РЕАГИРУЮЩИЕ СИСТЕМЫ И СТЕЙТЧАРТЫ В МОДЕЛИРУЮЩЕЙ СИСТЕМЕ ANYLOGIC

Реагирующая система (reactive system) это система, постоянно ожидающая внешних или внутренних событий и реагирующая на них. Реагирующие системы являются типичными системами дискретно-событийного типа: события происходят в дискретные моменты времени, реакция системы на события состоит в изменении переменных состояния этих систем и формально такая реакция является мгновенной.

При описании реагирующих систем оказалось удобным использовать стейтчарты (или карты состояний). Стейтчарты представляют собой графический язык диаграмм. Язык стейтчартов в настоящее время широко применяется для спецификации, моделирования и прототипирования протоколов коммуникации, систем управления в авиации, в научной и бытовой электронике.

Стейтчарты строятся из состояний и переходов между ними. Система может находиться в каждый момент времени только в одном состоянии. Переходы из состояния в состояние случаются, если происходит событие, связанное с этим переходом, и условие, связанное с переходом, выполнено. На диаграмме система из состояния *A* переходит в состояние *B*, если наступит событие *a* и при этом условие *P* будет выполнено. Событием может быть, например, истечение таймаута, переключение в истину предиката (условия), определенного на переменных модели и т. п. Графически состояния представляются прямоугольниками или овалами, а переходы дугами. Короткая стрелка-указатель, входящая в состояние *A*, говорит о том, что это состояние начальное: в начальный момент времени система будет находиться именно в этом состоянии. Очевидно, что у системы может быть ровно одно начальное состояние. С каждым переходом может быть связано некоторое действие - изменение переменных, посылка сигнала и т. п. С каждым состоянием также могут быть связаны действия. Одно действие выполняется в момент входа в это состояние, другое действие выполняется при выходе из состояния. На *Y1* и *Y2* условно обозначены действия, выполняемые при срабатывании соответствующих переходов.

