Дисциплина «Проектный практикум»

Практическое задание № 2.

Определение функциональных/нефункциональных требований и планирование разработки программного продукта

2.1 Функциональные требования

2.1) Тезариус:

* Преобразователь – программа, которая на основе полученного кода, написанного на одном языке, формирует соответствующий код (тот, что выполняет то же самое, что и входной), на другом языке.
* ЯП- язык программирования.
* Пользователь – программист, работающий с программой и желающий получить преобразованный код.
* Формат xml-файла – специально составленный формат с определенной структурой, который передается на следующие модули системы.
* Входной код – код, который вставляет пользователь в виде файла или просто вставкой в соответствующее окно системы.
* Выходной/Преобразованный код – код, полученный в результате преобразования.

2.2) Функциональные требования:

|  |  |  |  |
| --- | --- | --- | --- |
| № | Описание функционального требования | Приоритет | [Tangible Software Solutions](https://www.tangiblesoftwaresolutions.com/index.html) |
| 1. | Система должна преобразовывать полученный код в xml-файл, соответствующего формата. | M | - |
| 1.1. | Система должна осуществлять над входным кодом лексический анализ. | M | + |
| 1.2. | Система должна осуществлять над входным кодом, прошедшим лексический анализ синтаксический анализ. | M | + |
| 1.3. | Система должна осуществлять над входным кодом, прошедшим синтаксический анализ семантический анализ. | M | + |
| 1.4. | Система должна на выходе получить xml-файл, соответствующего формата, решающий те же задачи, что и входной код. | M | - |
| 2. | Система должна определять принадлежность входного ЯП к указанному пользователем. | C | - |
| 3. | Система должна определять соответствие формата входного файла с допустимым и оповещать Пользователя при несоответствии. | М | + |
| 4. | Система должна позволять Пользователю выбрать входной ЯП и выходной либо граф. | М | - |

Следует отметить, что если пользователь использует функцию выбора файла с компьютера, то данный модуль системы передает выходной xml-файл дальше и не выводит результат в поле вывода. Файл с выходным кодом формируется другой частью системы и сберегается на компьютере пользователя в соответствующей папке программного продукта.

2.2 Нефункциональные требования

2.2.1) Внешние интерфейсы:

1. Окно взаимодействия с пользователем содержит только самые важные и функциональные элементы системы.
2. Некоторые поля являются неактивными пока не будет указанно обратное (Например, по умолчанию активно поле ввода кода, а неактивным является поле выбора входного файла. Управление осуществляется за счет radio button).
3. Шрифты и стили должны быть едиными для всей системы. Размер шрифта необходимо сделать читабельным, окно – масштабируемым.
4. Кнопка справки должна находиться в верхнем правом углу, а сама справка поделена на разделы, описывающие отдельные модули системы, такие как модуль редактирования и тестирования графа, модуль взаимодействия с диаграммами, модуль преобразования кода и др.
5. Формат входного файла с четко установленным форматом.
6. Реализация на ЯП Javа.

2.2.2) атрибуты качества:

- Производительность: Программа является не сетевой, по этой причине в один момент времени обслуживается только один пользователь. Время отклика варьируется в зависимости от выбранного действия:

* Выбор элементов управления – не более 1 с
* Определение соответствия кода с указанным ЯП – не более 5 с
* Преобразование кода может занимать от 5 с в зависимости от размера входных данных, количества строк и специфических элементов языка.

- Надежность: Так как в данном модуле нет никаких промежуточных этапов, связанных с пользователем, то создание резервных копий не является необходимым.

2.3 Планирование разработки

С помощью метода Use Case Points производится оценка длительности разработки системы. Данный метод подразделяется на 5 этапов. Первым этапом является оценка сложности актеров системы. Она производится с учетом взаимодействия актера с системой.

В табл.1 предоставлена оценка сложности актеров системы.

Таблица 1 - Оценка сложности актеров системы

|  |  |  |  |
| --- | --- | --- | --- |
| Актер | Сложность | Взаимодействие | Вес |
| Пользователь | Сложный | Взаимодействует с системой через графический интерфейс | 3 |

Рассчитываем нескорректированную оценку актеров(*UAW*):

*UAW = 1\*3=3*

Далее рассчитываем оценку вариантов использования(ВИ), что позволяет оценить масштабы системы –это второй этап данного метода.

Оценивается сложность с помощью метода транзакций.

В табл.2 предоставлена оценка сложности вариантов использования.

