**1.3 Неформальная постановка задачи**

Основные принципы и концепции объектной модели в программировании развивались в процессе эволюции множества разных объектных и объектно-ориентированных языков.

Основной идеей объектного подхода является объединение данных и производимых над этими данными операций в одно концептуально замкнутое понятие — класс. Данные класса не должны изменяться извне его, доступ к данным стоит осуществлять только через функции-члены (методы класса).

Класс - группировка объектов, которые имеют те же самые свойства, общее поведение и одинаковые отношения.

Класс определяет характеристики объектов. Однако значения могут быть назначены только после того, как объект будет создан. Только в этом случае появляется фактический образец объекта.

Объект - понятие или вещь с определенными границами, которая является уместной в данной проблеме, с которой мы имеем дело. Объекты позволяют выполнить две цели:

1.Они помогают понимать окружающий мир.

2. Они обеспечивают практическую реализацию для создаваемого нами приложения.

Программа, написанная на объектном языке, представляет собой совокупность объектов, каждый из которых принадлежит к определенному абстрактному типу данных (классу) и имеет интерфейс в виде набора методов для взаимодействия друг с другом (посылки сообщений).

Все объектно-ориентированные языки разбивают программу на части, многократно используемые и расширяемые. Программы представляют собой объекты многократного использования. Эти объекты могут группироваться различными способами, формируя новые программы.

Объектно-ориентированное проектирование — это методология проектирования, соединяющая в себе процесс объектной декомпозиции и приемы представления логической и физической, а также статической и динамической моделей проектируемой системы.

Объектно-ориентированное проектирование дает возможность создавать расширяемые системы (extensible systems). Расширяемость (extensibility) означает, что существующую систему можно заставить работать с новыми компонентами без внесения в нее каких-либо изменений. . Это уникальное и очень мощное понятие, потому что реальная задача постоянно требует изменений. Это уникальное и очень мощное понятие, потому что реальная задача постоянно требует изменений.

Таким образом, чтобы составить корректную оценку объектно-ориентированной модели программного обеспечения, следует обратиться к описанию объекта исследования, или, иными словами, к постановке задачи, в которой описаны необходимые для реализации функции и устройство банкомата, для которого проектируется информационная система.

Электромеханический замок, позволяет контролировать доступ в помещение, посредством контрольного кода, и пароля.

В замке предусмотрена возможность голосовой связи с человеком, по ту сторону двери.

С внешней стороны имеется возможность изменить пароль (При открытой двери, нажав кнопку «КОНТРОЛЬ») и контрольный код (При открытой двери, нажав кнопку «ВЫЗОВ»).

Use Case

Модель вариантов использования представляет собой модель взаимодействия пользователей для решения проблем и задач.

Модель вариантов использования описывает цели пользователей, взаимодействие между пользователями и системой и требуемое поведение системы для удовлетворения этих целей.

Модель вариантов использования состоит из ряда модельных элементов (или графических примитивов). К наиболее важным из них относятся варианты использования [use cases], действующие лица или акторы [actors] и отношения (связи) между ними [relationships].

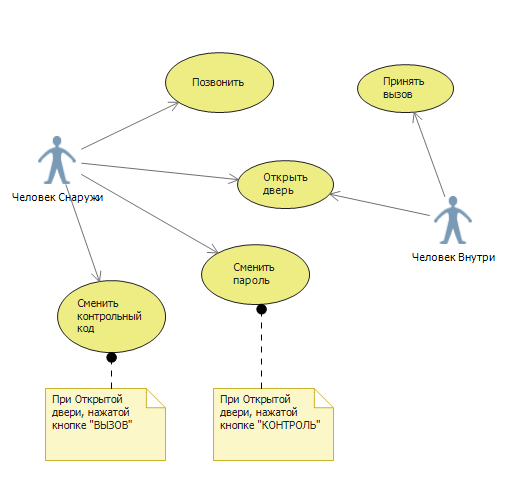
Для того, чтобы описать то, как может использоваться данная проектируемая система, необходимо выявить действующие лица – то есть, все объекты, которые будут взаимодействовать с системой. В данном случае, выделяется один объект и рассматривается его поведение в системе.

Действующее лицо:

Пользователь – Любой человек, пользующееся программным обеспечением будильника.

Варианты использования:

Исходя из потребностей пользователя, выделяются следующие варианты использования:



**2.Требования к окружению**

Не слишком шумное помещение, где звук звонящего будильника будет услышан.

**2.1 Требования к аппаратному обеспечению**

Компьютер, построенный на aрхитектуре :

Intel x86-x64 (Windows 7, Windows 8)

Минимальное количество ОЗУ: 512 мб.

Минимальное количество ПЗУ: 1Гб.

**3 . Архитектура системы (Общие требования)**

Требуется разработать модель программногообеспечения встроенного микропроцессора для кодового замка, регулирующего доступ в помещение.

Кодовый замок состоит из панели с кнопками (цифры "0"..."9", кнопка "Вызов", кнопка "Контроль"), цифрового дисплея, электромеханического замка, звонка. Панель с кнопками устанавливается с наружной стороны двери, замок устанавливается с внутренней стороны двери, звонок устанавливается внутри охраняемого помещения. В обычном состоянии замок закрыт. Доступ в помещение осуществляется после набора кода доступа, состоящего из четырех цифр. Во время набора кода введенные цифры отображаются на дисплея. Если код набран правильно, то замок открывается на некоторое время, после чего дверь снова закрывается. Содержимое дисплея очищается.

Кнопка "Вызов" используется для подачи звукового сигнала внутри помещения. Кнопка "Контроль" используется для смены кодов. Смена кода доступа осуществляется следующим образом. При открытой двери нужно набрать код контроля, состоящий из четырех цифр, и новый код доступа. Для смены кода контроля нужно при открытой двери и нажатой кнопке "Вызов" набрать код контроля, после чего -новый код контроля.

Библиотека подпрограмм (классов)

