|  |  |
| --- | --- |
|  | МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РЕСПУБЛИКИ ТАТАРСТАН **Государственное автономное профессиональное образовательное учреждение** **«ЗЕЛЕНОДОЛЬСКИЙ МЕХАНИЧЕСКИЙ КОЛЛЕДЖ»** (ГАПОУ «ЗМК») |

**09.02.03 «Программирование в компьютерных системах»**

# **МДК 03.01. ТЕХНОЛОГИЯ РАЗРАБОТКИ ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ**

**Отчет о практических работах**

**Исполнитель**: Лялин Даниил Дмитриевич

**Группа**: 205

**Преподаватель:** Алемасов Евгений Павлович

**Дата сдачи** 14.11.2023 **Оценка** \_\_\_\_\_\_\_\_\_\_\_\_

**Подпись преподавателя**  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**ЗЕЛЕНОДОЛЬСК – 2023**

# **Работа №5. Создание диаграмм состояний**

**Цель** работы – получение навыков построения Statechart Diagram.

**Диаграмма состояний (State diagram)** — это один из видов диаграмм UML, используемых в разработке программного обеспечения, чтобы визуализировать и моделировать поведение объекта или системы в различных состояниях. Она позволяет описать все возможные состояния объекта, а также переходы между ними в ответ на определенные события.

Диаграмма позволяет визуально представить, как объекты изменяются в ответ на внешние условия. Переходы обозначают изменение состояния и указывают, при каких условиях происходит переход между состояниями. События являются внешними сигналами или действиями, которые вызывают переходы между состояниями.

На диаграмме состояний UML состояния обычно представляются в виде прямоугольников с названиями, а переходы — стрелками. Дополнительные аннотации могут использоваться для указания условий переходов или действий, выполняемых в определенных состояниях.

Диаграмма состояний по существу является графом специального вида. Вершинами этого графа являются состояния и некоторые другие типы элементов автомата (псевдосостояния), которые изображаются соответствующими графическими символами. Дуги графа служат для обозначения переходов из состояния в состояние. Диаграммы состояний могут быть вложены друг в друга, образуя вложенные диаграммы более детального представления отдельных элементов модели.

**Состояние (state)** – ситуация в жизненном цикле объекта, во время которой он удовлетворяет некоторому условию, выполняет определенную деятельность или ожидает какого-либо события. Состояние объекта определяется значениями некоторых его атрибутов и присутствием или отсутствием связей с другими объектами.

Секция «Список внутренних действий» содержит перечень внутренних действий или деятельностей, которые выполняются в процессе нахождения моделируемого элемента в данном состоянии. Каждое из действий записывается в виде отдельной строки и имеет следующий формат:

# <метка-действия / выражение-действия>

Метка действия указывает на обстоятельства или условия, при которых будет выполняться деятельность, определенная выражением действия:

* **entry** – эта метка указывает на действие, специфицированное следующим за ней выражением действия, которое выполняется в момент входа в данное состояние (входное действие);
* **exit** – эта метка указывает на действие, специфицированное следующим за ней выражением действия, которое выполняется в момент выхода из данного состояния (выходное действие);
* **do** – эта метка специфицирует выполняющуюся деятельность («do activity»), которая выполняется в течение всего времени, пока объект находится в данном состоянии, или до тех пор, пока не закончится вычисление, специфицированное следующим за ней выражением действия. В последнем случае при завершении события генерируется соответствующий результат;
* **include** – эта метка используется для обращения к подавтомату, при этом следующее за ней выражение действия содержит имя этого подавтомата.

**Переход (Transition)** — это элемент диаграммы состояний, который представляет собой переход объекта из одного состояния в другое. Он определяет событие или условие, которое вызывает изменение состояния, и указывает, какой переход должен быть выполнен при наступлении этого события или условия.

**Простой переход (simple transition)** представляет собой отношение между двумя последовательными состояниями, которое указывает на факт смены одного состояния другим. До срабатывания перехода объект находится в предыдущем от него состоянии, называемым исходным состоянием, или в источнике (не путать с начальным состоянием – это разные понятия), а после

его срабатывания объект находится в последующем от него состоянии (целевом состоянии).