Таблица 2 - Оценка сложности вариантов использования.

|  |  |  |  |
| --- | --- | --- | --- |
| ВИ | Количество транзакций | Сложность | Вес |
| Преобразование кода с одного ЯП в xml | 3 | Простой | 5 |

Рассчитываем нескорректированную оценку ВИ (*UUCW*):

*UUCW=5\*1=5*

Показатель UCP вычисляется по формуле:

*UCP= UAW+ UUCW=8*

Следующий этап состоит в оценке технических факторов и рассчитывается коэффициент для оценки сложности архитектуры системы. Для оценки используется шкала от 0 до 5, где 0 означает отсутствие влияния, 3 - среднее влияние, 5 – сильное влияние.

В табл. 3 производится оценка технических факторов.

Таблица 3 – Оценка технических факторов системы

|  |  |  |  |
| --- | --- | --- | --- |
| Фактор | Описание | Вес | Оценка |
| T1 | Распределенность системы | 2 | 0 |
| T2 | Время отклика | 1 | 5 |
| T3 | Эффективность конечного пользователя | 1 | 5 |
| T4 | Сложность обработки | 1 | 3 |
| T5 | Фокусировка на повторном использовании кода | 1 | 2 |
| T6 | Простота установки | 0,5 | 2 |
| T7 | Простота использования | 0,5 | 3 |
| T8 | Портативность | 2 | 1 |
| T9 | Простота изменений | 1 | 4 |
| T10 | Паралельные вычисления | 1 | 0 |
| T11 | Способы защиты | 1 | 0 |
| T12 | Доступ к третьей стороне | 1 | 0 |
| T13 | Потребность в специальном обучении | 1 | 4 |

*TFactor* рассчитывается как сумма произведения весов и оценок всех техничных факторов:

*TFactor = 2\*0 + 1\*5 + 1\*5 + 1\*3 + 1\*2 + 0,5\*2 + 0,5\*3 + 2\*1 + 1\*4 + 1\*0 + 1\*0 + 1\*0 + 1\*4 =27,5*

Оценка технического фактора рассчитывается по формуле:

TCF = 0.6 + (*TFactor* /100) = 0,875

Оценка внешних факторов системы представлена в табл. 4.

Таблица 4 - Оценка внешних факторов системы.

|  |  |  |  |
| --- | --- | --- | --- |
| Фактор | Описание | Вес | Оценка |
| F1 | Знакомство с процессомразработки | 1,5 | 2 |
| F2 | Опыт в подобных проектах | 0,5 | 1 |
| F3 | Опыт объектно-ориентированной разработки | 1 | 3 |
| F4 | Опытность ведущего аналитика | 0,5 | 2 |
| F5 | Мотивация | 1 | 5 |
| F6 | Стабильность требований | 2 | 3 |
| F7 | Частичная занятость работников | -1 | 3 |
| F8 | Сложность ЯП | -1 | 2 |

*EFactor* рассчитывается как сумма произведения весов и оценок всех внишних факторов:

*EFactor= 1,5\*2 + 0,5\*1 + 1\*3 + 0,5\*2 + 1\*5 + 2\*3 - 1\*3 – 1\*2 =13,5*

Оценка внешнего фактора рассчитывается по формуле:

ECF = 1.4 + (-0.03 x *EFactor*) = 0,995

Следующий этап состоит в подсчете результирующей оценки:

UCP = (UUCW + UAW) x TCF x ECF = 8 x 0,875x 0,995 = 6,965

Для определения длительности разработки *ТUCP* необходимо знать какому количеству рабочих часов отвечает один UCP. Для этогоподсчитывается количество факторов из множества F1 – F8, оценки которых по абсолютному значению превышают 3.

В данном случае их количество – 2, что отвечает 20 часам.

Расчет *ТUCP* :

*ТUCP* = 6,965 \* 20 = 139.3 час.

Опираясь на полученную продолжительность разработки построим план разработки проекта. Построение плана выполнения проекта начинается с определения работ, которые должны быть выполнены в проекте.

