Введение  
  
 В наше время существует огромное количество программ для обновления софта, каждое программное обеспечение имеет своё отличие от другого, среди которых значимым является количество обновлений и способ их установки. Также обновление «софта» имеет важную роль в жизни операционной системы для нормальной работы ОС. В следствии чего и нужно правильно выбрать приложение для обновления программного обеспечения.

Именно поэтому я и решил разработать «софт», который будет обновлять установленные программы на компьютере пользователя.  
 В качестве типа программного продукта я выбрал прикладное (desktop) ПО, так как на компьютерах пользователей выходит много новых программ, в особенности под ОС Windows, одной из самых популярных ОС, пользователем этой операционной системы наиболее часто требуется дополнительные приложения для обновлений ПО.

Также важно заметить что найти удобный и бесплатный продукт для обновления очень сложно.

Для использование моей программы можно просто установить её, для того что бы обновить ваш софт, через интернет. Программа достаточно простая в использование и ей могут пользоваться люди всех возрастных категорий и рабочих специальностей.  
 Открыв программу, пользователь сможет увидеть список установленных программ на его компьютере и доступны обновления для них, и выбрать которые ему нужны для обновления или выбрать обновить все программы.  
 Моя программа будет на уровне таких программ для обновления ПО как: SUMo, Secunia PSI, Update Checker, UpdateStar и т.д. Но она будет распространяться по лицензии Open Source.   
 Данная программа будет создана для курсовой работы, и расширенная для дипломного проекта, а также будет увеличен функционал, и осуществлено продвижение ПО.

1 Сбор и анализ требований  
  
 Так как разрабатываемая программа не является уникальным продуктом, был   
 Проведен анализ требований рынка схожих продуктов. Оценивались их идеи, функционал, дизайн следующих программ:  
SUMo, Secunia PSI, Update Checker, UpdateStar.

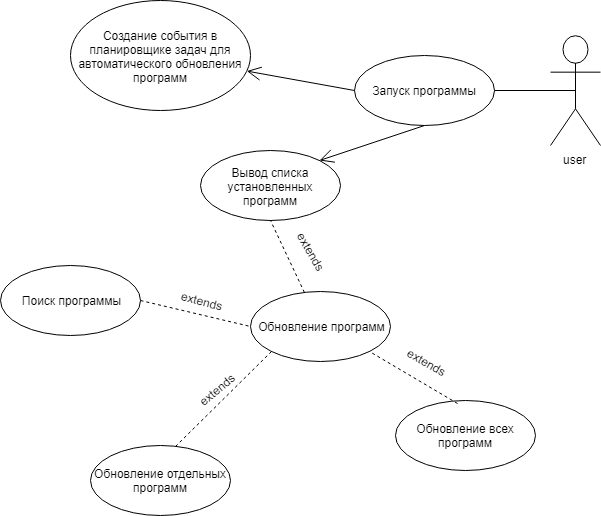
Выбор данных программ был обусловлен их высокой релевантностью в поисковых системах оп запросу: “ Программа для обновления софта ”.

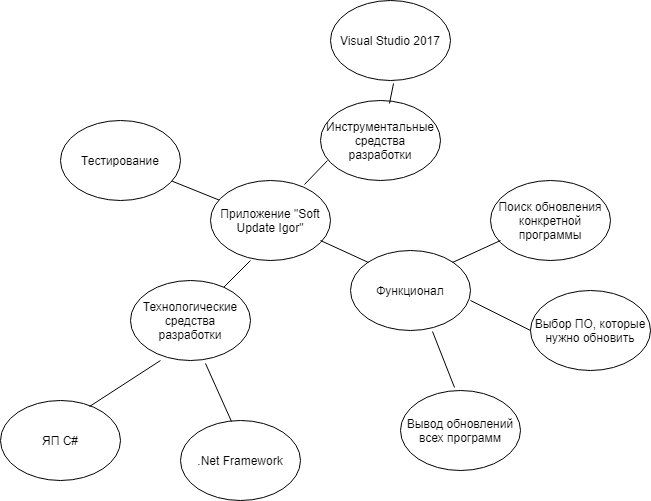
У данной выборки были установлены общие характеристики функциональности:  
   
- Обнаружение необходимых обновлений и установка;  
  
Из не функциональных особенностей было выявлено:  
  
- Они бесплатные;  
- Работают под ОС Windows 10,7, Vista, XP;  
- У них схожий интерфейс;  
  
Формирование требований к программному продукту

Исходя из полученных в ходе анализа данных были установлены следующие требования к разрабатываемому ПО.