Переход осуществляется при наступлении некоторого события: окончания выполнения деятельности (doactivity), получении объектом сообщения или приемом сигнала. На переходе указывается имя события. Кроме того, на переходе могут указываться действия, производимые объектом в ответ на внешние события при переходе из одного состояния в другое. Срабатывание перехода может зависеть не только от наступления некоторого события, но и от выполнения определенного условия, называемого сторожевым условием. Объект перейдет из одного состояния в другое в том случае, если произошло указанное событие и защитное условие приняло значение «истина».

На диаграмме состояний переход изображается сплошной линией со стрелкой, которая направлена в целевое состояние. Каждый переход может помечен строкой текста, которая имеет следующий общий формат:

**<сигнатура события>[<защитное условие>]<выражение действия> Событие** представляет собой спецификацию некоторого факта, имеющего место в пространстве и во времени. Про события говорят, что они

«происходят», при этом отдельные события должны быть упорядочены во времени. После наступления некоторого события нельзя уже вернуться к предыдущим событиям, если такая возможность не предусмотрена явно в модели.

**Защитное условие (guard condition)**, если оно есть, всегда записывается в прямых скобках после события и представляет собой некоторое булевское выражение.

Если защитное условие принимает значение «истина», то соответствующий переход может сработать, в результате чего объект перейдет в целевое состояние. Если же защитное условие принимает значение «ложь», то переход не может сработать, и при отсутствии других переходов объект не может перейти в целевое состояние по этому переходу. Однако вычисление

истинности защитное условия происходит только после возникновения ассоциированного с ним события, инициирующего соответствующий переход. **Составное состояние (composite state)** – такое сложное состояние, которое состоит из других вложенных в него состояний. Последние будут выступать по отношению к первому как подсостояния (substate).

**Последовательные подсостояния (sequential substates)** используются для моделирования такого поведения объекта, во время которого в каждый момент времени объект может находиться в одном и только одном подсостояний. Поведение объекта в этом случае представляет собой последовательную смену подсостояний, начиная от начального и заканчивая конечным подсостояниями. Хотя объект продолжает находиться в составном состоянии, введение в рассмотрение последовательных подсостояний позволяет учесть более тонкие логические аспекты его внутреннего поведения.

**Параллельные подсостояния (concurrent substates)** позволяют специфицировать два и более подавтомата, которые могут выполняться параллельно внутри составного события. Каждый из подавтоматов занимает некоторую область (регион) внутри составного состояния, которая отделяется от остальных горизонтальной пунктирной линией. Если на диаграмме состояний имеется составное состояние с вложенными параллельными подсостояниями, то объект может одновременно находиться в каждом из этих подсостояний

Поведение параллельных подавтоматов независимо друг от друга, что позволяет реализовать многозадачность в программных системах. Однако в отдельных случаях может возникнуть необходимость учесть в модели синхронизацию наступления отдельных событий. Для этой цели в языке UML имеется специальное псевдосостояние, которое называется **синхронизирующим состоянием**.

**Синхронизирующее состояние (synch state)** обозначается небольшой окружностью, внутри которой помещен символ звездочки "\*". Оно используется совместно с переходом-соединением или переходом-ветвлением

для того, чтобы явно указать события в других подавтоматах, оказывающие непосредственное влияние на поведение данного подавтомата.