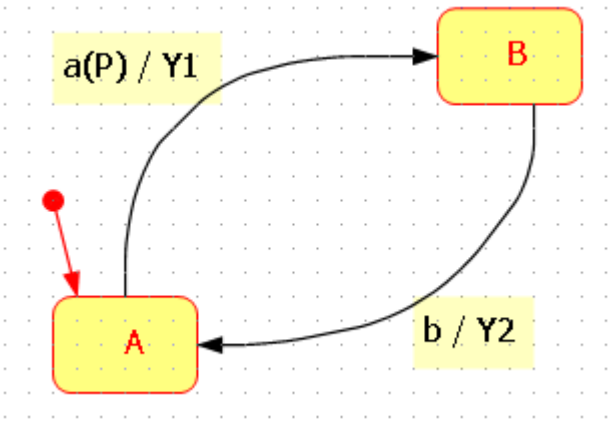


Рис. 6.3. Простой граф переходов

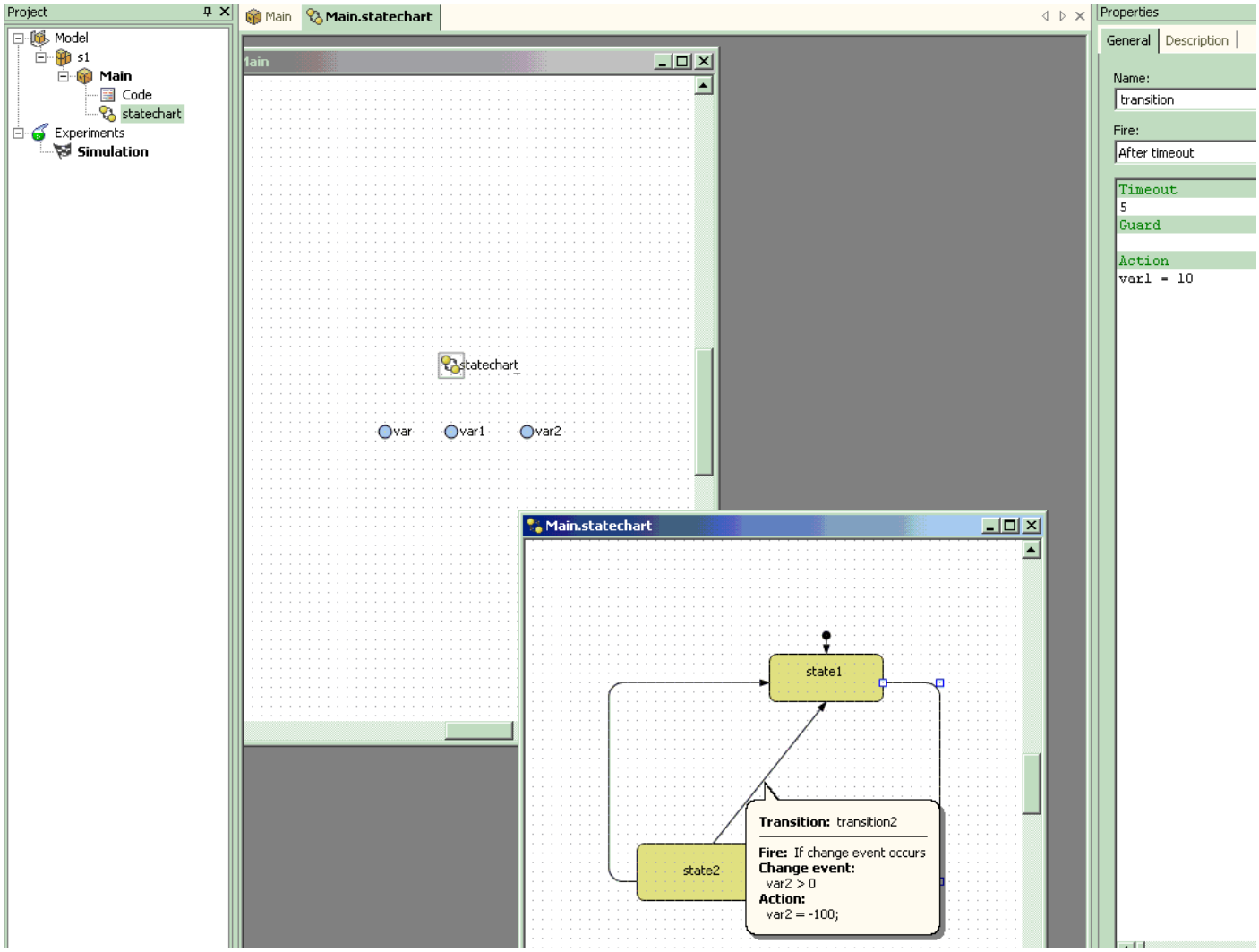
Модель s1.alp

var – показывает в каком состоянии находится модель (значения 1 и 2)

var1 – показывает какой переход сработал (1 и 2)

var2 > 0 – включает дополнительный переход

переходы происходят по таймаутам



В общем случае в стейтчартах можно использовать расширения этой простейшей модели переходов: иерархические состояния (гиперсостояния), исторические состояния, условные переходы и некоторые другие возможности. Рассмотрим их по порядку.

Иерархические состояния или ***гиперсостояния*** вводятся для того, чтобы объединить несколько состояний, имеющих одну и ту же реакцию на событие. На рис. 6.4 справа гиперсостояние *D* позволяет упростить граф переходов, представленный слева: переход в состояние *A* при наступлении события *b* происходит вне зависимости от того, в каком из состояний, *B* или *C* находилась система. Иными словами, два стейтчарта на рис. 6.4 эквивалентны