1.1 Функциональные требования:  
  
- Программа позволяет решить проблему обновления программ;  
- Вывод доступных обновлений;  
- Выбор для обновления ПО;  
- Создание в планировщике заданий автоматическое обновление;  
  
1.2 Не функциональные требования:  
  
- ПО является бесплатным;  
- Разработан для ОС Windows 10,7;  
- Стандартный интерфейс для программ;  
- Поиск программы по названию для обновления;

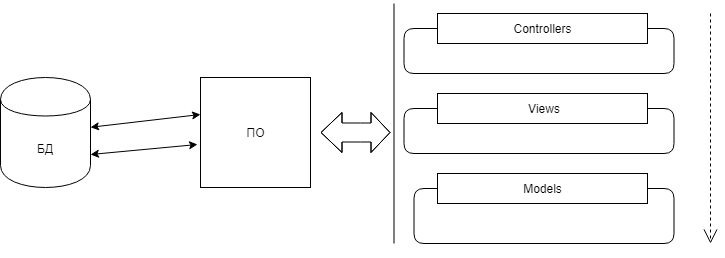
2 Проектирование и разработка архитектуры  
   
  
2.1 Построение диаграммы use-case  
  
Разработка use-case диаграммы, для визуального отображения взаимодействия пользователя с приложением.

Для разработки use-case диаграммы был выбран сервис draw.io, так как это удобный сервис для создание данной схемы.  
  
  
  
  
Рисунок 1 – Диаграмма use-case

2.2 mind-map  
  
После анализа требований к ПО была разработана Mind-map диаграмма.  
Mind-map нужен для построения диаграмм связей, для того чтобы понять процесса мышления, наиболее похожий на то, как рождаются и развиваются мысли и идеи в нашем мозгу.   
Для построения диаграммы связей был выбран сервис draw.io, поскольку сервис является бесплатным и простым в использовании.  
  
  
  
  
  
  
Рисунок 2 – Диаграмма mind-map

2.3 Прототип приложения   
Далее был разработан прототип окна приложения.

Для прототипирования был использован сервис ninjamock.com. Данный сервис был выбран потому, что понятный и удобный интерфейс, а также бесплатный.  
Мокап нужен чтобы показать как интерфейс приложения будет выглядеть в реальной жизни.  
  


Рисунок 3 – Мокап главного окна приложения  
  
2.4 Архитектура программного продукта  
  
Для реализации программного продукта была выбрана многоуровневая архитектура и MVC – Model View Controller.  
Выбор данных архитектур был основан идеей чтобы отделить модели данных, их отображение и операции взаимодействия с пользователем. Выигрыш от использования такой архитектуры заключается в том, что она позволяет упорядочить код, распределив его по уровням, каждый из которых определяет сферу ответственности. Это минимизирует взаимозависимость программных компонент, что в свою очередь облегчает их последующую модификацию. Доработка и развитие такой системы становится проще. Каждая часть соотносится с определённым уровнем, для каждого уровня заданы выполняемые им функции.  
 В данном программном продукте используется 2 уровня: программное приложение и база данных.  
  
  
  
Рисунок 4 – Взаимодействие уровней в приложении

3 Технологические и инструментальные средства разработки программного продукта  
  
3.1 Технические средства разработки программного продукта.  
Основным техническим средством для разработки ПО был выбран C# .NET Framework.   
  
- .Net Framework;   
- ЯП С# 7.0;   
  
В качестве языка программирования был выбран С#.   
   
- Относительно не сложный язык по сравнению с С/C++/Java;  
- Обширная документация и развитое комьюнити;  
- Возможность использования множества сторонних и встроенных библиотек, сервисов;   
- большой спектр возможностей реализации проектов;  
- реализация проекта будет под ОС Windows;  
- высокая скорость разработки;  
- гибкость языка;  
  
В качестве фреймворка был выбран .Net Framework:  
  
- совместимость различных служб написанных на разных языках;  
- разнообразие технологий;  
- мощная библиотека классов;  
- кроссплатформенность;  
  
3.2 Инструментальные средства для разработки программного продукта   
  
 Для разработки приложения использовались:  
  
- Visual Studio 2017;  
  
Собственно почему была выбрана IDE Microsoft Visual Studio.  
  
Так как разработка ПО будет проводится под ОС Windows 10, самая популярная для это ОС IDE Microsoft Visual Studio.  
  
Данный выбор может быть обоснован следующим:   
  
- Она является бесплатной;  
- много функциональная для разработки программ;  
- большой выбор служб(сервисов);  
- кроссплатформенная;   
- большой список документации;  
- выбор множества языков программирования;   
  
  
  
3.3 Методология разработки

Для разработки ПО была выбрана XP (ExtremeProgramming) экстремальная разработка. При выборе методологии был проведен анализ при котором сравнивались типы методологий:

* RUP (Rational Unified Process) — рациональный.
* Agile — общая методология гибкой разработки.
* Crystal Clear — подход с уравниванием разработчиков в коллективе.
* Spiral — спиральный метод.
* DSDM (Dynamic Systems Development Model) — динамическая модель.
* FDD (Feature Driven Development) — методология, рассматривающая будущие изменения.
* JAD (Joint Application Development) — ориентированный на пользователя подход.
* RAD (Rapid Application Development) — модель быстрой разработки.
* Scrum — концепция работы в условиях сорванных сроков и идеологического кризиса.
* XP (Extreme Programming) — экстремальная разработка в динамической среде.
* LD (Lean Development) — метод, предполагающий бережное отношение ко всем участникам процесса.

