Диаграмма последовательности централизованного управления

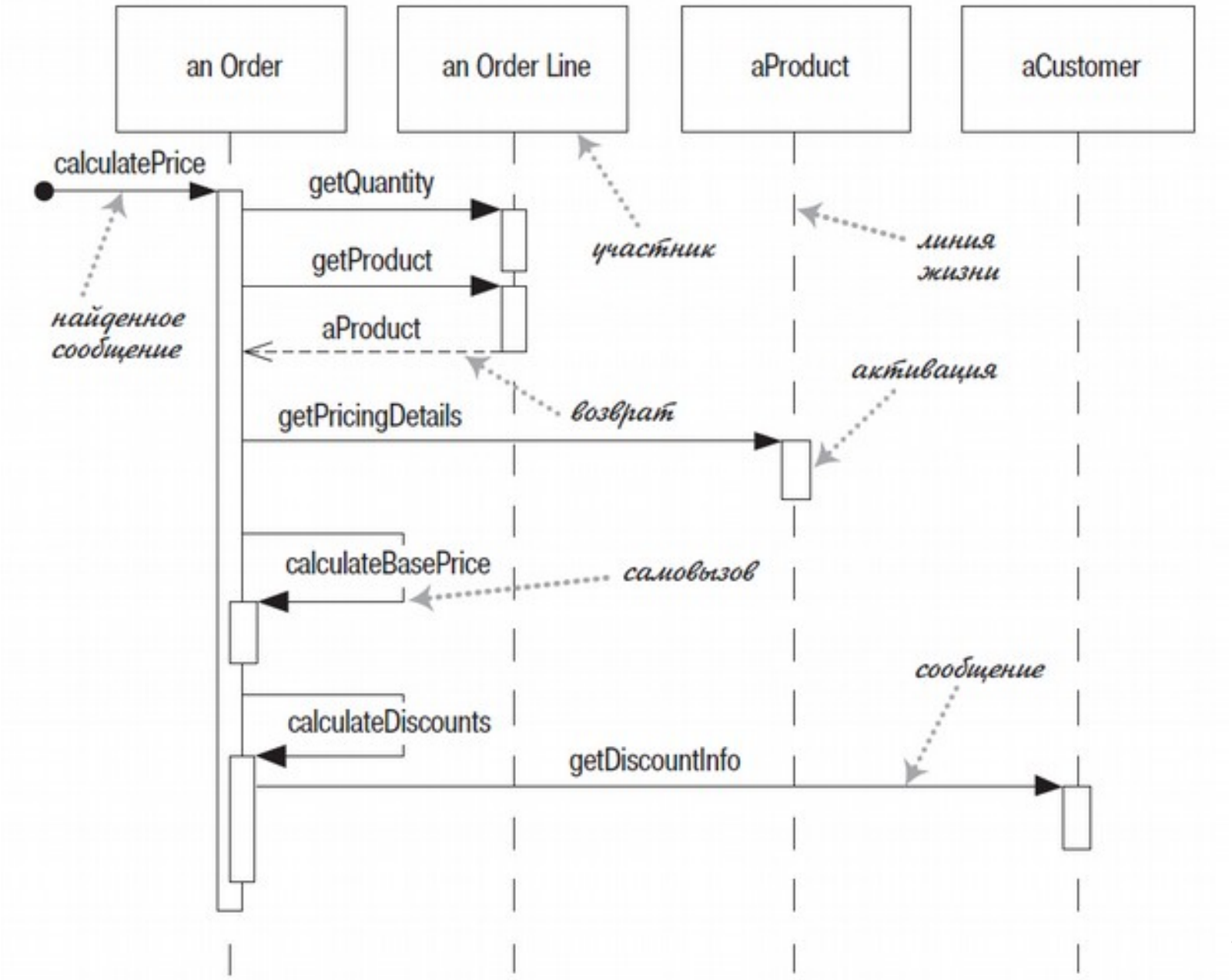