# Пример диаграммы деятельности:

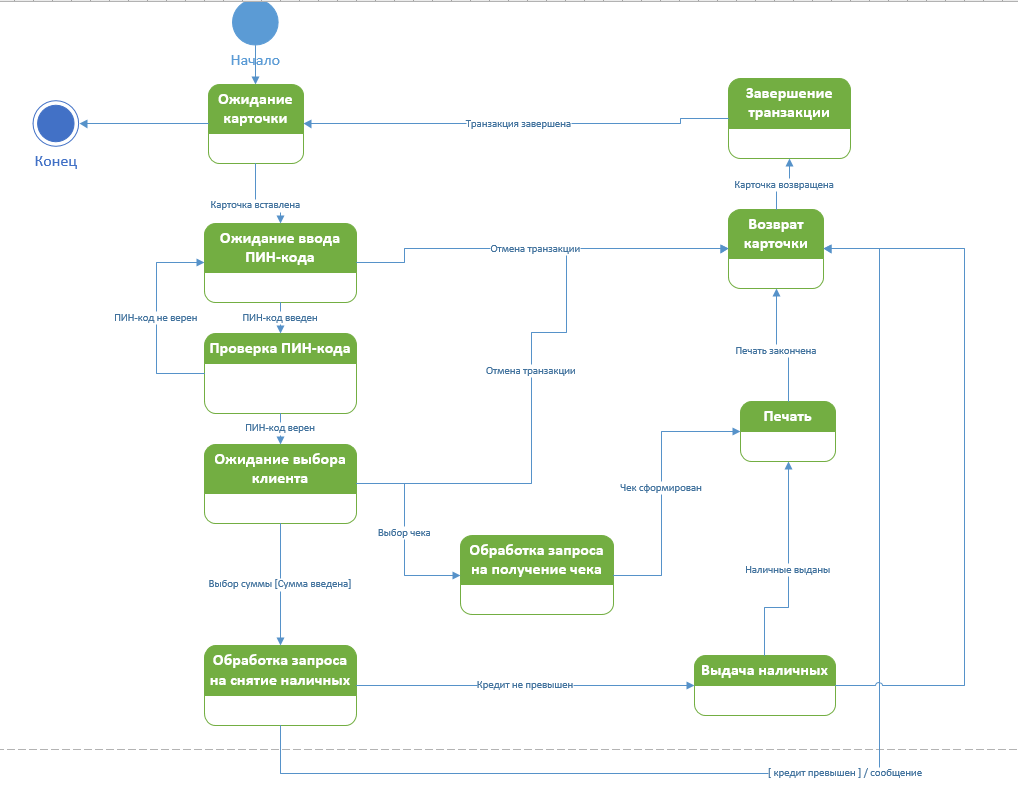


Рисунок 1- Диаграмма состояний

**Задания на самостоятельную работу:**

1. Постройте диаграмму состояний для проверки кредитного рейтинга клиента. После запуска программы система запрашивает рейтинг клиента. Если рейтинг окажется положительным, система должна вернуть кредитный рейтинг клиента, после чего работа системы завершается. Если рейтинг отрицательный, то из состояния ожидания результатов система переходит в состояние «Отклонить заказ», затем через 10 секунд завершает работу.

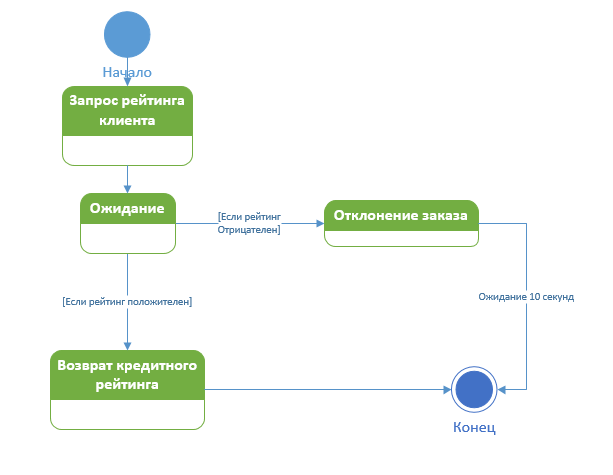


Рисунок - Диаграмма состояний

1. Постройте диаграмму состояний процесса функционирования телефонного аппарата.

Диаграмма состояний представляет единственный автомат с одним составным состоянием. Вне этого составного состояния имеется только

одно состояние «Ожидание», которое характеризует исправный и подключенный к телефонной сети телефонный аппарат. Переход в следующее состояние происходит при поднятии телефонной трубки. Переход с атомарным действием «подать тоновый сигнал» переводит аппарат в составное состояние, а точнее – в начальное его подсостояние. Далее телефонный аппарат будет находиться в состоянии «тоновый сигнал». При этом будет непрерывно издавать этот сигнал до тех пор, пока либо не произойдет событие-триггер «набор цифры (п)», либо не истечет 15 секунд с момента поднятия трубки. В первом случае аппарат перейдет в состояние «набор номера», а во втором – в состояние

«истечение времени ожидания». Последняя ситуация может быть результатом сомнений по поводу «звонить – не звонить?», следствием чего могут стать гудки в трубке. При этом нам ничего не остается делать, как опустить ее на рычаг.

