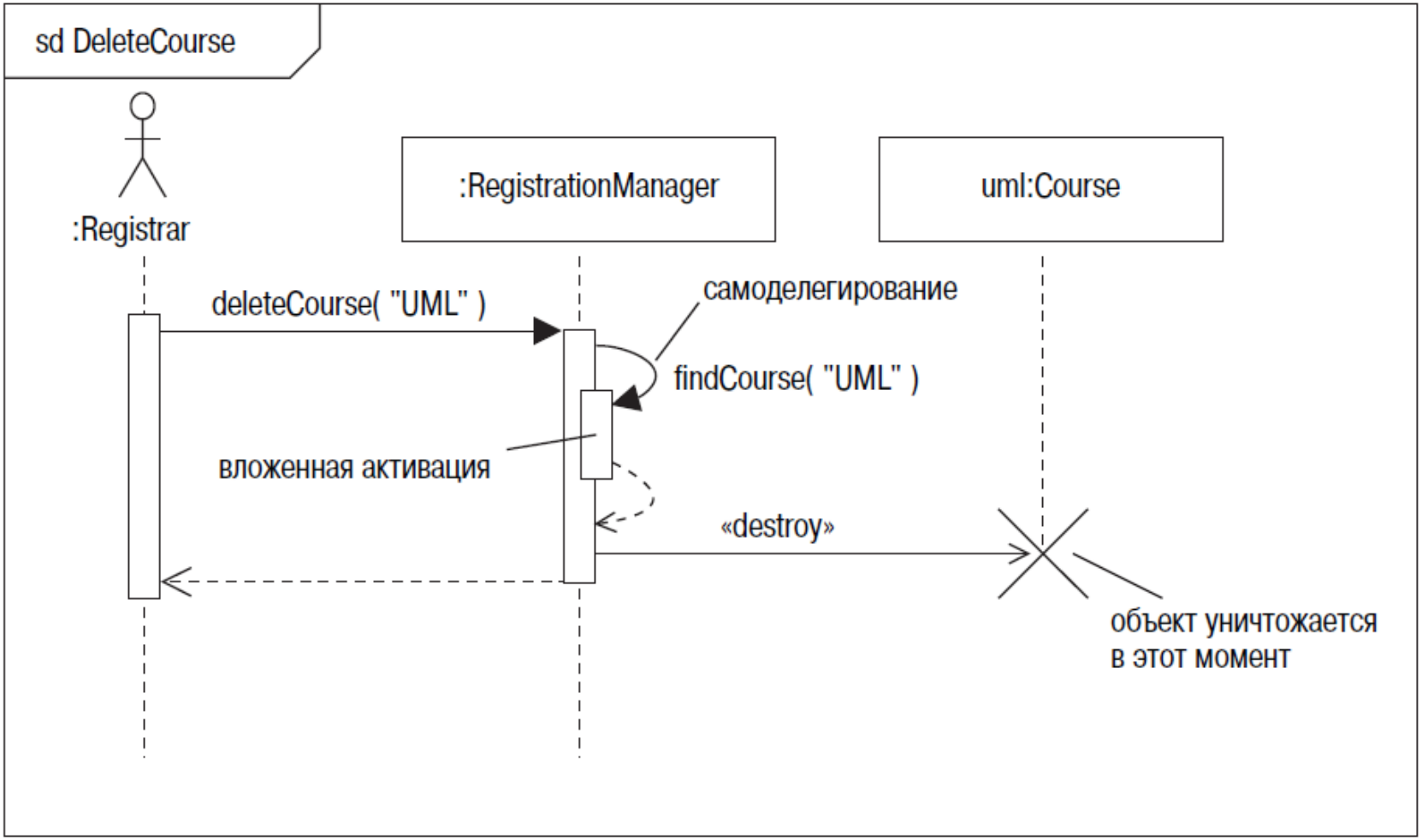
ТИПЫ СООБЩЕНИЙ

Синтаксис	Имя	Семантика
	Синхронное сообщение	Отправитель ожидает завершения выполнения сообщения получателем.
	Асинхронное сообщение	Отправитель посылает сообщение и продолжает исполнение – он *не* ожидает возврата от получателя.
	Возврат	Получатель сообщения возвращает фокус управления отправителю этого сообщения.
	Создание объекта	Отправитель создает экземпляр классификатора, определенного получателем.
	Уничтожение объекта	Отправитель уничтожает получателя. Если у линии жизни есть «хвост», он завершается символом X.
	Найденное сообщение	Отправитель сообщения находится вне области видимости взаимодействия. Используется, когда необходимо показать получение сообщения без указания его источника.
	Потерянное сообщение	Сообщение никогда не достигает точки своего назначения. Может использоваться для обозначения состояний ошибки, при которых пропадают сообщения.

ПРИМЕР

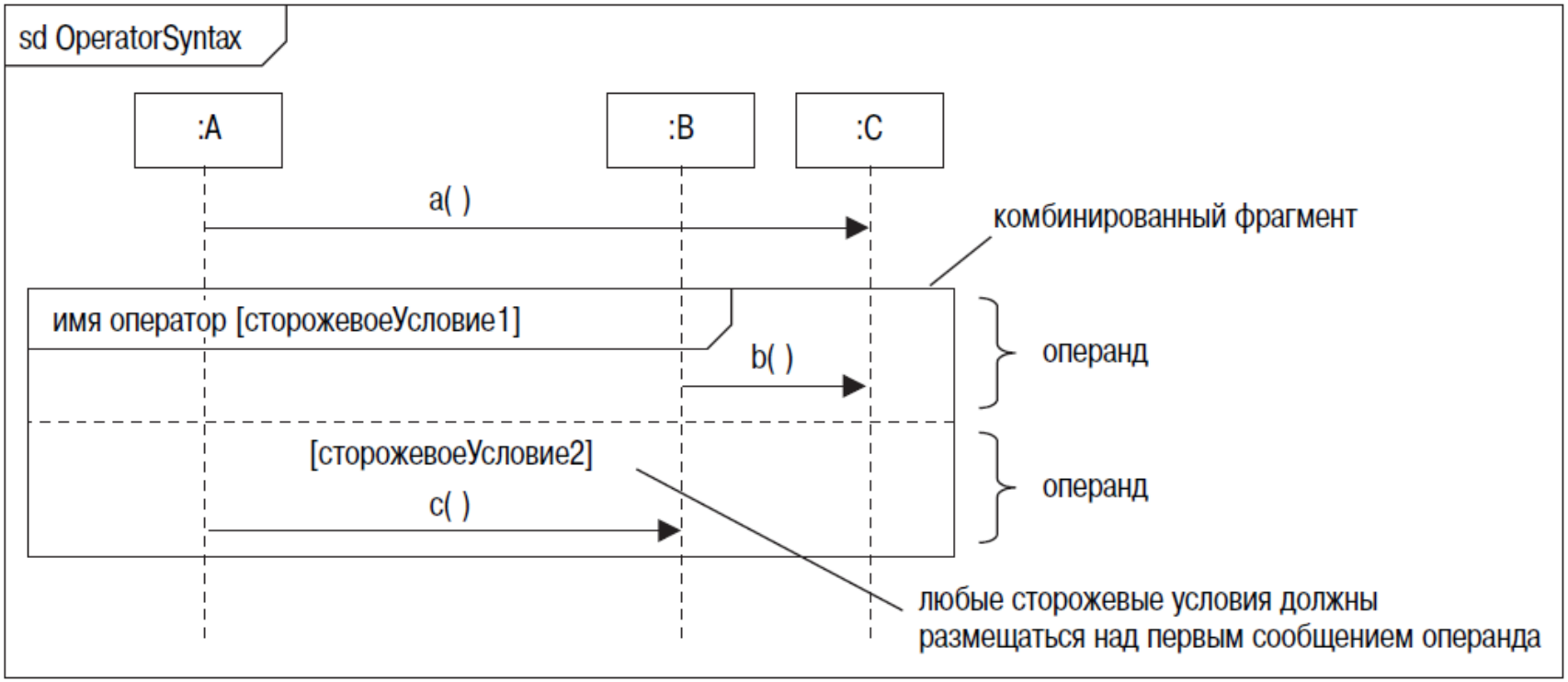
Прецедент: DeleteCourse
ID: 8
Краткое описание: Удаляет курс из системы.
Главные актеры: Registrar
Второстепенные актеры: Нет.
Предусловия: 1. Registrar вошел в систему.
Основной поток: 1. Registrar выбирает «delete cource». 2. Registrar вводит имя курса. 3. Система удаляет курс.
Постусловия: 1. Курс удален из системы.
Альтернативные потоки: CourseDoesNotExist