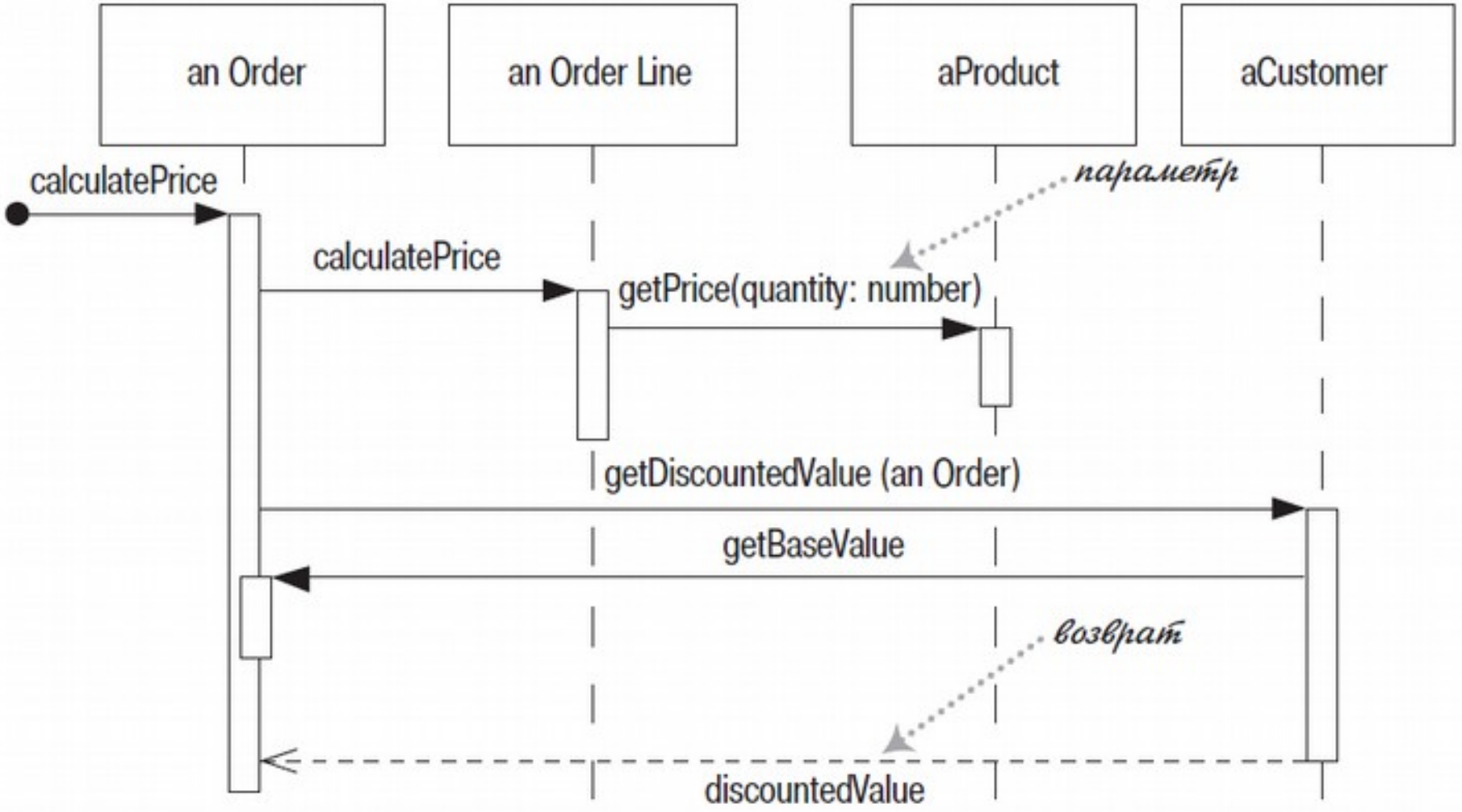
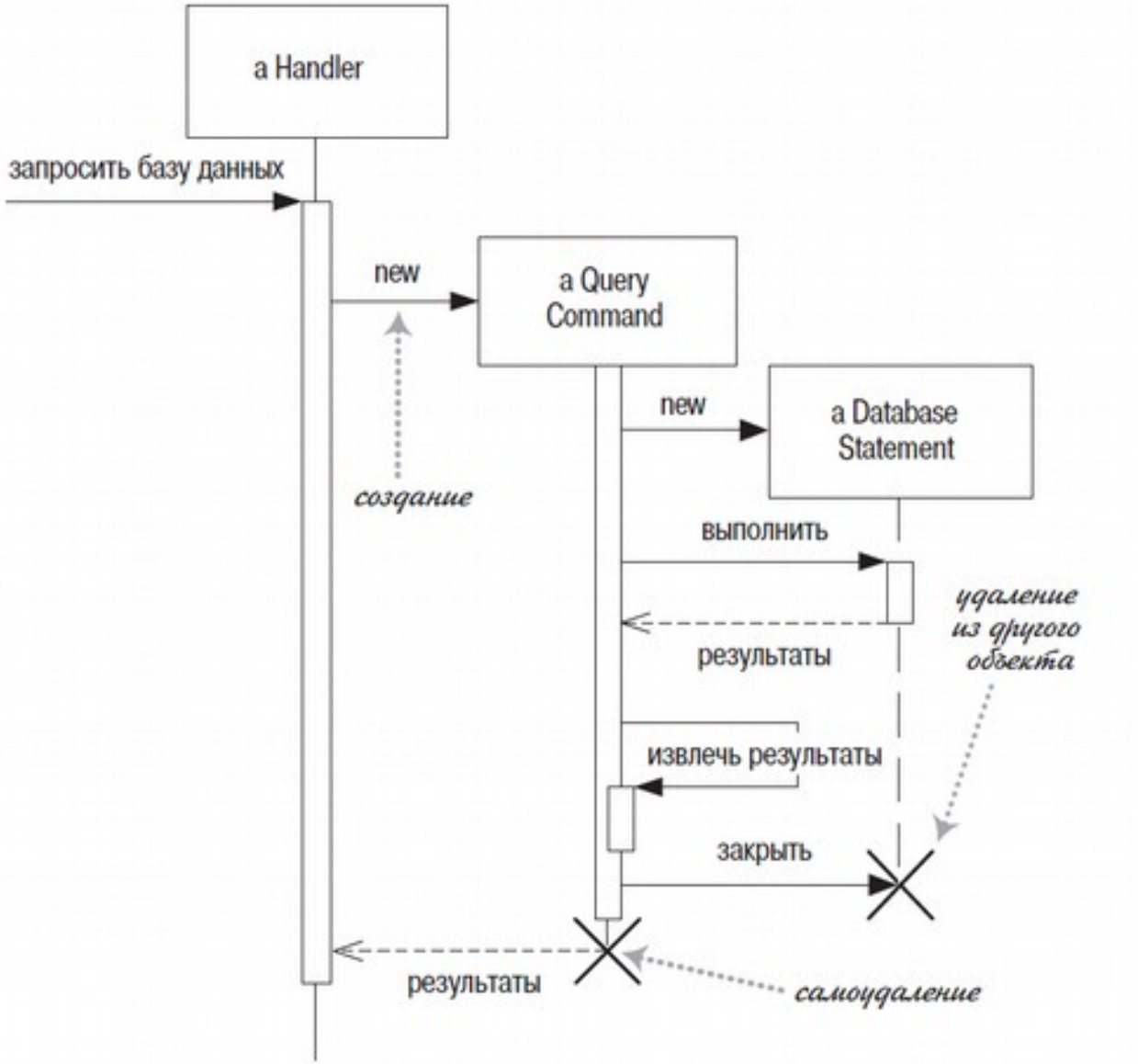