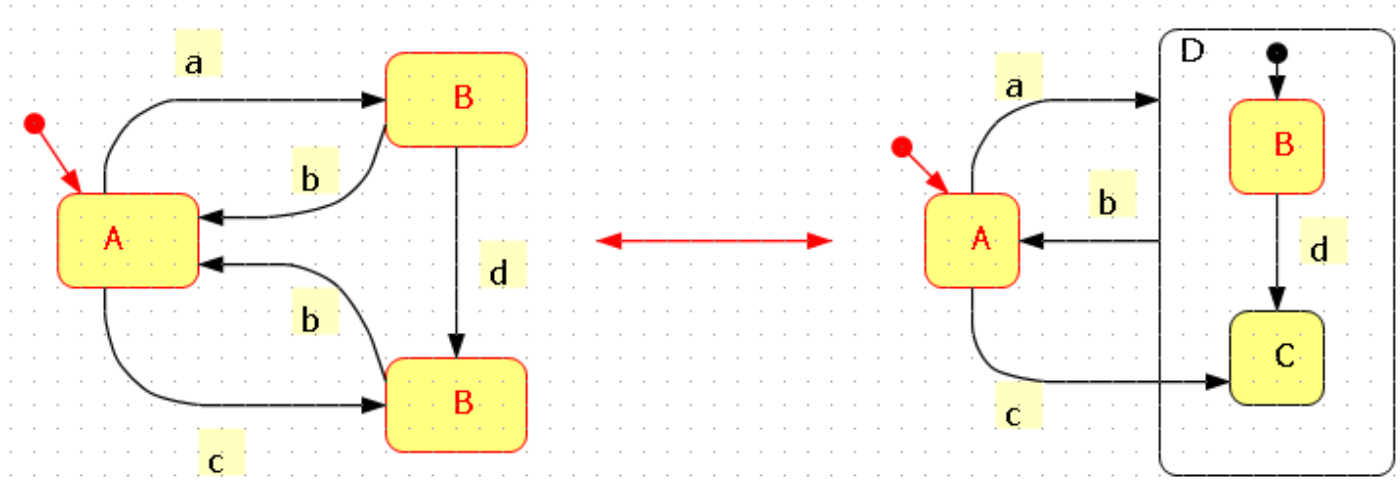


Рис. 6.4. Использование гиперсостояния упрощает граф переходов

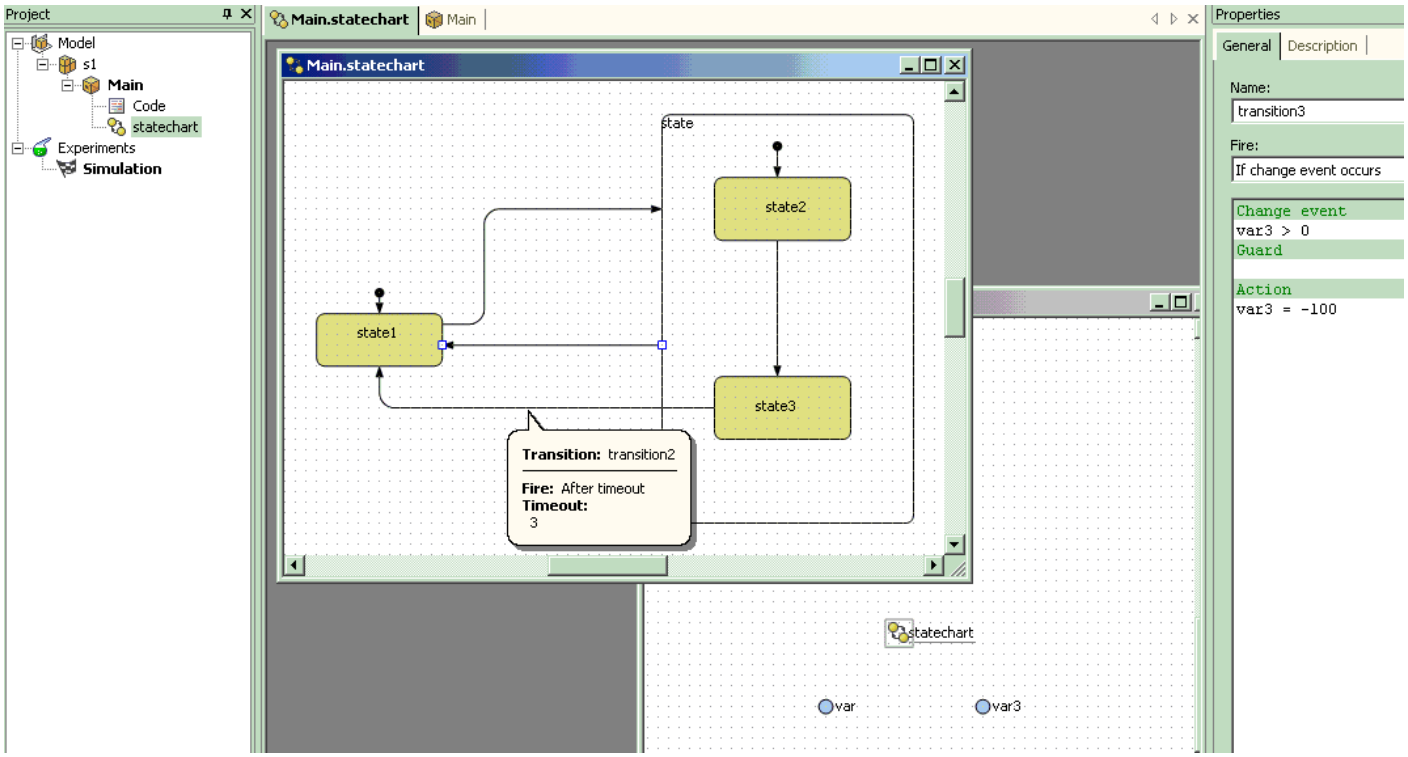
Каждое гиперсостояние требует, чтобы точно одно из включенных в него состояний было помечено как начальное. Это позволяет трактовать переход из состояния *A* при наступлении события *a* в гиперсостояние *D* как переход из *A* в элементарное состояние *B*. На рис. 6.4 начальным состоянием системы является состояние *A*, а состояние *B* является начальным только для множества состояний *{B, C},* входящих в гиперсостояние *D*.