ПРИМЕР


sd DeleteCourse
:Registrar
:RegistrationManager
uml:Course
deleteCourse("UML")
самоделегирование
findCourse("UML")
вложенная активация
«destroy»
объект уничтожается
в этот момент

КОМБИНИРОВАННЫЕ ФРАГМЕНТЫ

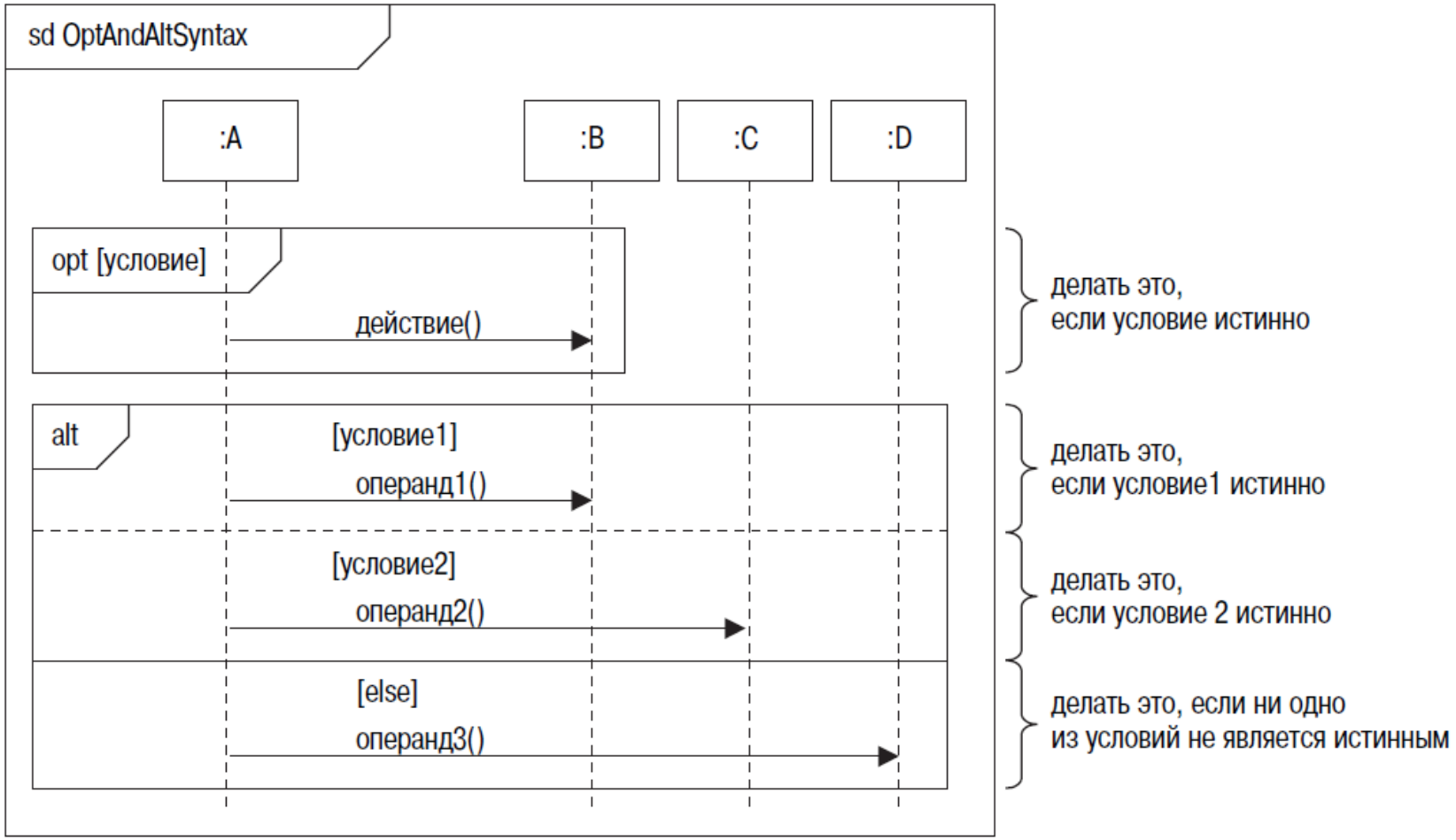
- Комбинированные фрагменты разделяют диаграмму последовательностей на области с различным поведением.