Диаграмма последовательности для распределенного управления

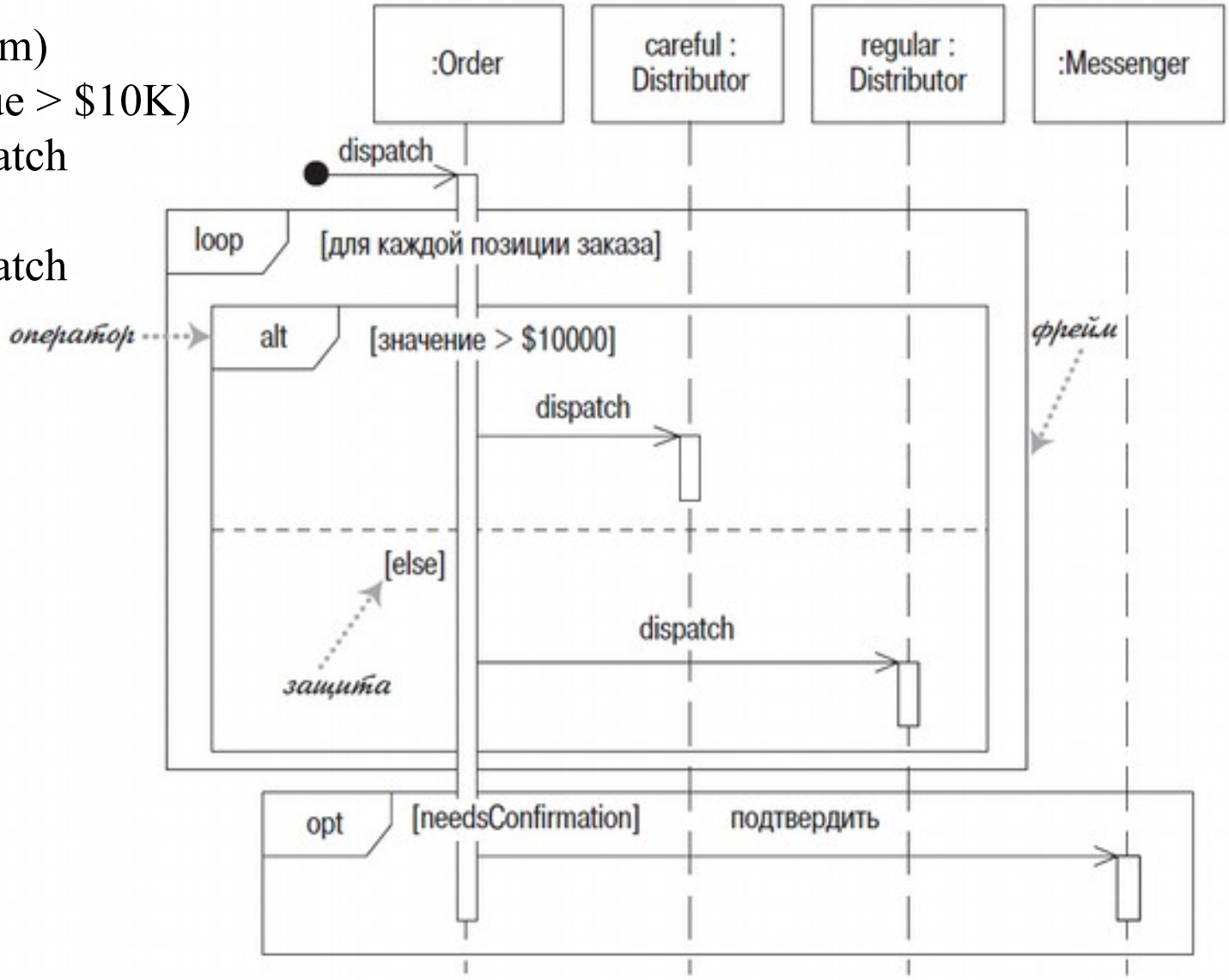


Создание и удаление участников



Фреймы взаимодействия

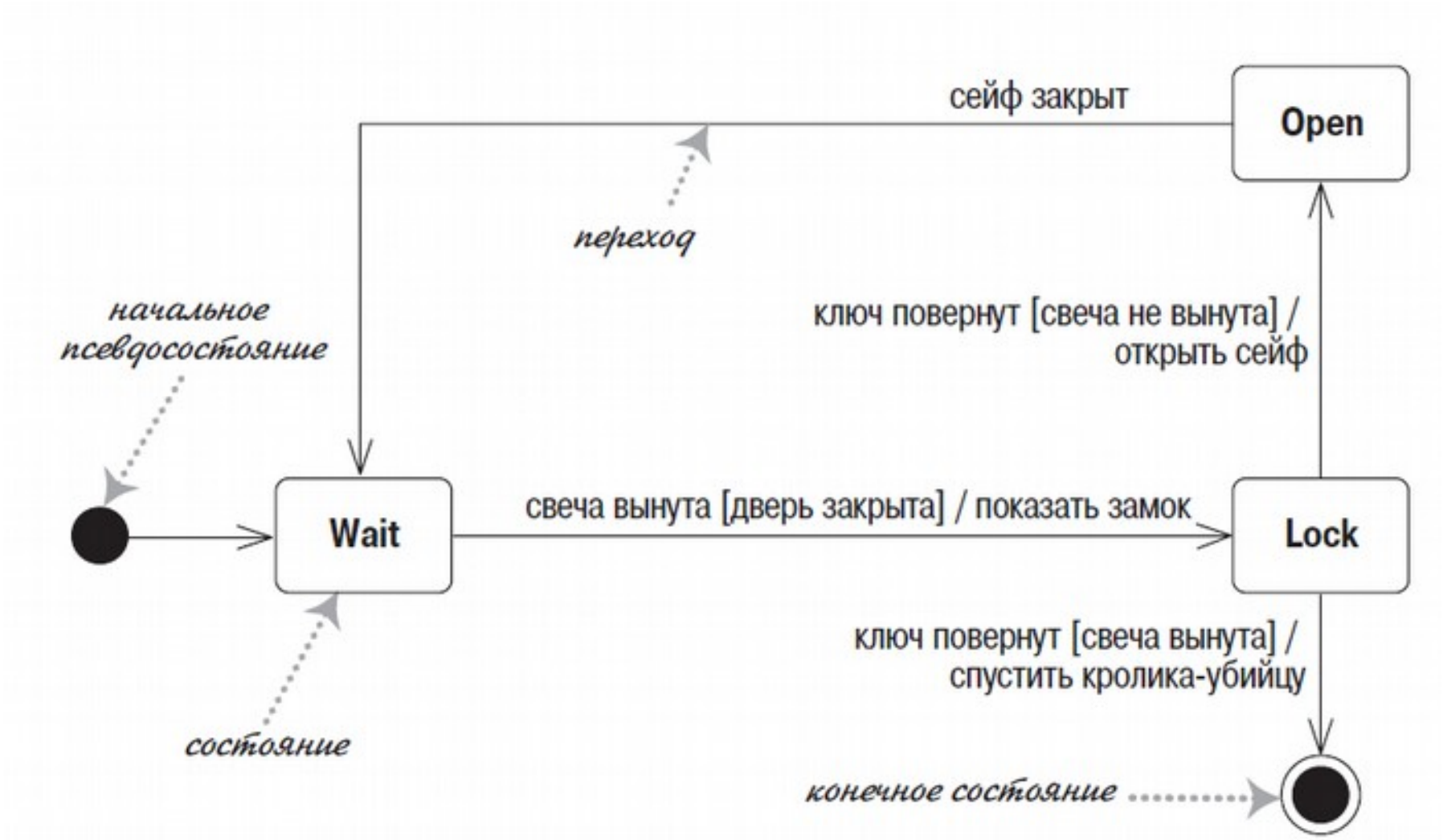
```
foreach (lineitem)
if (product.value > $10K)
    careful.dispatch
else
    regular.dispatch
end if
```