Модель

var – показывает в каком состоянии находится модель (значения 1, 2, 3)

var3 > 0 – включает выход из гиперсостояния

переходы происходят по таймаутам



Историческое состояние хранит то состояние внутри данного гиперсостояния, в котором система находилась последний раз.

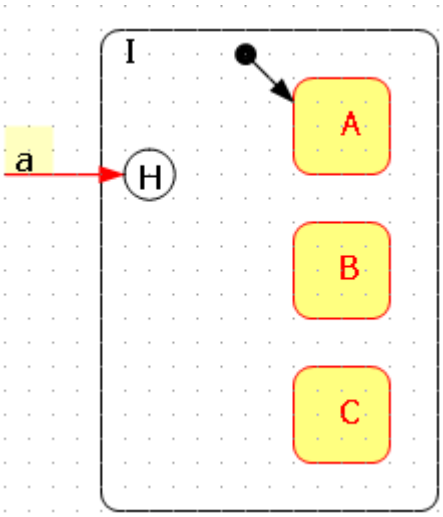


Рис. 6.5. Историческое состояние

На рис. 6.5 при наступлении события *a* система вернется в то состояние из множества состояний *{A, B, C},* в котором она была последний раз (независимо от того, какими переходами связаны эти состояния). Исторические состояния удобны, например, для описания продолжения функционирования системы после прерываний.