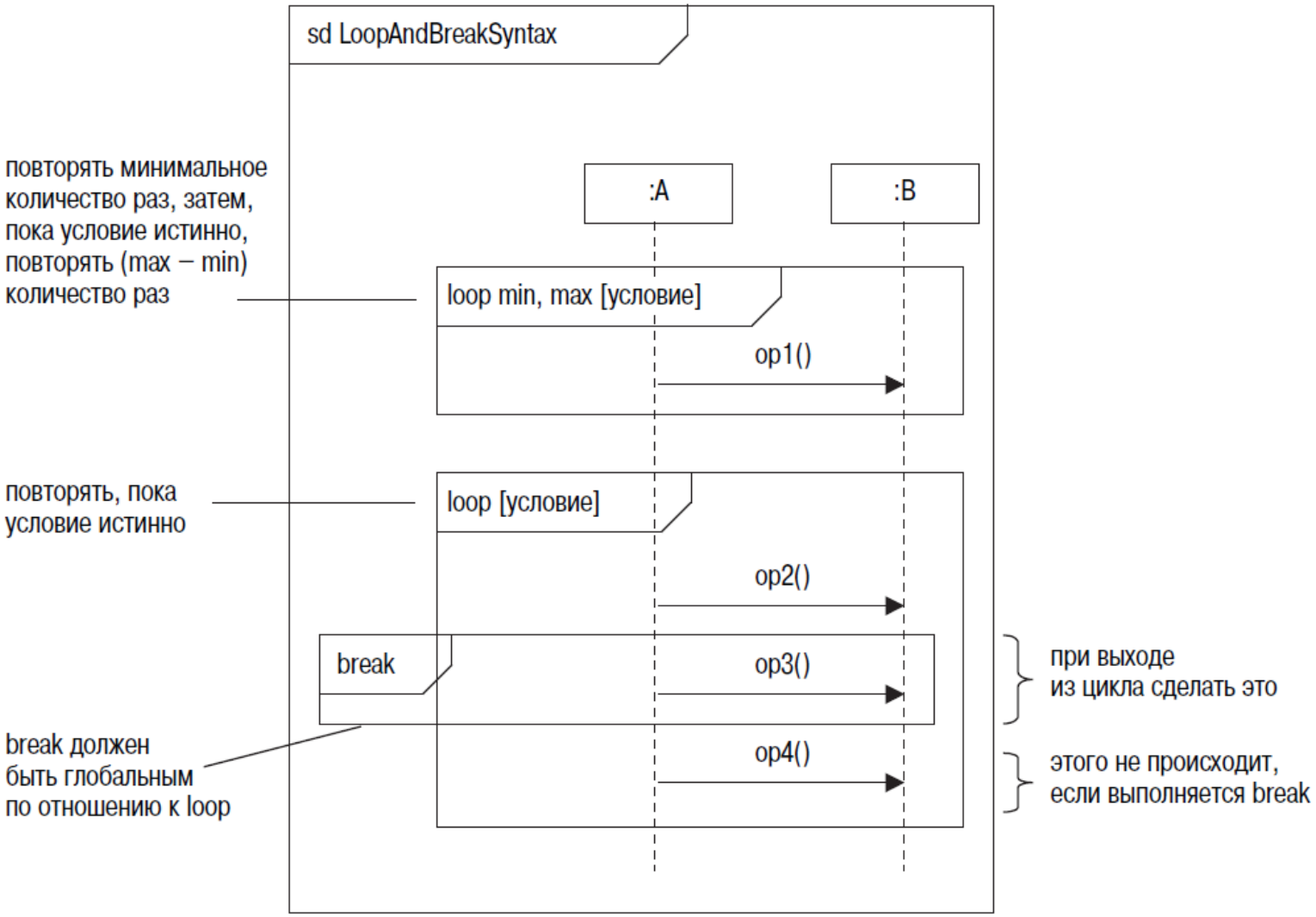
ОСНОВНЫЕ ОПЕРАТОРЫ

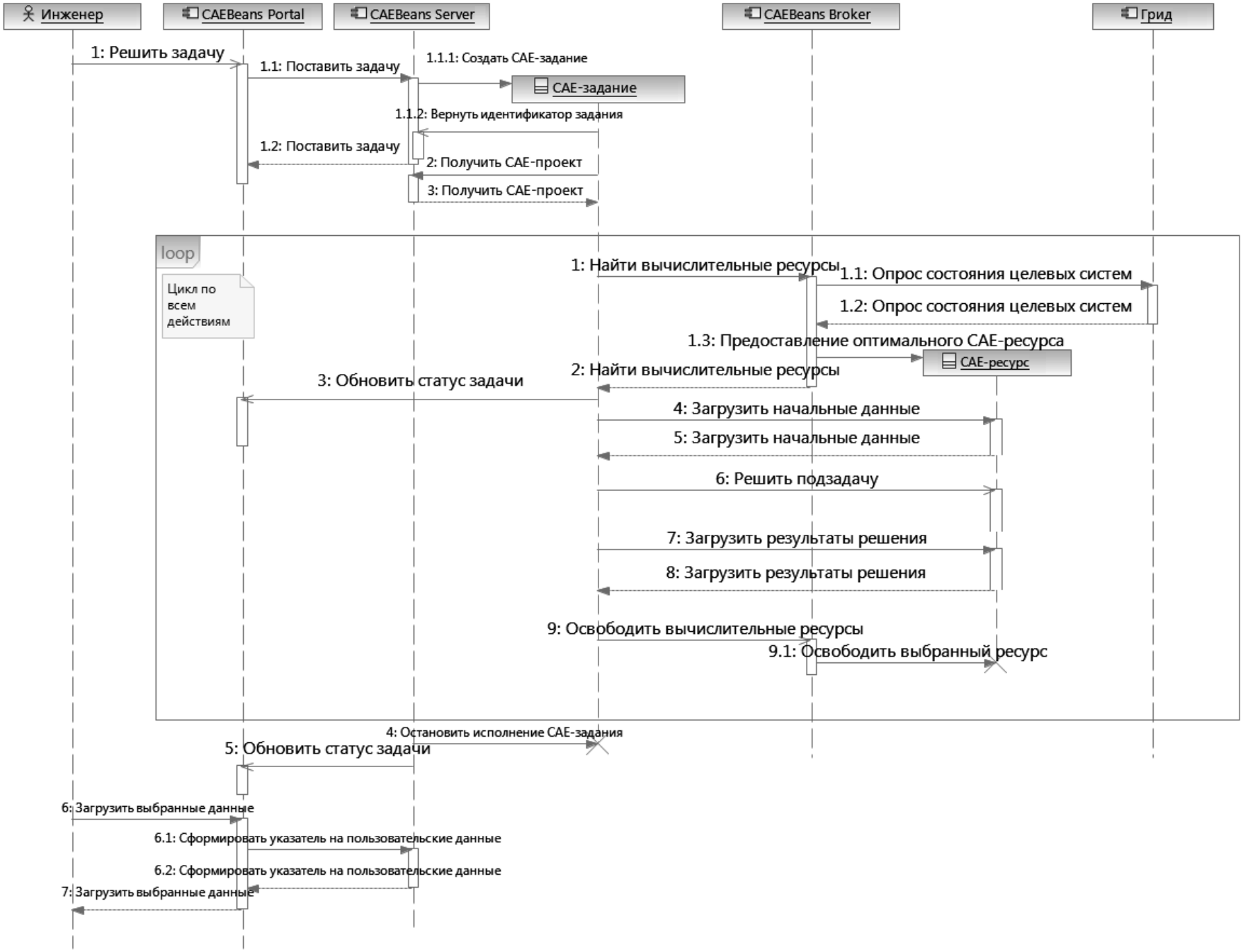
- **opt** (option) - Единственный операнд выполняется, если условие истинно (как if … then).
- **alt** (alternatives) - Выполняется тот операнд, условие которого истинно. Вместо логического выражения может использоваться ключевое слово else (как select … case).
- **loop** - loop min, max [condition] повторять минимальное количество раз, затем, пока условие истинно, повторять еще (max – min) число раз.

