Министерство образования и науки Российской Федерации
Федеральное государственное бюджетное образовательное учреждение высшего профессионального образования Петрозаводский государственный университет Факультет математики и информационных технологий
Кафедра информатики и математического обеспечения

Отчет по дисциплине «Верификация ПО»

Выполнил:

Студент группы 22608 С.С. Ткач

_____

подпись

Преподаватель:

к.ф-м.н., доцент К. А. Кулаков

_____

подпись

Петрозаводск

2016

Описание тестируемого приложения

Лексическая многозначность — это фундаментальное свойство естественных языков: каждое слово может иметь более одного значения. Разрешение лексической многозначности (англ. word sense disambiguation или WSD) — задача определения смысла (значения) слова, которое принимается в определенном контексте. Данная задача является одной из важнейших задач обработки текстов. Для человека процесс устранения многозначности во многом является подсознательным и не представляет особых трудностей, но как вычислительная проблема он представляет собой сложнейшую задачу.

Система «NERP» представляет собой веб-приложение, которое использует метод построения лексических цепочек с целью разрешения лексической многозначности. В качестве машиночитаемого словаря система использует Русский Викисловарь. Пользователь вводит в форму целевое слово и фрагмент текста, в котором данное слово употребляется, затем система строит лексическую цепочку из слов введенного текста и исходя из данной цепочки предоставляет пользователю значение слова, соответствующее контексту.

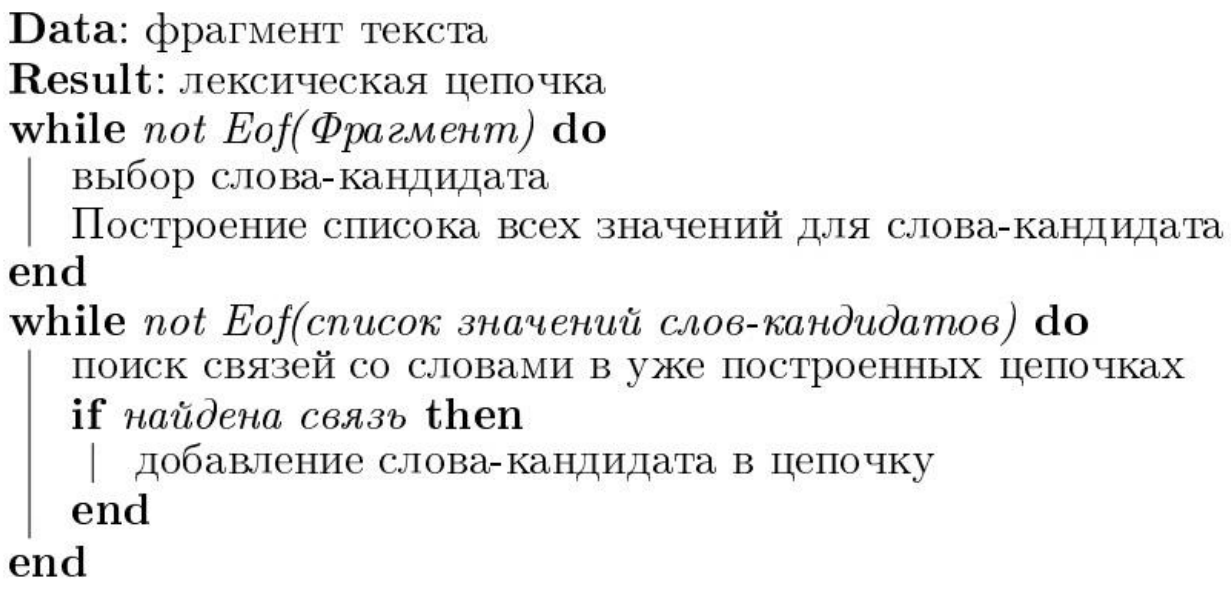

Рисунок 1. Алгоритм построения лексической цепочки

Приложение включает клиентскую и серверную части, для клиентской используются такие технологии, как HTML5 и CSS, для серверной – Apache 2.4, PHP 5, MySQL 5.6.