Группы работ, выделенные в проекте:

* Проектирование
* Реализация
* Обеспечение качества

На следующем этапе происходит распределения времени между работами, результаты которого представлены в таблице 5. Данные предоставлены с учетом общего времени выполнения проекта.  
Время было распределено с учетом стандартного распределения времени по WBS, с внесением определенных поправок. Например, на реализацию проекта было выделено 50% общего времени, что равно 69,65 рабочих часов.  
Таблица 5 - Распределение времени между работами

|  |  |  |
| --- | --- | --- |
| Работа | Доля времени, % | Время, час |
| Проектирование | 25 | 34,825 |
| Реализация | 50 | 69,65 |
| Обеспечение качества | 25 | 34,825 |

Работы, требующие декомпозиции делятся на подработы, между которыми распределяется время выполнения, выделенное на групповую работу. Также определяются зависимости между работами. Структурную декомпозицию работ предоставлено в таблице 6.  
Таблица 6 - Структурная декомпозиция работ

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| № | Название работы | WBS | Длительность, час | Предшественник |
| **1** | **Проектирование** | **IG.A** |  |  |
| 2 | Проектирование архитектуры | IG.A.A | 12 |  |
| 3 | Проектирование интерфейса пользователя | IG.A.B | 10.9 | 2 |
| 4 | Проектирование программных классов | IG.A.C | 12 | 2, 3 |
| **5** | **Реализация** | **IG.B** |  |  |
| 6 | ВИ Преобразование кода с одного ЯП в xml | IG.B.A | 69.65 | 3, 4 |
| **7** | **Тестирование** | **IG.C** |  |  |
| 8 | ВИ Преобразование кода с одного ЯП в xml | IG.С.A | 20 | 6 |
| 9 | Интеграционное тестирование | IG.С.B | 14.9 | 8 |

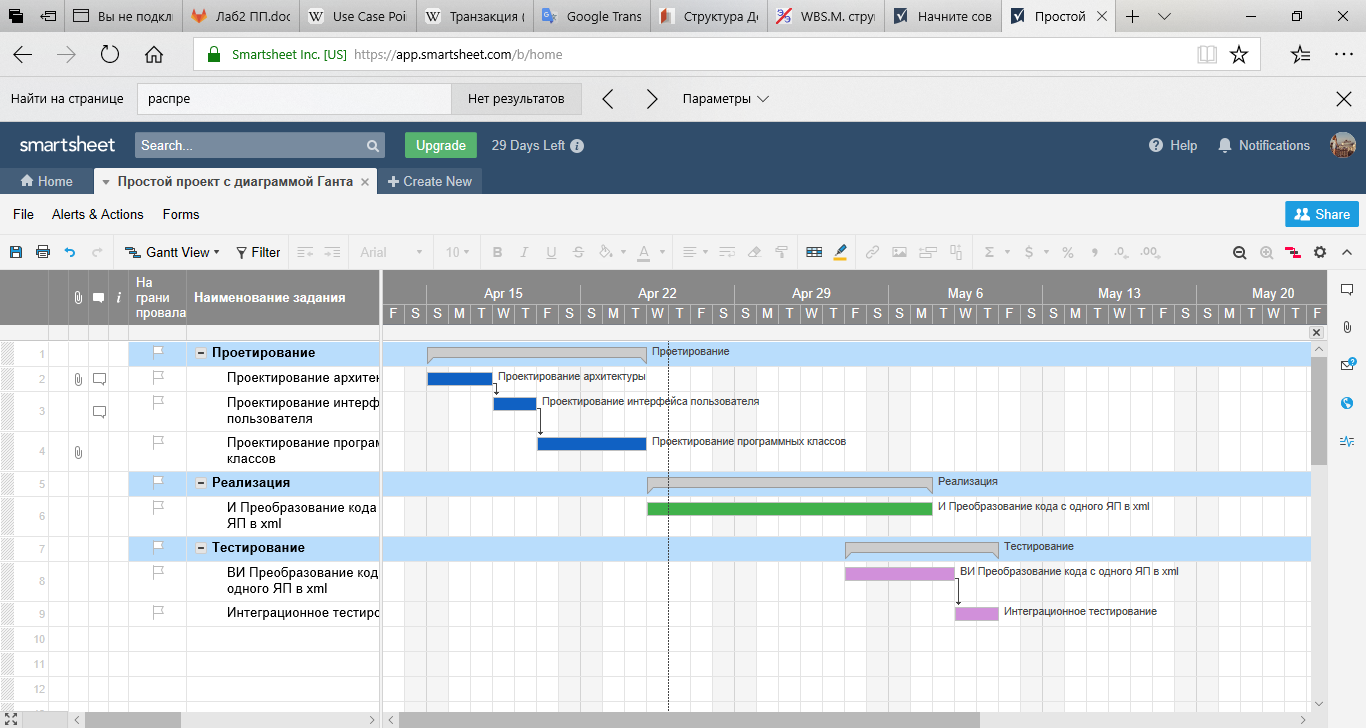


Рисунок 1 – Диаграмма Ганта