ПРИМЕНЕНИЕ OPT И ALT



ПРИМЕНЕНИЕ LOOP




Инженер
CAEBeans Portal
CAEBeans Server
CAEBeans Broker
Грид
1: Решить задачу
1.1: Поставить задачу
1.1.1: Создать CAE-задание
CAE-задание
1.1.2: Вернуть идентификатор задания
1.2: Поставить задачу
2: Получить CAE-проект
3: Получить CAE-проект
loop
Цикл по всем действиям
1: Найти вычислительные ресурсы
1.1: Опрос состояния целевых систем
1.2: Опрос состояния целевых систем
1.3: Предоставление оптимального CAE-ресурса
CAE-ресурс
2: Найти вычислительные ресурсы
3: Обновить статус задачи
4: Загрузить начальные данные
5: Загрузить начальные данные
6: Решить подзадачу
7: Загрузить результаты решения
8: Загрузить результаты решения
9: Освободить вычислительные ресурсы
9.1: Освободить выбранный ресурс
4: Остановить исполнение CAE-задания
5: Обновить статус задачи
6: Загрузить выбранные данные
6.1: Сформировать указатель на пользовательские данные
6.2: Сформировать указатель на пользовательские данные
7: Загрузить выбранные данные

ДИАГРАММЫ ДЕЯТЕЛЬНОСТИ

ДИАГРАММЫ ДЕЯТЕЛЬНОСТИ

- **Диаграммы деятельности** – это ОО блоксхемы.
- Они позволяют моделировать процесс как деятельность, состоящую из коллекции соединенных ребрами узлов.
- Деятельность может быть добавлена к **любому элементу модели** с целью моделирования его поведения. *Обычно: прецеденты, классы, интерфейсы, компоненты, операции.*
- Диаграмма деятельности позволяет моделировать процесс без необходимости определения статической структуры классов и объектов, реализующих процесс.

ДИАГРАММЫ ДЕЯТЕЛЬНОСТИ

Диаграммы деятельности применяются:

В процессе анализа:

- для графического моделирования потока прецедента.
- для моделирования потока между прецедентами.

При проектировании:

- для моделирования деталей операции или алгоритма.

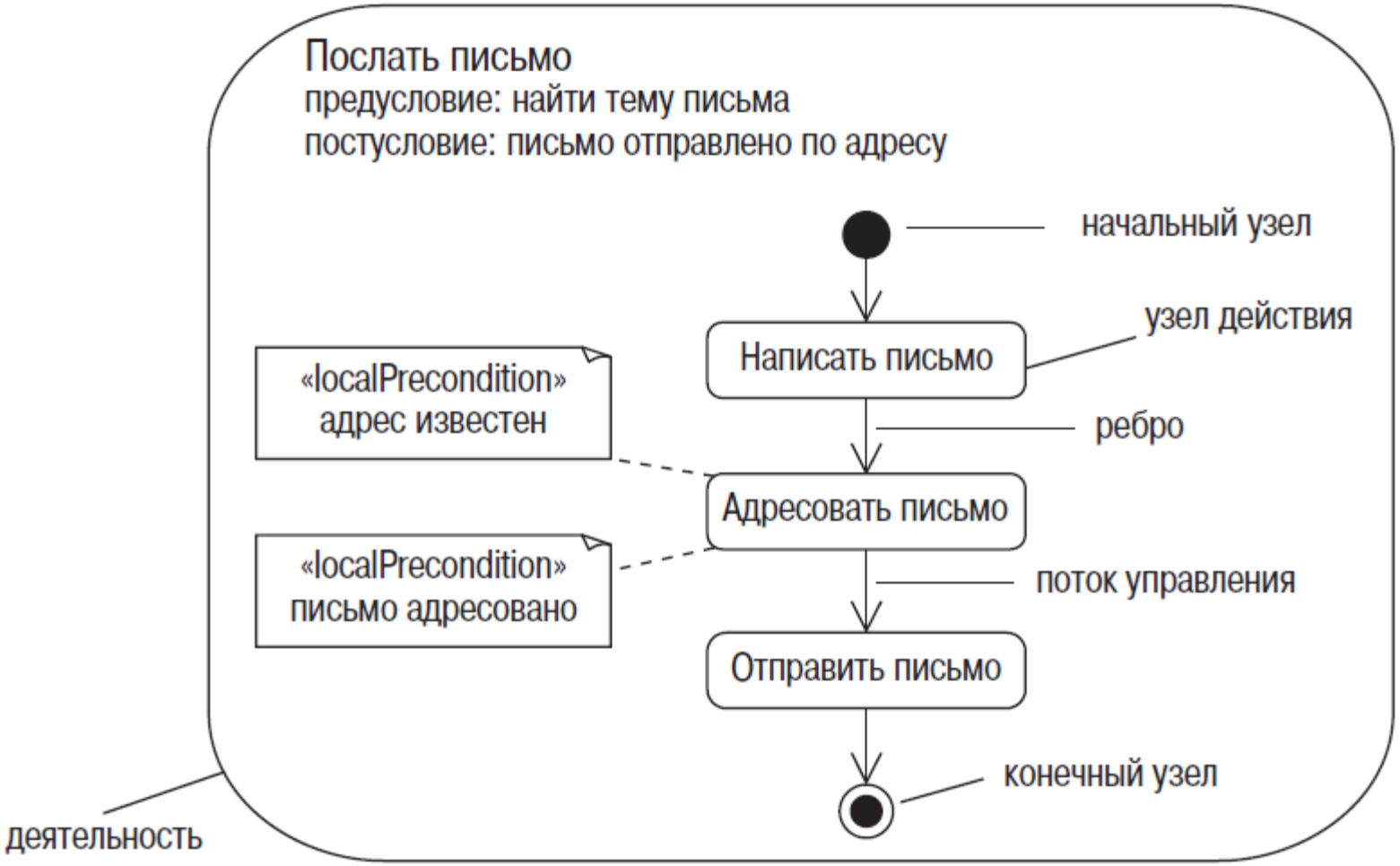
При моделировании деловой активности:

- для моделирования бизнес-процесса

ДЕЯТЕЛЬНОСТИ

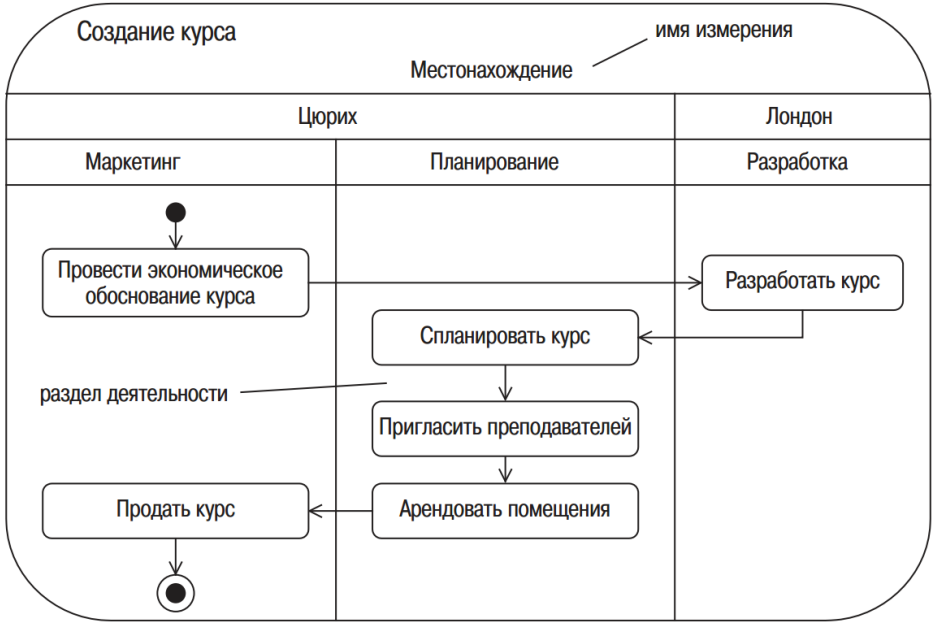
- **Деятельности** – это системы *узлов, соединенных ребрами.* Существует три категории узлов:
  - **Узлы действия** (action nodes) – представляют отдельные единицы работы, элементарные *в рамках деятельности;*
  - **Узлы управления** (control nodes) – управляют потоком деятельности;
  - **Объектные узлы** (object nodes) – представляют объекты, используемые в деятельности.
- Диаграммы деятельности моделируют поведение с помощью **«игры» маркеров (token game)**. Эта игра описывает поток маркеров, движущийся по сети узлов и ребер согласно определенным правилам.

ПРИМЕР ДИАГРАММЫ ДЕЯТЕЛЬНОСТИ



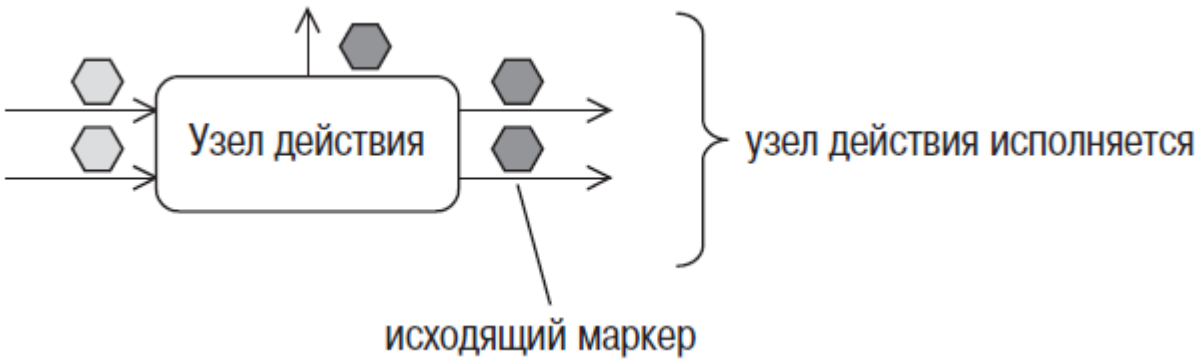
Разделы деятельности

- Чтобы облегчить чтение диаграмм деятельности, можно разбить деятельности на разделы с помощью вертикальных, горизонтальных или кривых линий.
- Каждый раздел деятельности – это группа взаимосвязанных действий с высоким уровнем вложенности.



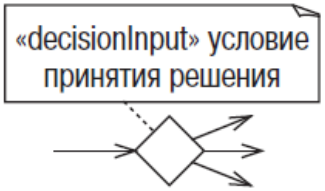
УЗЕЛ ДЕЙСТВИЯ

- Узел действия может инициировать деятельность, поведение или операцию (имена – глагольные группы)
- Узлы действия осуществляют операцию логическое И над своими входящими маркерами – узел не готов к исполнению до тех пор, пока маркеры не будут присутствовать на *всех входящих ребрах.*