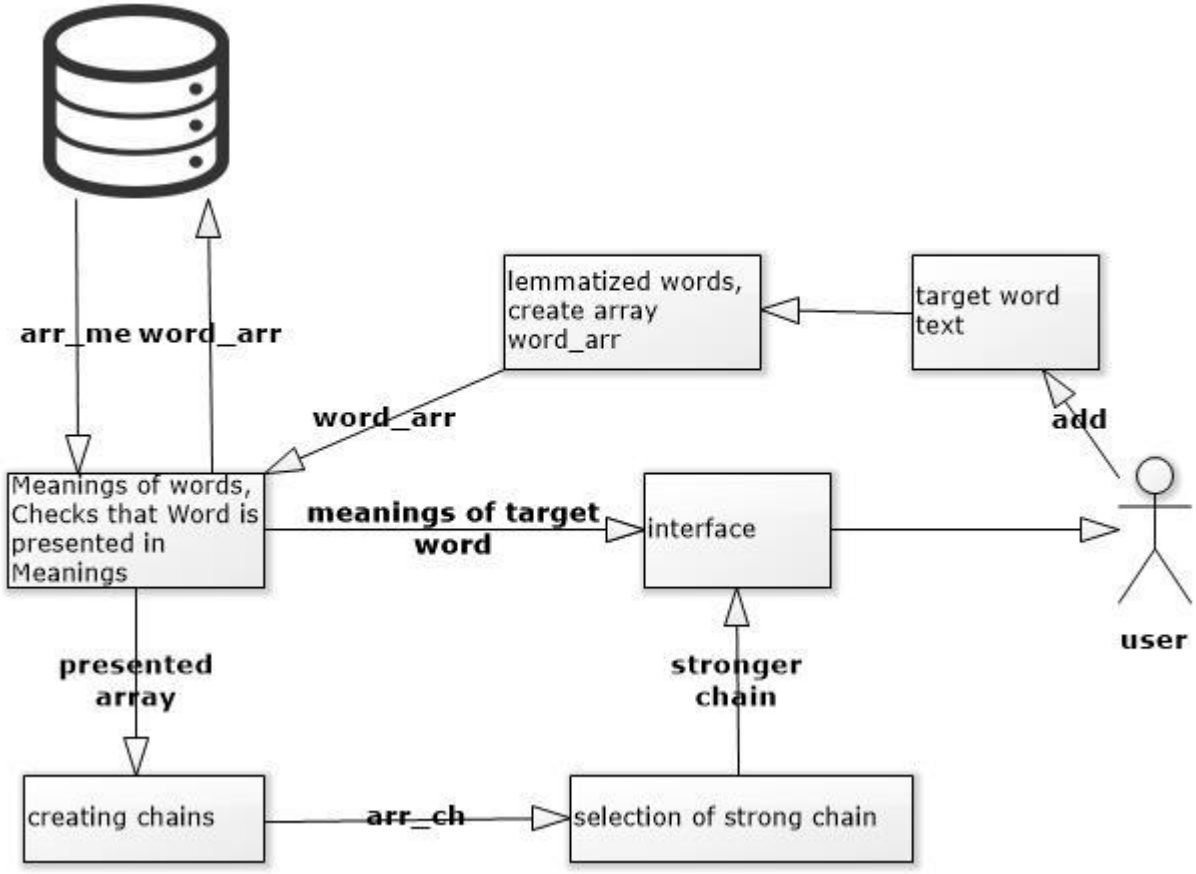

Рисунок 2. Архитектура приложения

Пользователь вводит в форму целевое слово и фрагмент текста, в котором данное слово употребляется (target word, text), затем фрагмент текста передается в блок lemmatized words, где текст разбивается на массив из нормализованных слов (word_arr), после этого полученный массив попадает в блок meanings of words, где происходит сопоставление каждого слова со словами в базе данных Викисловаря и получение списков всех значений каждого слова. Далее пользователю выводится список всех значений целевого слова, а список всех значений слов контекста передается в блок creating chains, где происходит построение различных интерпретаций лексической цепочки. Полученный массив интерпретаций передается в блок selection of strong chain, где выбирается самая сильная интерпретация

цепочки и извлечение из данной цепочки значения целевого слова, которое получает пользователь.

Стратегия тестирования

Для проведения тестирования используется библиотека PHPUnit.

PHPUnit – это специальный фреймворк, предназначенный для модульного тестирования скриптов языка PHP, разработанный Себастьяном Бергманом. Преимущества PHPUnit:

- Инструменты для создания модульных тестов и организации их в иерархические наборы.
- Интерфейс командной строки для выполнения тестов.
- Провайдеры данных – генераторы наборов данных, для тестирования элементов скрипта, используя различные входные параметры.
- Поддержка тестирования кода, работающего с базой данных.
- Тестирование исключений.
- Поддержка так называемых фиктивных объектов.
- Генераторы отчетов.

Блочное тестирование

Объектами для блочного тестирования выбраны: функции divideText, hasWord и getMeaningsArrayWithLemma.

