Практическая работа № 5
Создание диаграммы состояний

Цель работы: получить навыки построения диаграмм состояний.

Задание: разработать диаграмму состояний для одного из ранее разработанных классов или прецедентов.

Содержание отчета: диаграмма состояний и описание состояний в виде таблицы:

Состояние	Описание состояния

Пример выполнения работы

Диаграммы состояний применяются, как правило, для моделирования поведения классов, прецедентов или системы в целом.

Составим диаграмму состояний для класса *Order (Заказ)*, поскольку в нашей модели он наиболее часто будет менять свое состояние. Заказ может находится в нескольких состояния:

- при создании заказа он переходит в состояние *Инициализация,* в котором выполняются некоторые предварительные действия;
- после завершения инициализации заказ переходит в состояние *Открыт*, в котором к заказу добавляются новые пункты. Выход из этого состояния возможен или в случае отмены заказа, или в случае заполненния всех необходимых пунктов заказа;
- если заполнены все необходимые пункты заказа, то он переходит в состояние *Закрыт*, в котором происходит выписка счета. Выход из этого состояния произойдет только после того, как счет будет выписан;
- если заказ отменен, то из состояния *Открыт* он переходит в состояние *Отменен*. При выходе из этого состояния происходит удаление всех пунктов заказа.

Диаграмма состояний для класса *Order* представлена на рис.1.

Первым состоянием на диаграмме состояний является начальное состояние. При выполнении события "заказ создан" заказ переходит в состояние *Инициализация.* При входе в это состояние выполняется входное действие "Сохранить дату заказа". Основное действие, которое будет выполняться в течении всего времени, пока заказ будет находится в этом состоянии, это "Внести информацию о клиенте". Переход из этого состояния в состояние *Открыт* произойдет только при выполнении сторожевого условия "инициализация завершена".

В состоянии *Открыт* имеется выходное действие и переход в себя. Переход в себя означает, что событие инициирует переход, происходит выход из текущего состояния, выполняется некоторое действие, после чего происходит возврат в исходное состояние. Поскольку при переходе в себя происходит выход из состояния и повторный вход в него же, то выполняется действие, ассоциированное с переходом, и, кроме того, действие при входе в состояние. В состоянии *Открыт* к заказу добавляются новые пункты, причем их можно добавить только в том случае, если есть незаполненные пункты. Для показа

этого мы использовали переход в себя "Добавление пункта заказа" со сторожевым условием "заполнены не все пункты заказа". Выход из этого состояния состоится в двух случаях - или когда выполнится сторожевое условие "заполнены все позиции заказа" (при этом заказ перейдет в состояние *Закрыт*), или когда наступит событие "заказ отменен" (при этом заказ перейдет в состояние *Отменен*). При выходе из состояния выполнится действие выхода "* OrderItem.Create()" (создание пункта заказа). Символ "*" указывает на то, что это действие выполнится много раз (по числу добавленных пунктов в заказ).
В состоянии *Закрыт* присутствует только внутреннее действие - "Выписать счет". В это состояние заказ переходит из состояния *Открыт* только при выполнении сторожевого условия "заполнены все позиции заказа". Выход из этого состояния и переход в конечное произойдет при наступлении события "счет выписан".

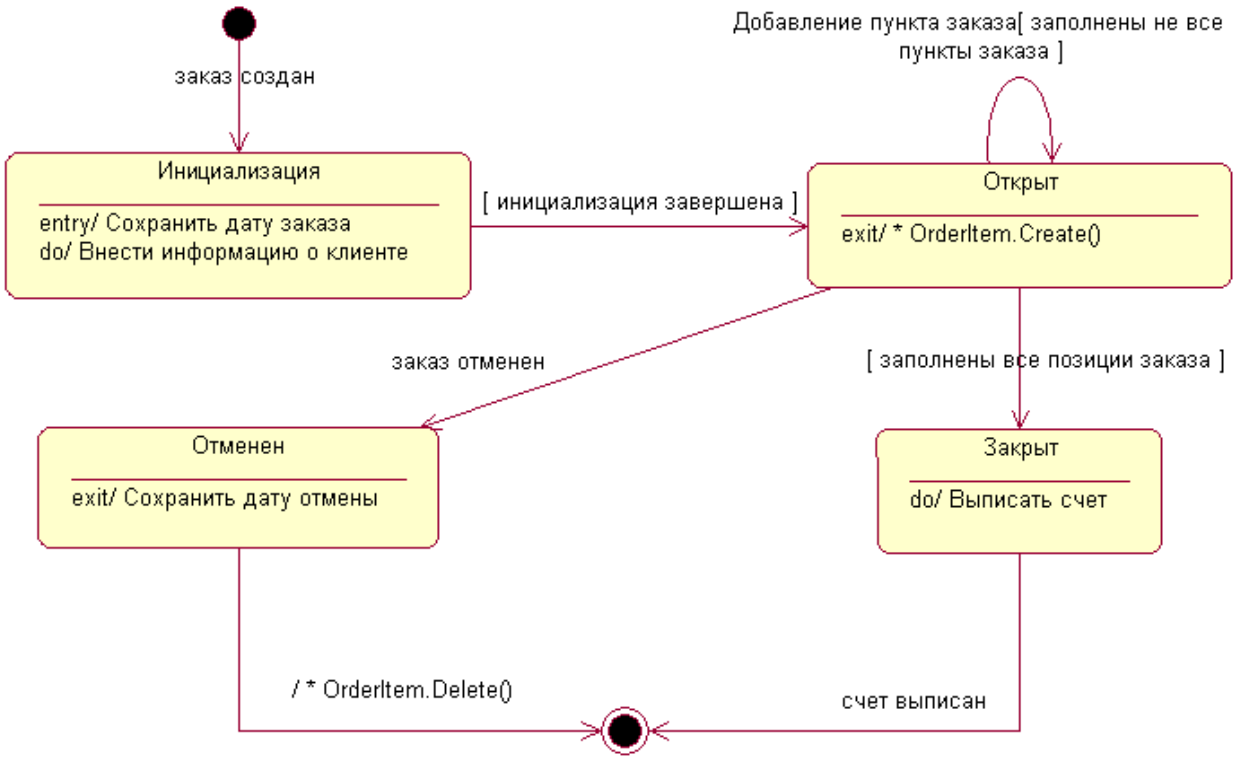

Рисунок 1. Диаграмма состояний для класса Order

В состояние *Отменен* заказ переходит из состояния *Открыт* при наступлении события "заказ отменен". При выходе из него выполняется действие выхода "Сохранить дату отмены". При переходе из этого состояния в конечное выполняется действие "* OderItem.Delete()" (удаление пункта заказа). Здесь также стоит "*", поскольку это действие будет выполняться много раз.