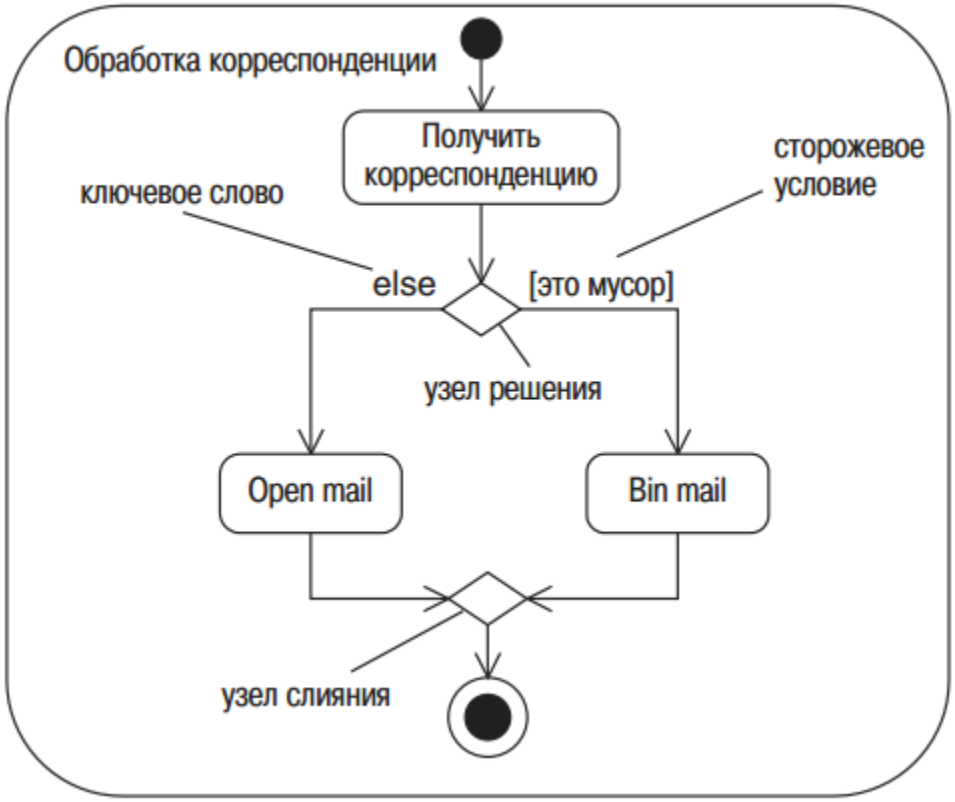


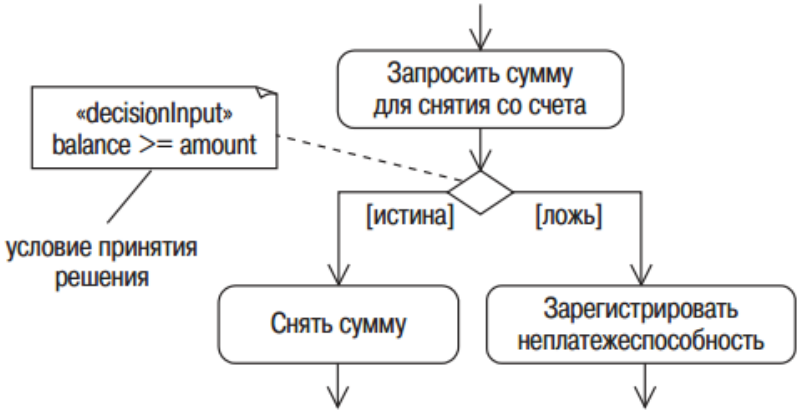
ПРИМЕРЫ ПРИМЕНЕНИЯ УЗЛА ДЕЙСТВИЯ

УЗЛЫ УПРАВЛЕНИЯ

Синтаксис	Имя	Семантика	
	Начальный узел	Указывает, где начинается поток при вызове деятельности.	
	Конечный узел деятельности	Завершает деятельность.	Конечные узлы
	Конечный узел потока	Завершает определенный поток деятельности – другие потоки не затрагиваются.	
	Узел решения	Поток проходит по исходящему ребру, сторожевое условие которого истинно. Может иметь входные данные (необязательно).	
	Узел слияния	Копирует входные маркеры в единственное выходное ребро.	
	Узел ветвления	Разделяет поток на несколько параллельных потоков.	
	Узел объединения	Синхронизирует несколько параллельных потоков. Может иметь описание объединения (не обязательно) для изменения его семантики.	