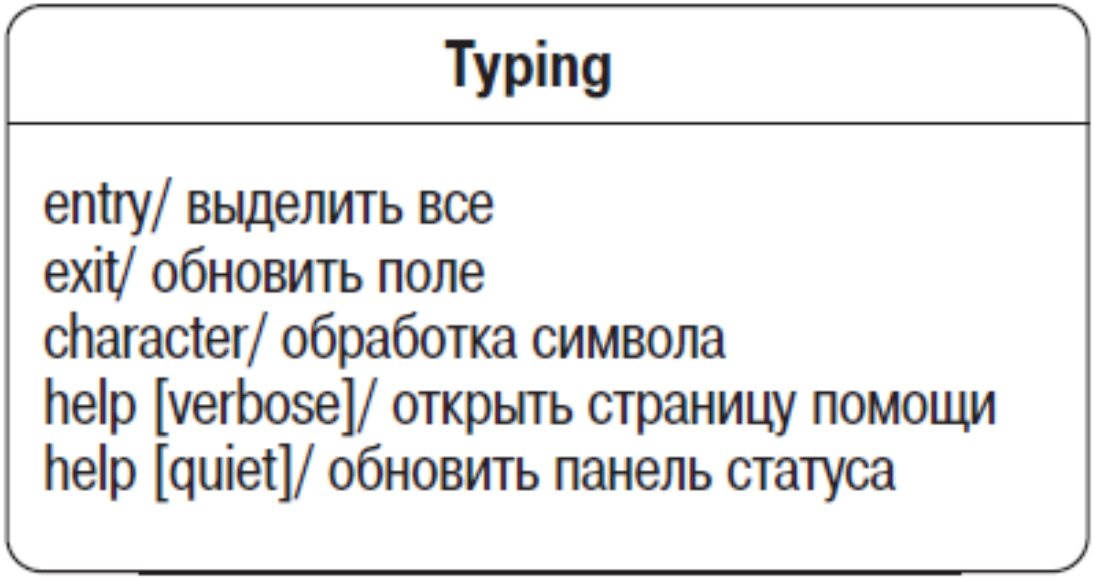
Общепринятые операторы для фреймов взаимодействия

Оператор	Значение
`alt`	Несколько альтернативных фрагментов (alternative); выполняется только тот фрагмент, условие которого истинно (рис. 4.4)
`opt`	Необязательный (optional) фрагмент; выполняется, только если условие истинно. Эквивалентно `alt` с одной веткой (рис. 4.4)
`par`	Параллельный (parallel); все фрагменты выполняются параллельно
`loop`	Цикл (loop); фрагмент может выполняться несколько раз, а защита обозначает тело итерации (рис. 4.4)
`region`	Критическая область (critical region); фрагмент может иметь только один поток, выполняющийся за один прием
`neg`	Отрицательный (negative) фрагмент; обозначает неверное взаимодействие
`ref`	Ссылка (reference); ссылается на взаимодействие, определенное на другой диаграмме. Фрейм рисуется, чтобы охватить линии жизни, вовлеченные во взаимодействие. Можно определять параметры и возвращать значение
`sd`	Диаграмма последовательности (sequence diagram); используется для очерчивания всей диаграммы последовательности, если это необходимо