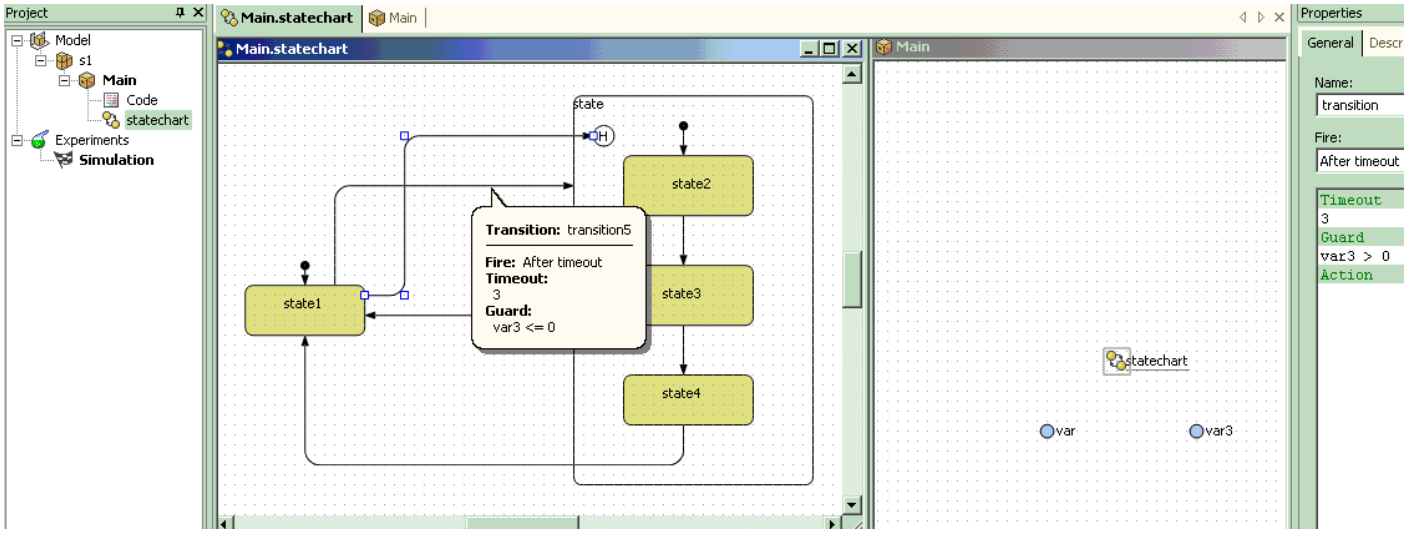
Модель s2.alp

var – показывает в каком состоянии находится модель (значения 1, 2, 3, 4)

var3 > 0 – включает выход из гиперсостояния

переходы происходят по таймаутам

!!! history Action var3 = 0



Условные состояния позволяют отложить проверку логического условия. Такая отложенная проверка удобна, например, в том случае, если определить дальнейшие действия системы можно только после реакции на событие.

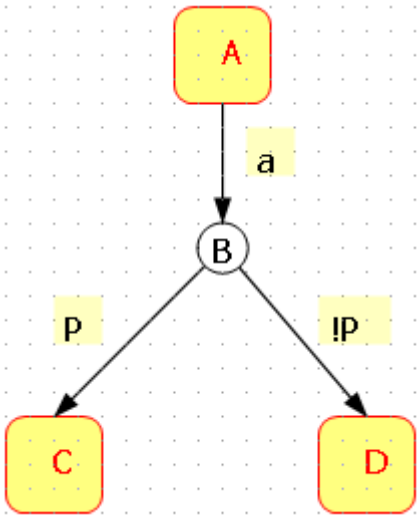
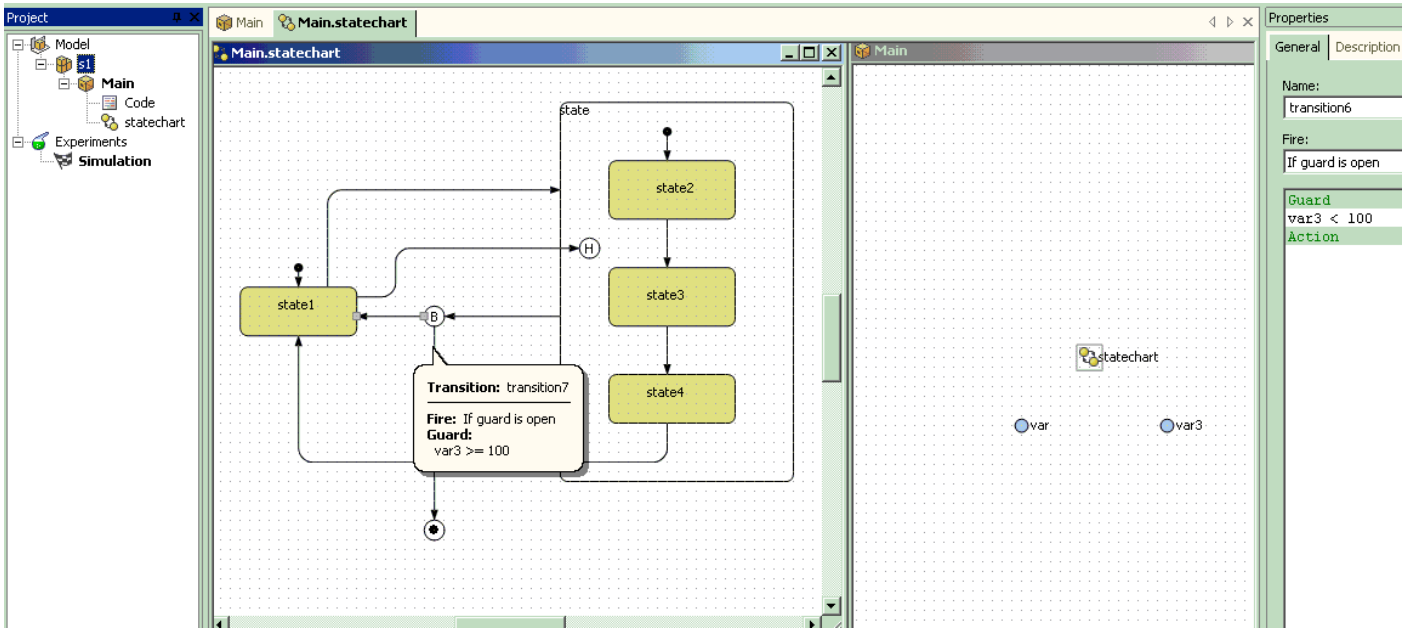