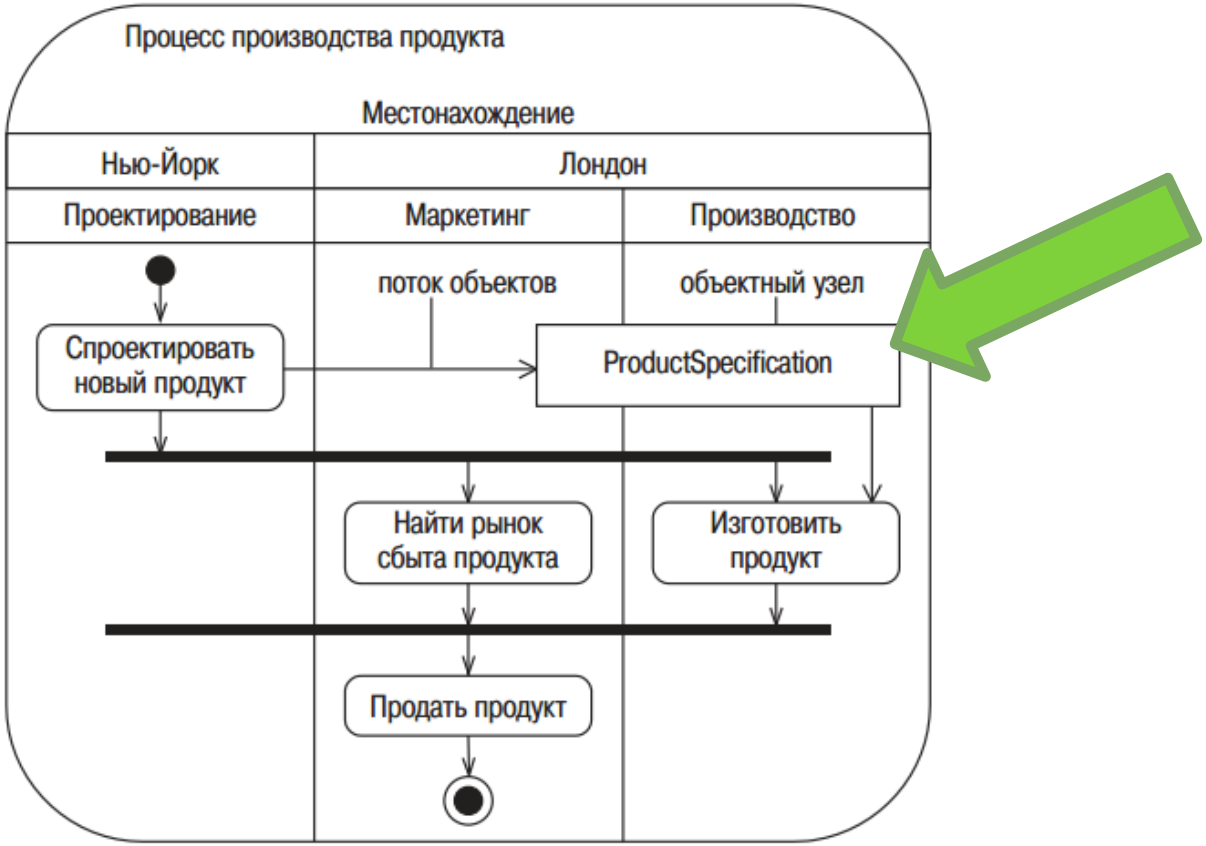
Узлы решения и слияния





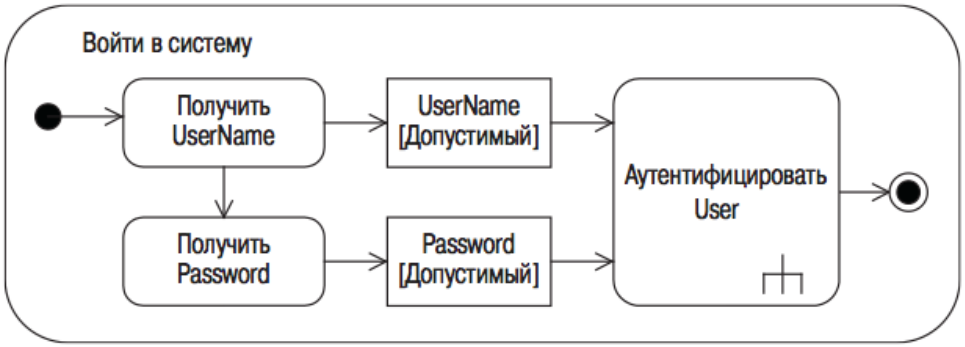
Объектные узлы

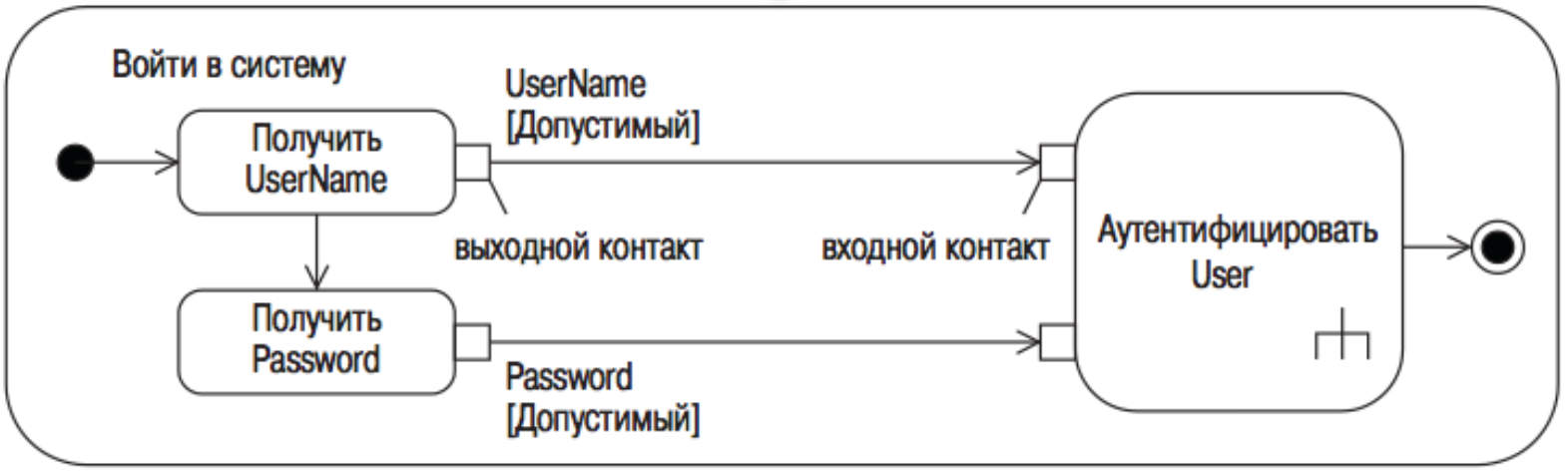
Объектные узлы – это специальные узлы, показывающие, что экземпляры конкретного классификатора (класса) доступны в данной точке деятельности.



Контакты

Контакт – это **объектный узел**, представляющий один вход или выход из действия.



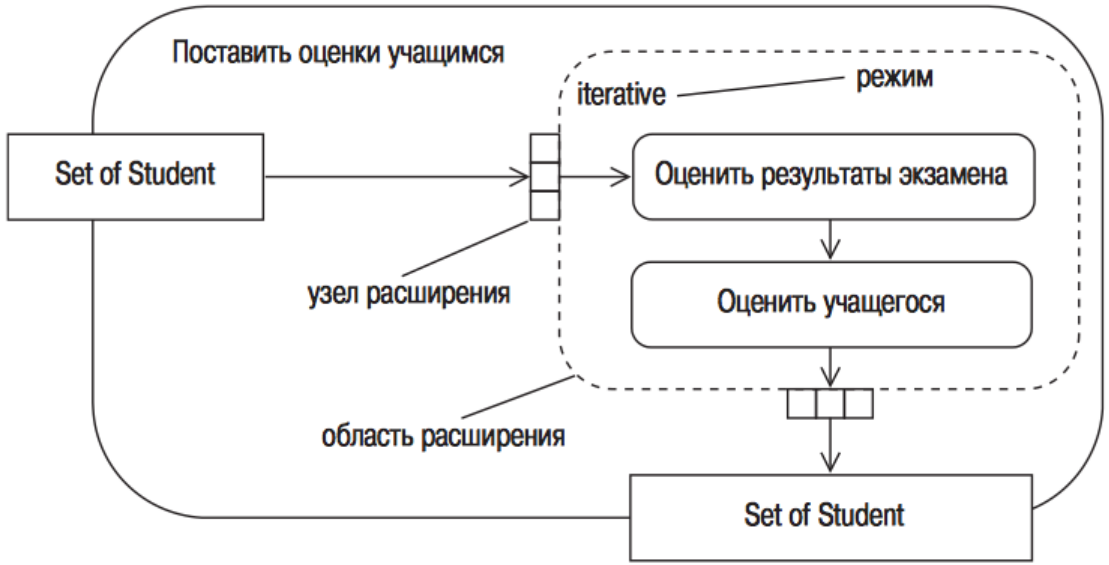


Узлы расширения

Узел расширения – это объектный узел, который представляет коллекцию объектов, входящую в или выходящую из области расширения. Область расширения исполняется по одному разу для каждого входного элемента.

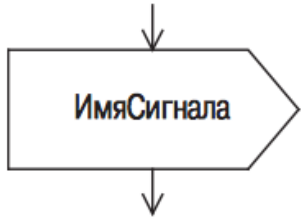
У каждой области расширения есть режим, определяющий порядок обработки элементов входной коллекции:

- **iterative (итеративный)** – каждый элемент входной коллекции обрабатывается последовательно;
- **parallel (параллельный)** – элементы входной коллекции обрабатываются параллельно;
- **stream (потоковый)** – элементы входной коллекции обрабатываются в порядке поступления в узел.



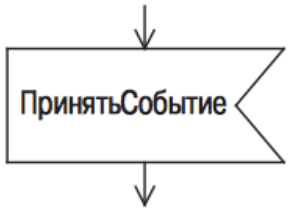
Отправка сигналов

- **Сигнал** представляет асинхронно передаваемую между объектами информацию. Сигнал моделируется как класс, отмеченный стереотипом «signal» (сигнал).