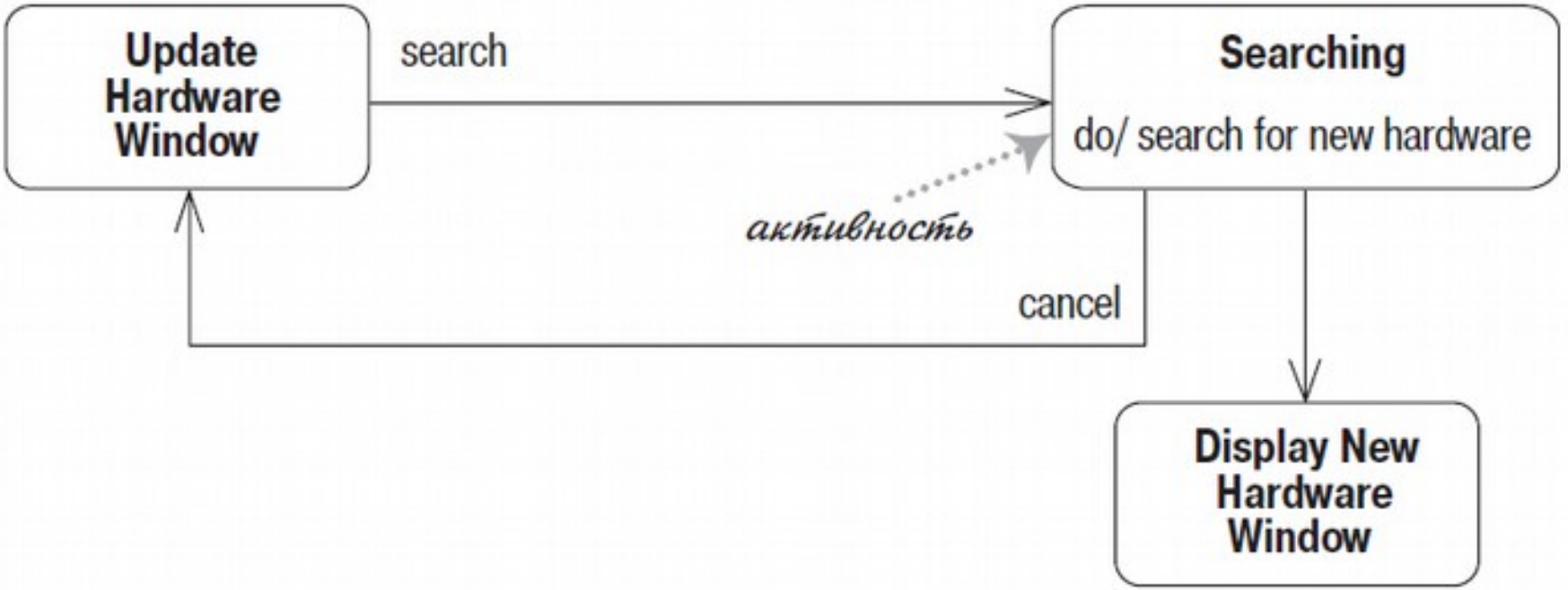
Простая диаграмма состояний



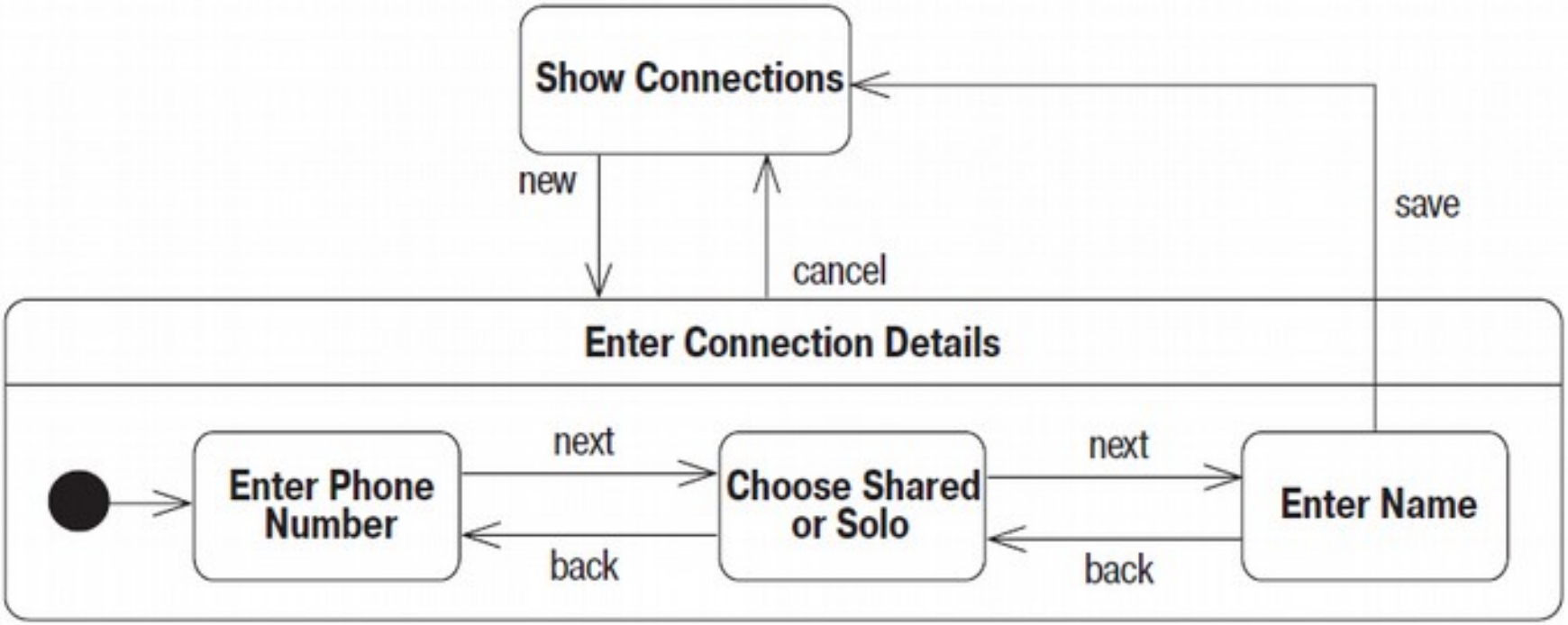
Внутренние события, показанные в состоянии набора текста в текстовом поле