Было принято решение использовать экстремальную разработку т. к. -Экстремальность метода связана с высокой степенью риска решения, обусловленного поверхностностью анализа и жестким временным графиком. Реализуется минимальный набор главных функций системы на первой и каждой последующей итерации, функциональность расширяется на каждой итерации.   
Разработка проекта командой разработчиков состоит из нескольких, или одного человека а также простота решения задач.

4 Тестирование программного продукта

В течение всей разработки программного продукта проводились тестирования, благодаря чему были выявлены и исправлены ошибки и недочёты приложения. В качестве технологии тестирования было выбрано тестирование серого ящика, позволяющее провести более тщательное тестирование и повысить процент покрытия кода, и тестировать программный продукт по частям.

4.1 Тест план

1. ID

* Тестирование приложения “SoftUpdate SC”.

1. Задачи для проведения тестирования:

* Разработать тест план
* Написать тест кейс
* Создать баг репорт

1. Виды тестирования:

* Тестирование серого ящика.

1. Объекты тестирования:

* Главное окно приложения;
* Отображение списка программ;
* Работоспособность кнопок;  
  5. Что и как будет тестироваться?  
  - Корректное отображение информации;   
  - Поиск программы по названию;  
  - Корректная работа функционала;

4.2 Test Case

В таблицах 1, 2 и 3 описаны основные test-case программного продукта.

Таблица 1 – Запуск приложения.

|  |  |  |
| --- | --- | --- |
| Название | Отображение главного окна | |
| Функции | Корректное отображение текста | |
| Действие | Ожидаемый результат: | Пройден |
| Предусловие | | |
| Зайти в приложение | Приложение запустилось. | |
| Шаги теста | | |
| Проверить приложение на запуск | Приложение работает коректно | |
| Постусловие | | |
| Выйти из приложения | Выход из приложения произошёл успешно | |
|  |  | |

Таблица 2 – Действия кнопки “Вывод данных”

|  |  |  |
| --- | --- | --- |
| Название | Вывод данных | |
| Функции | Проверка на вывод данных в datagrid WPF | |
| Действие | Ожидаемый результат: | Пройден |
| Предусловие | | |
| Зайти в приложение | Приложение запустилось | |
| Шаги теста | | |
| Проверить вывод данных, кол-во выводимых данных. | Данных о программах не повторяются. | |
| Постусловие | | |
| Выйти из приложения | Выход из приложения произошёл успешно | |

Таблица 3 – Проверка на поиск данных кнопка “Найти”

|  |  |  |
| --- | --- | --- |
| Название | Проверка работы кнопки поиска | |
| Функции | Проверка на правильную работу навигационной панели | |
| Действие | Ожидаемый результат | Провален |
| Статус ошибки | S4 | Незначительный |
| Предусловие | | |
| Зайти в приложение | Приложение запустилось | |
| Шаги теста | | |
| В текстовом поле ввести текст для поиска. | Выводит только полную строку названия программы. | |
| Проверить работу ссылочных кнопок | Все кнопки работают корректно | |
| Постусловие | | |
| Выйти из приложения | Выход из приложения произошёл успешно | |

4.3 Баг репорт

После тестирования по выявленной ошибке был сделан баг-репорт который отображён в таблице 4.

Таблица 4 – баг репорт некорректный вывод по запросу

|  |  |
| --- | --- |
| Краткое описание | Отображение текста. |
| Проект | Soft Update CH. |
| Компонент приложения | Отображение списка установленных программ. |
| Номер версии | 0.01. |
| Важность | S4 Незначительная |
| Статус | Новая |
| Автор | Игорь Злобин |
| Назначен разработчика | Игоря Злобина |
| Шаги воспроизведения | Запускаем приложение. |
| Вводим тест в текстовом поле. |  |
| Фактический результат | Должно было выводится по любой строке похожие на название программ. |
| Ожидаемый результат | Корректное отображение текста |

При тестировании было покрыто более 90% кода, была выявлена и исправлена незначительная ошибка и проверен программный продукт на наличие ошибок и недочётов.

5 Документация

В файле PC.cpp есть класс PC, который содержит информацию о установленных программах. В дальнейшем данные об установленном ПО, версия программы будут браться из BD(базы данных) и подгружаться в приложение с помощью облачного хранилища.

Функции класса:

Конструктор класса принимает 3 параметра:

1. String information – содержит в себе основную информацию о породе собаки, происхождении и истории.
2. String temperament – содержит в себе информацию о темпераменте той или иной породе собаки.
3. String care – содержит в себе информацию об уходе за определённой собакой

Заключение

При разработке курсового проекта были проанализированы конкуренты, вследствие чего были сформированы основные требования к программному продукту и основные направления развития. Были созданы use-case, mind-map диаграммы, создан прототип программы а также выбрана архитектура программного продукта. До начала разработки была выбрана методология. Исходя из требований, был выбраны технологические и инструментальные средства разработки. После разработки основной части проекта, было проведено тестирование с помощью серого ящика, и составлены test-case и выявлены bug-report благодаря которым были выявлены и устранены ошибки, который были допущены при разработке.

Данный проект был создан для защиты курсовой работы, в дальнейшем проект не будет развиваться.