Рис. 6.6. Условное состояние

Например, пусть событием является приход сообщения, а реакция на него зависит от содержимого этого сообщения. На рис. 6.6 представлен фрагмент стейтчарта, описывающего эту ситуацию. По приходе сообщения (событие *a*) если система находилась в состоянии *A*, то она перейдет в состояние

C только в том случае, если условие *P* выполнено. Если условие *P* не выполнено, система перейдет в состояние *D*. В этом примере удобно использовать именно условное состояние, в котором система не задерживается, а после приема сообщения мгновенно определяет, в какое состояние перейти.

Модель s3.alp на основе модели №2



В качестве примера использования всех этих расширений рассмотрим спецификацию процесса доступа к среде протокола IEEE 802.12 обмена сообщениями в высокоскоростной локальной сети 100VG-AnyLAN (рис. 6.7). В сети работает множество станций, и в каждой из них активизирован свой процесс доступа к сети. Все процессы идентичны.

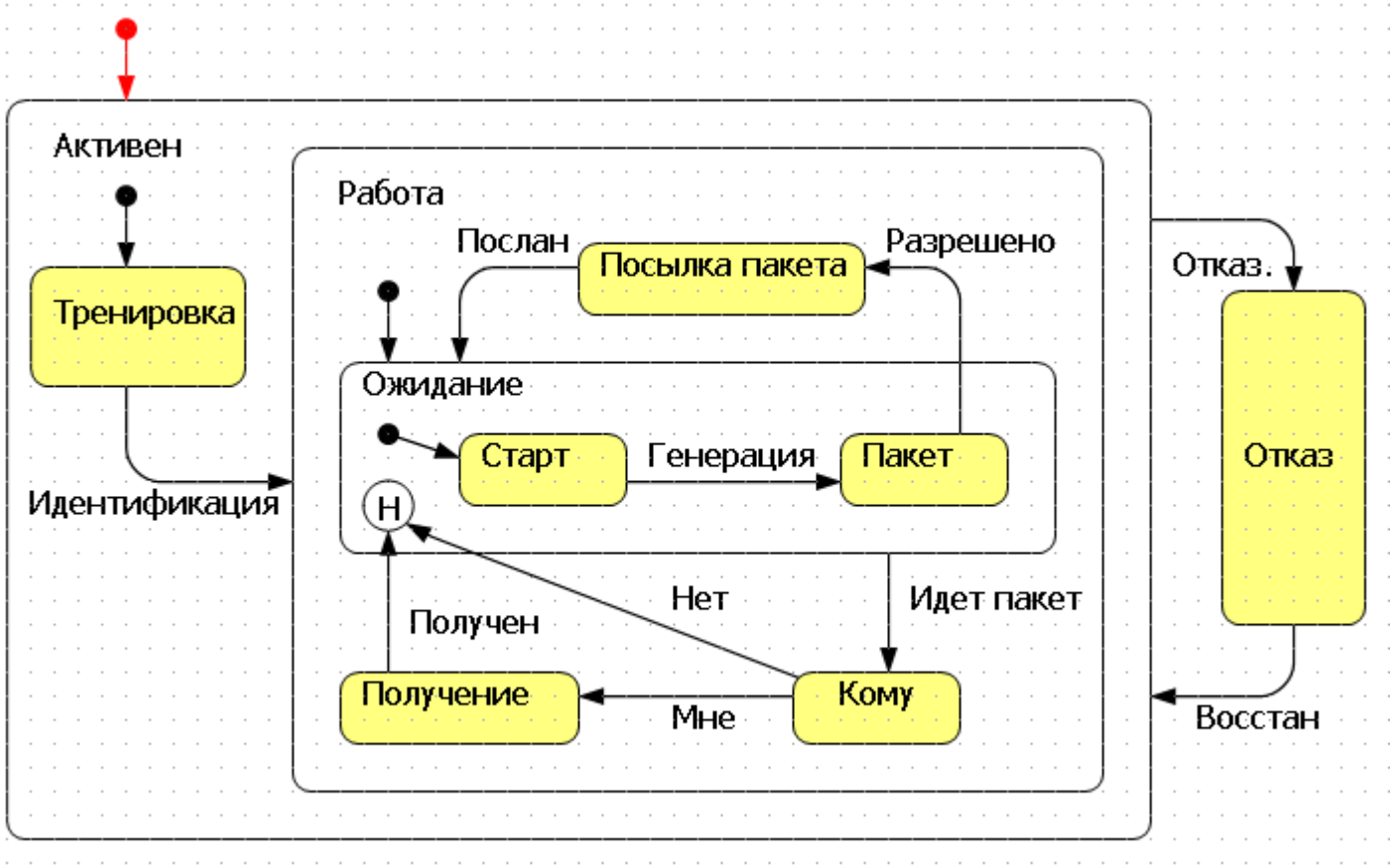


Рис. 6.7. Стейтчарт процесса доступа к среде протокола IEEE 802.12

Каждый процесс начинает свою работу в элементарном состоянии *Тренировка* гиперсостояния *Активен*. В состоянии *Тренировка* выполняются операции по идентификации данной рабочей станции, проверке канала и верхнего уровня стека протоколов. Если идентификация выполнилась без ошибок, процесс переходит в состояние *Старт,* в котором