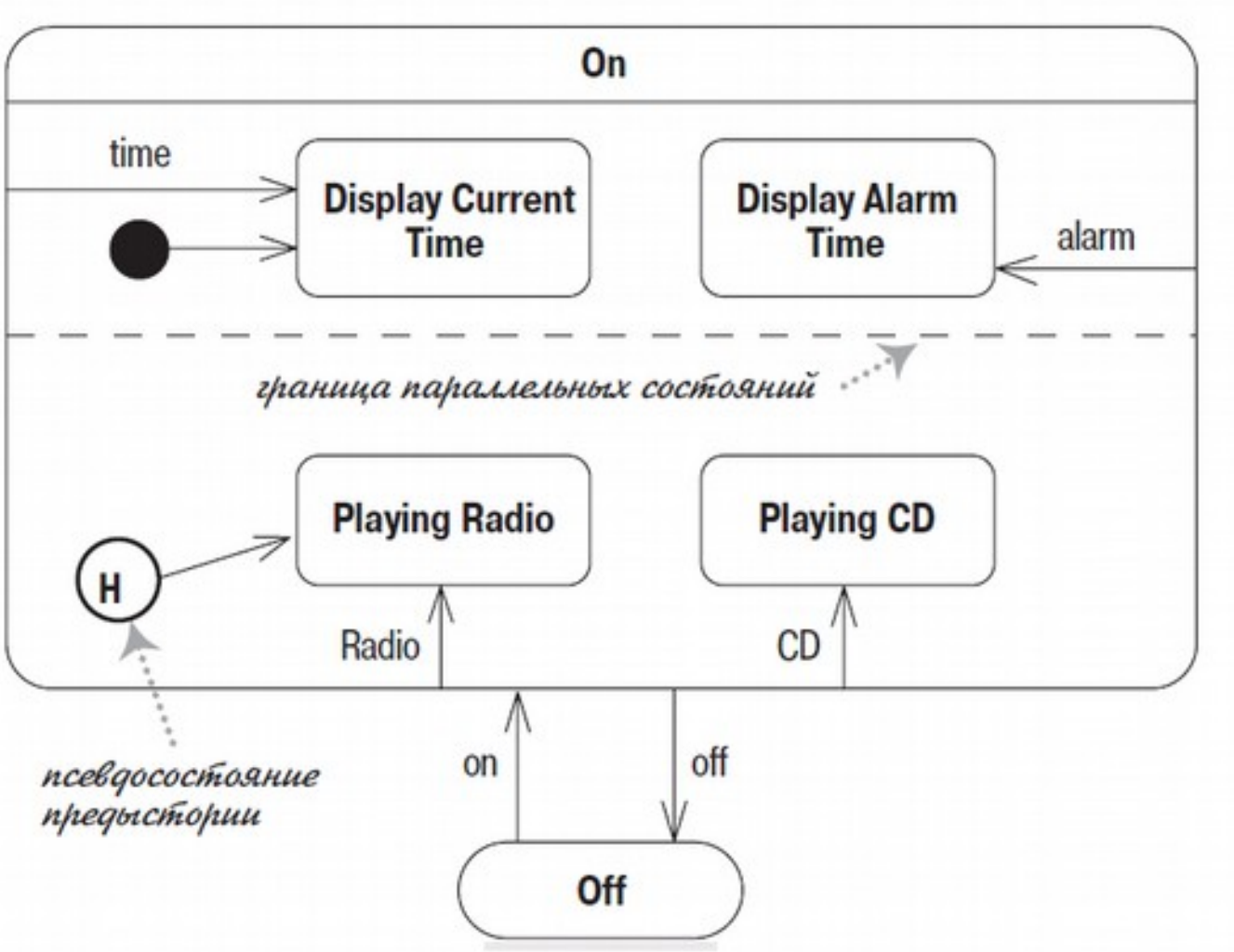
Состояние с активностью



Суперсостояние с вложенными подсостояниями



Параллельные состояния

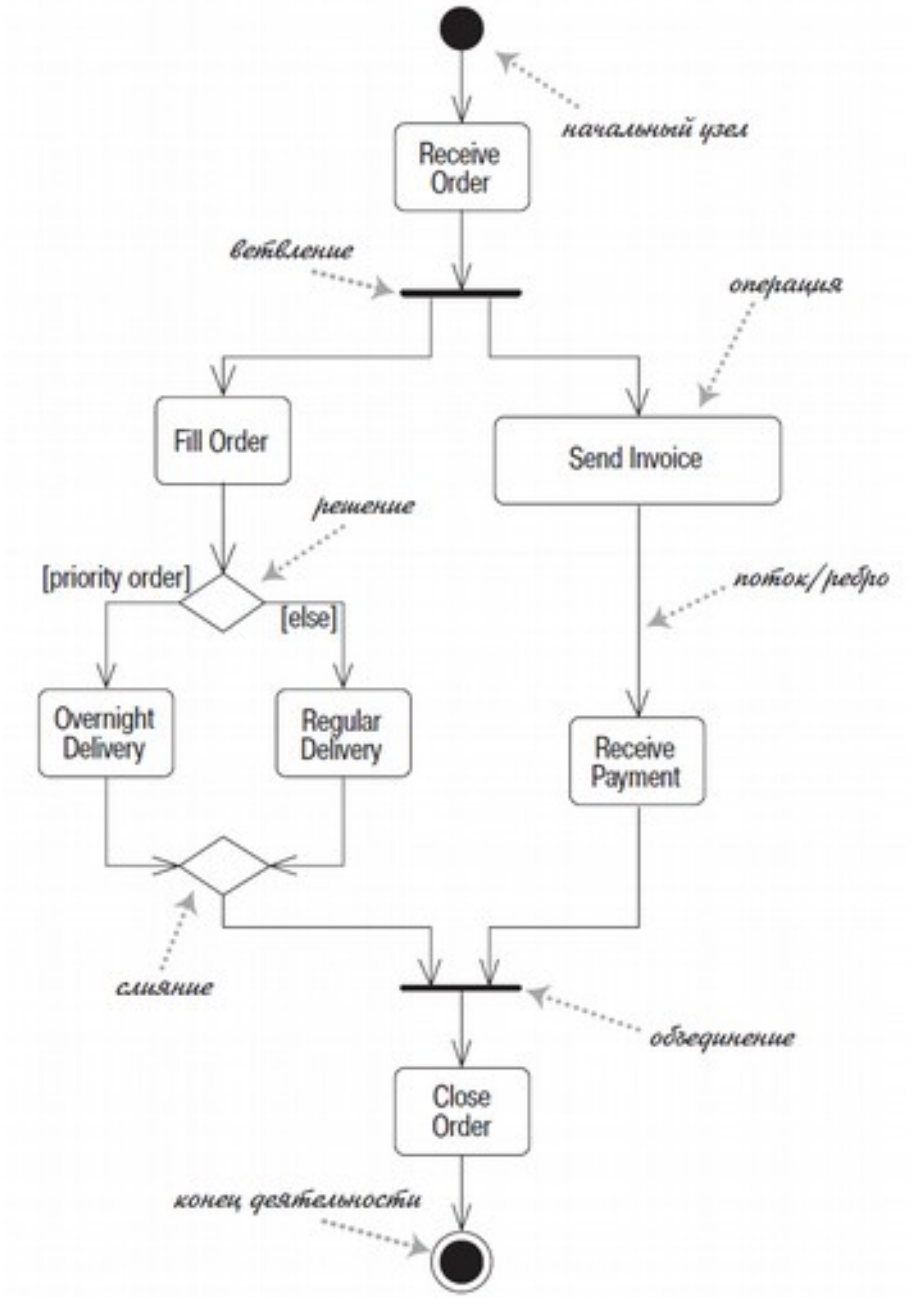


```
public void HandleEvent (PanelEvent anEvent) {
   switch (CurrentState) {
      case PanelState.Open :
         switch (anEvent) {
            case PanelEvent.SafeClosed :
               CurrentState = PanelState.Wait;
               break;
         }
         break;
      case PanelState.Wait :
         switch (anEvent) {
            case PanelEvent.CandleRemoved :
               if (isDoorOpen) {
                  RevealLock();
                  CurrentState = PanelState.Lock;
               }
              break;
         }
         break;
      case PanelState.Lock :
         switch (anEvent) {
            case PanelEvent.KeyTurned :
               if (isCandleIn) {
                  OpenSafe();
                  CurrentState = PanelState.Open;
               } else {
                  ReleaseKillerRabbit();
                  CurrentState = PanelState.Final;
               }
               break;
         }
         break;
     }
  }
}
```