При наборе номера выполняется событие-триггер "набор цифры (п) со сторожевым условием «номер неполный». Это означает, что если набранный телефонный номер не содержит необходимого количества цифр, то нам следует продолжать набор очередной цифры, оставаясь в состоянии «набор номера».

Если же набранный номер полный, то можно перейти в состояние

«неверный номер» или «соединение». В случае неверного номера (сторожевое условие «неверный» истинно) ничего не остается, как покинуть составное состояние, опустив трубку на рычаг. Если же номер верный, то происходит соединение по этому номеру.

Однако в результате соединения может оказаться, что аппарат абонента занят (переход в состояние «занято») либо свободен (переход в состояние «звонок у абонента»). В первом случае можно повторить дозвон, предварительно опустив трубку на рычаг (выход из составного состояния). Во втором случае происходит проверка сторожевого условия «разговор доступен». Если оно истинно, что соответствует

снятию трубки абонентом, начинается телефонный разговор. В противном случае (это условие не выполняется, т. е. оно ложно) телефон абонента будет продолжать звонить, извещая нас об отсутствии последнего либо о невозможности по какой-либо причине вести разговор по телефону. При этом нам ничего не остается, как опустить трубку на рычаг.

Если же разговор состоялся, то после слов прощания и выполнения сторожевого условия «подтверждение» на окончание разговора мы снова опускаем трубку. При этом телефонный аппарат переходит в состояние «ожидание», в котором может находиться неопределенно долго.

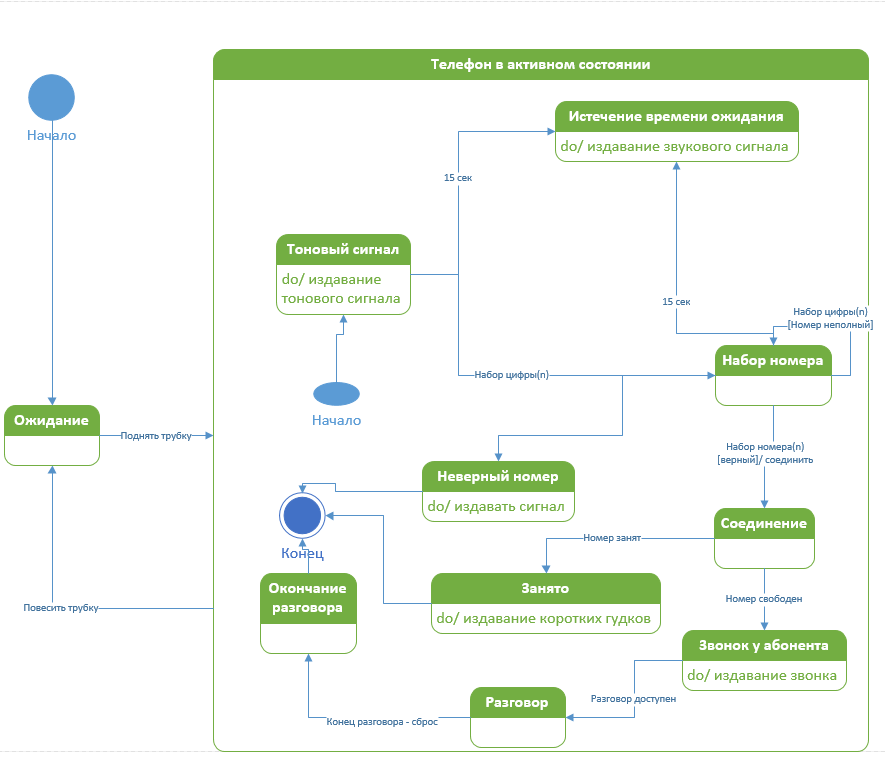


Рисунок - Диаграмма состояний