ожидает очередной пакет, сгенерированный пользователем **данной** рабочей станции для передачи. Когда пакет для передачи сгенерирован, процесс переходит в состояние *Пакет*, посылая верхнему уровню сигнал о наличии пакета. В этом состоянии процесс ждет от верхнего уровня разрешения послать пакет в канал, и когда разрешение получено, процесс переходит в состояние *Посылка пакета*. По завершении пересылки процесс возвращается в состояние *Старт*. В любом из двух состояний (*Старт* и *Пакет*) гиперсостояния *Ожидание* процесс может быть прерван приходом сигнала *Идет пакет* от другой рабочей станции. Этот сигнал извещает каждую станцию локальной сети о том, что в сети начал передаваться пакет, адресату необходимо подготовиться к его приему. Поскольку имя адресата находится в заголовке пакета, пакет начинают принимать все процессы. После приема пакета (или после обнаружения, что пакет чужой после приема заголовка) процесс возвращается в то состояние, включенное в гиперсостояние *Ожидание*, из которого он был прерван сигналом *Идет пакет*. В любом из обсуждавшихся ранее состояний процесс может быть прерван сигналом *Отказ*, который заставляет процесс перейти в состояние *Отказ*. После восстановления процесс входит в нормальную работу через состояние *Тренировка*.

Этот пример показывает, что стейтчарты позволяют наглядно и экономно выразить с помощью графической нотации весьма сложное поведение.