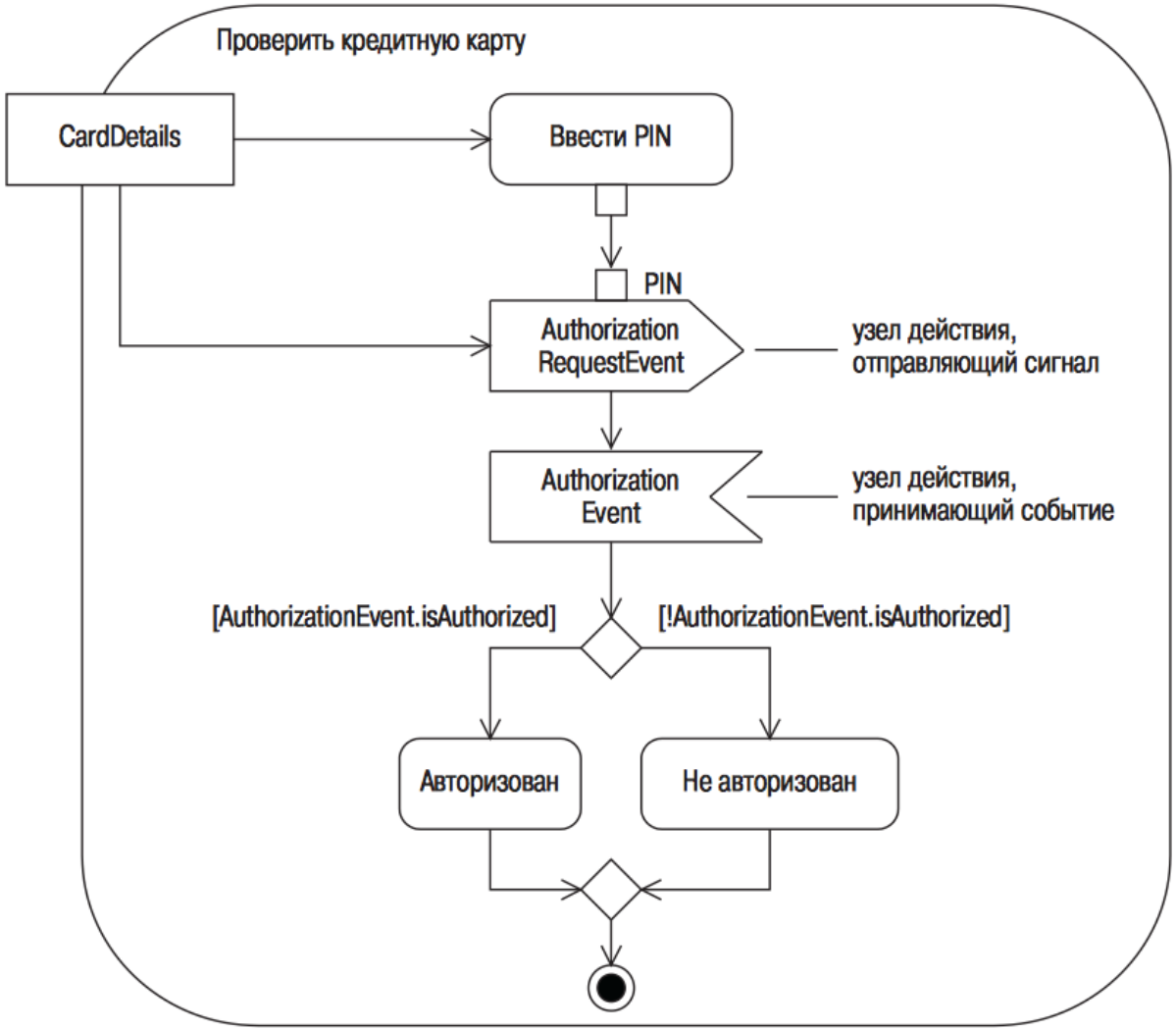
- Сигнал можно послать с помощью узла действия, отвечающего за отправку сигнала. Он посылает сигнал асинхронно – деятельность, отравившая сигнал не ожидает подтверждения получения сигнала:
  - Посылающая сигнал деятельность инициируется тогда, когда маркер одновременно присутствует на всех ее входных ребрах. Если у сигнала имеются входные контакты, он должен получить входной объект соответствующего типа для каждого из своих атрибутов.
  - При выполнении действия создается и посылается объект сигнала.
  - Действие отправки не ожидает подтверждения принятия сигнала – оно асинхронно.
  - Действие завершается, и маркеры управления предлагаются на его выходных ребрах.

Прием событий



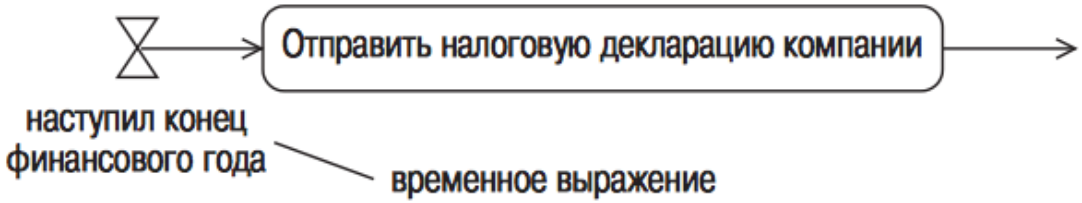
- Узел действия, принимающий событие, может не иметь ни одного ребра, или же имеет одно входное ребро.
- Он ожидает асинхронных событий, выявленных его контекстом-владельцем, и предлагает их на своем единственном **выходном** ребре:
  - Действие приема события запускается входящим ребром управления; если входящих ребер нет, оно запускается при вызове деятельности-владельца.
  - Действие ожидает получения события определенного типа. Это событие называют триггером (trigger).
  - Когда действие получает триггер события соответствующего типа, оно выдает маркер, описывающий событие. Если событие было сигналом, маркер является сигналом.
  - Действие продолжает принимать события до тех пор, пока выполняется деятельность.

Отправка сигналов и прием событий

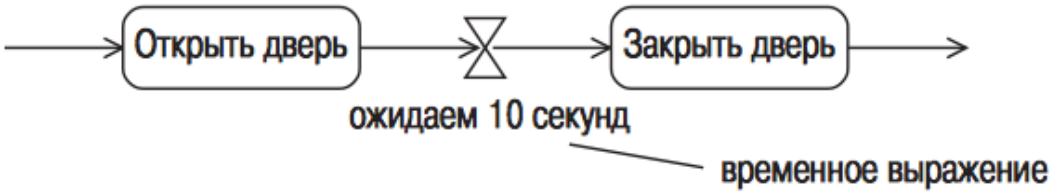


Обработка времени

- Для обработки событий по времени, можно использовать узел действия, реагирующий на время. Этот тип узла имеет временное выражение и генерирует событие времени, когда это выражение становится истинным.
- Если такой узел не имеет входящего ребра, он будет работать в «глобальном» времени, генерируя события самостоятельно.

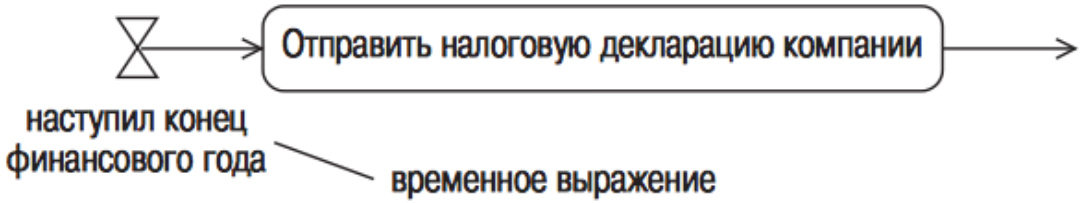


- Если у узла есть входное ребро, он будет активирован только при появлении маркера действия на входном ребре.

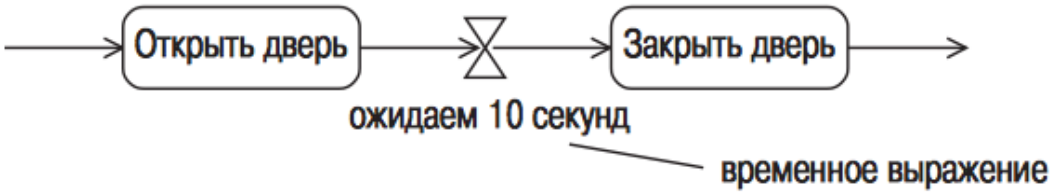


Обработка времени

- Для обработки событий по времени, можно использовать узел действия, реагирующий на время. Этот тип узла имеет временное выражение и генерирует событие времени, когда это выражение становится истинным.
- Если такой узел не имеет входящего ребра, он будет работать в «глобальном» времени, генерируя события самостоятельно.



- Если у узла есть входное ребро, он будет активирован только при появлении маркера действия на входном ребре.



Пример диаграммы деятельности