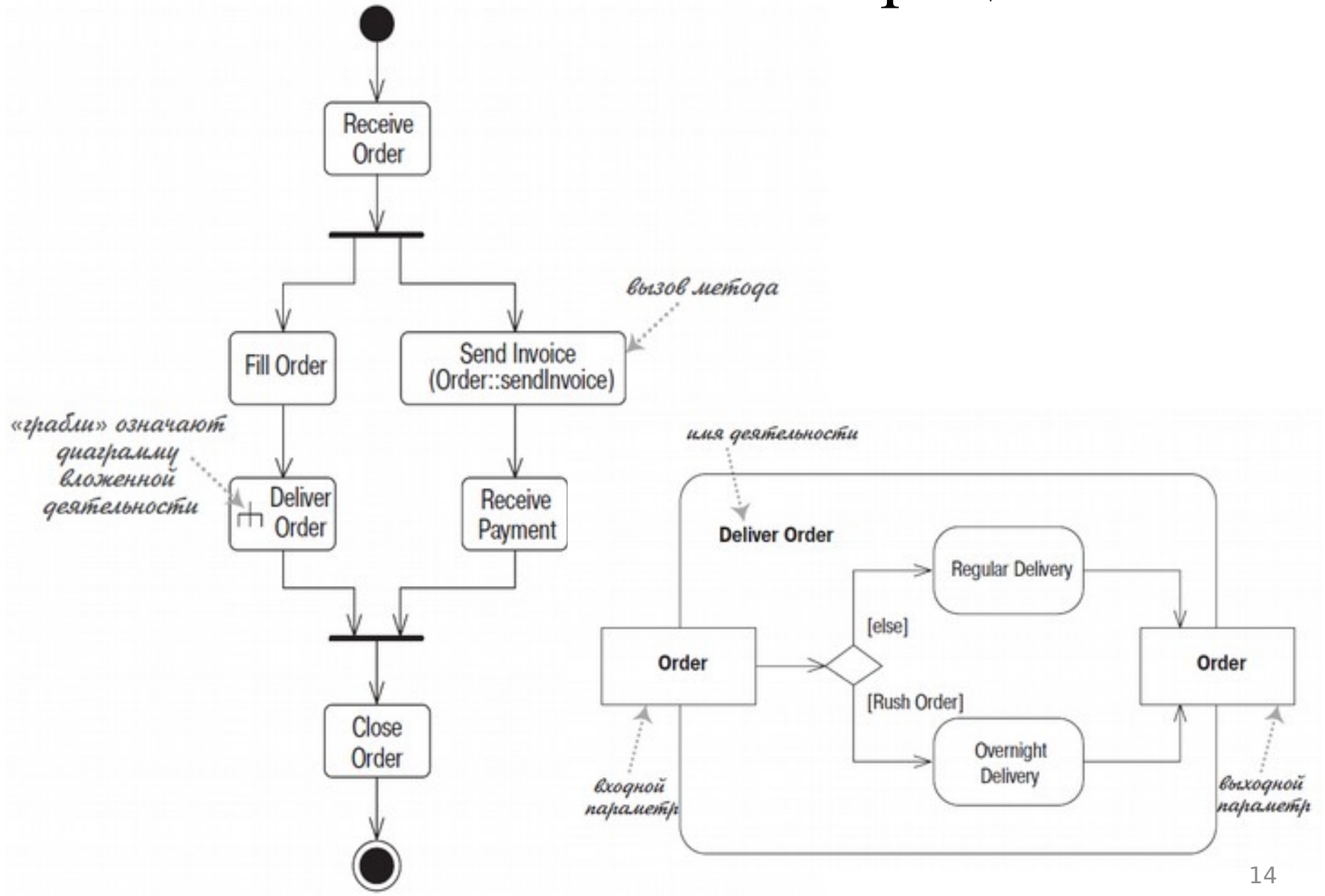
Таблица состояний для открытия сейфа

Исходное состояние	Целевое состояние	Событие	Защита	Процедура
Wait	Lock	Candle removed (свеча удалена)	Door open (дверца открыта)	Reveal lock (показать замок)
Lock	Open	Key turned (ключ повернут)	Candle in (свеча на месте)	Open safe (открыть сейф)
Lock	Final	Key turned (ключ повернут)	Candle out (свеча удалена)	Release killer rabbit (освободить убийцу-кролика)
Open	Wait	Safe closed (сейф закрыт)		

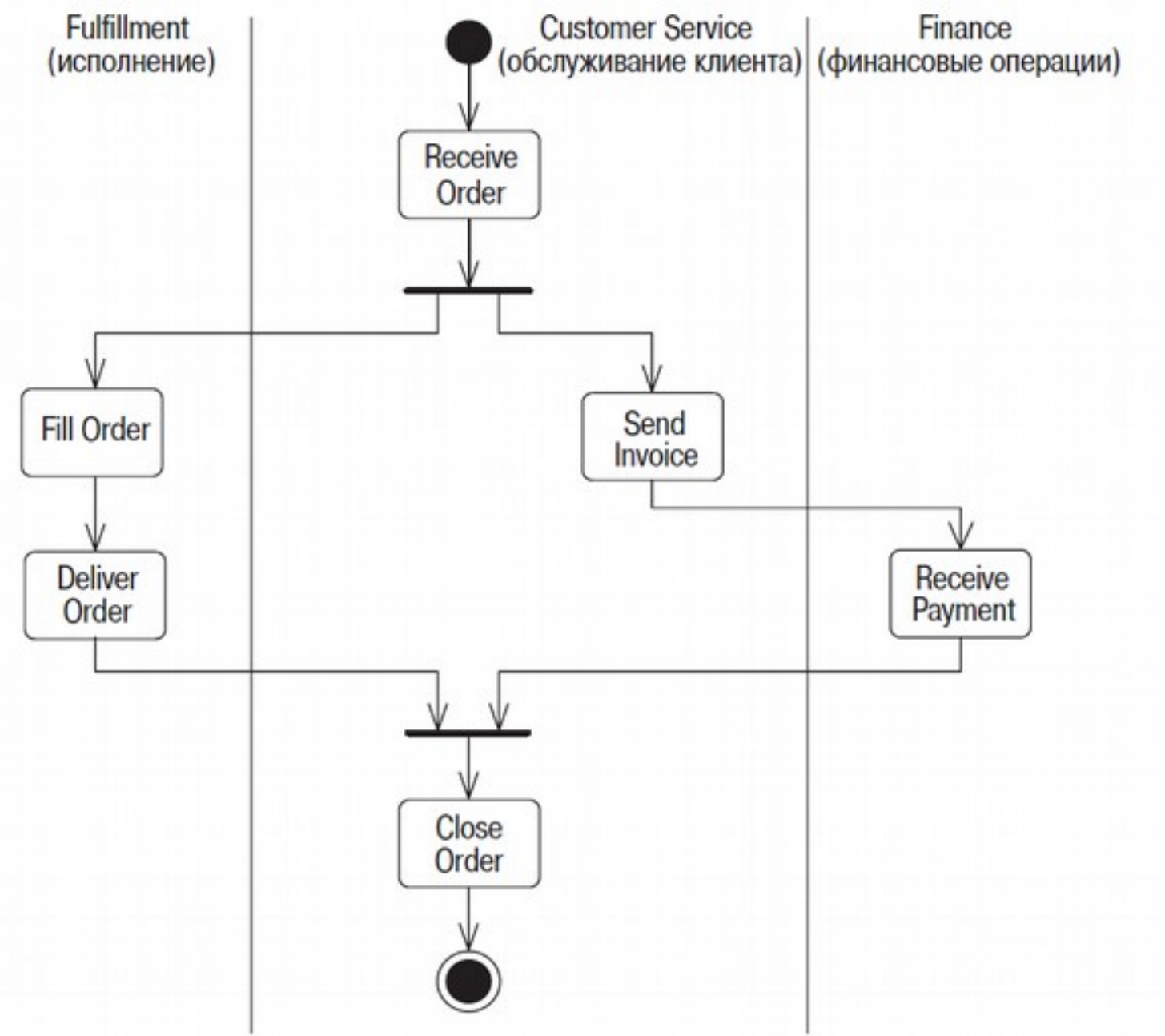
Простая диаграмма деятельности



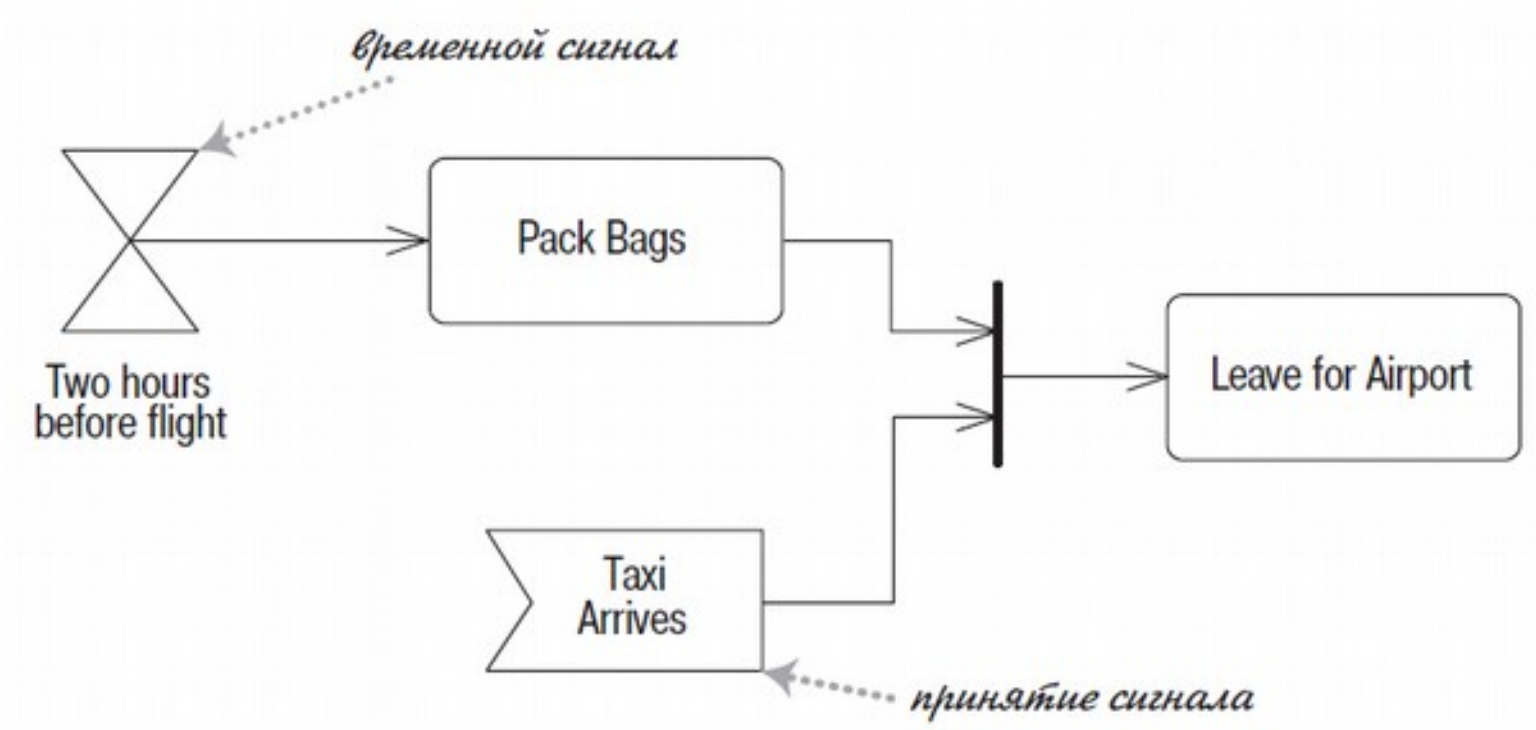
Декомпозиция операции



Разбиение диаграммы деятельности на разделы



Сигналы в диаграмме деятельности



Отправка и прием сигналов