Интеграционное тестирование

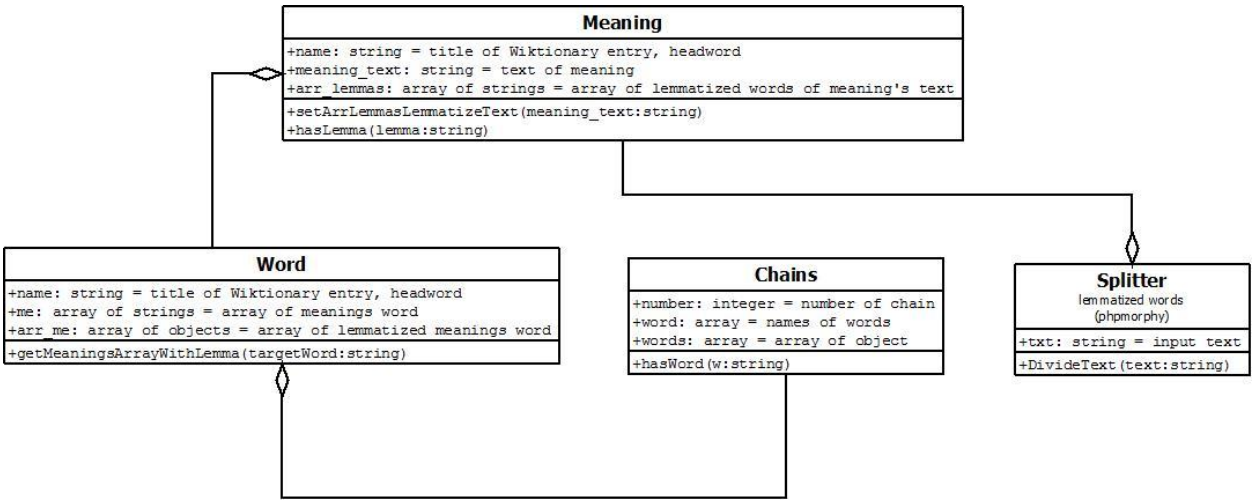

Рисунок 3. Зависимость классов приложения

Для проведения интеграционного тестирования применяется стратегия восходящего тестирования.

Порядок интеграции:

1. Splitter() + Meaning()
2. Meaning () + Word()
3. Word() + Chains()

Интеграция между модулями Splitter и Meaning осуществлена с помощью метода setArrLemmasLemmatizeText класса Meaning. Метод setArrLemmasLemmatizeText устанавливает объекту Meaning свойство $arr_lemmas, которое получается путем взаимодействия с методом DivideText класса Splitter.

Интеграция между классами Meaning и Word осуществлена с помощью метода getMeaningsArrayWithLemma класса Word, который ищет употребление слова lemma в значениях других слов контекста. Данный метод в свою очередь использует метод hasLemma класса Meaning, который проверяет, совпадает ли целевое слово с каким либо из слов в конкретном значении.

Интеграция между модулями Word и Chains осуществлена с помощью метода HasWord класса Chains. Метод HasWord проверяет принадлежность

словарного объекта конкретной лексической цепочке путем сравнения имени слова в объекте со словами, используемыми в значениях слов цепочки.

Аттестационное тестирование

Аттестационное тестирование проводится для всей системы, что подразумевает выполнение действий в пользовательском интерфейсе. Для проведения тестирования в данном случае необходим браузер.

Нагрузочное тестирование

Так как точность решения задачи зависит от количества слов контекста, из которых строится лексическая цепочка, то главным параметром для оценки работы программы было выбрано время, которое требуется для построения лексической цепочки.

В процессе тестирования выполнялась следующая последовательность действий:

1. запуск программы
2. задание критических значений
3. оценка времени работы программы
4. завершение программы

Критерии остановки и возобновления тестирования

Тест считается успешно пройденным, если ожидаемый и фактический результаты совпадают. В противном случае производится заключение о найденной ошибке. В связи со стратегией тестирования, тестирование сценариев более высокого уровня иерархии должно быть приостановлено при нахождении критических ошибок в вызываемых ими сценариях. В случаях возникновения существенных ошибок тестирование может быть продолжено на усмотрение тестировщика. Работы по тестированию возобновляются после исправления ошибок, вызвавших приостановку тестирования.

Детальный план тестов

Блочное тестирование

Объектами для блочного тестирования выбраны: функции divideText, hasWord и getMeaningsArrayWithLemma.

Функция divideText предназначена для разбиения текста на массив из слов. Слова в полученном массиве приводятся к нормализованному виду и все символы переводятся в нижний регистр. Для данной функции было разработано несколько блочных тестов:

- Сопоставление ожидаемого и реального значения результата:

№	Условия	Ожидаемый результат	Тип теста
1.1	Пустая строка	Array()	Негативный
1.2	'яблоко'	Array('яблоко')	Общий
1.3	'как-никак'	Array('как-никак')	Общий
1.4	"microsoft windows"	array('microsoft', 'windows')	Общий
1.5	"ЯБЛОКО, БаНан, лимон!!!"	array("яблоко", "лимон", "банан")	Общий

- Проверка существования элемента 'банан' в результирующем массиве при входном тексте "ЯБЛОКО, БаНан, лимон!!!"

№	Условия	Ожидаемый результат	Тип теста
1.6	'банан'	true	Общий

- Проверка существования элемента типа string в результирующем массиве при входном тексте "ЯБЛОКО, БаНан, лимон!!!"

№	Условия	Ожидаемый результат	Тип теста
1.7	string	true	Общий

Функция hasWord предназначена для определения принадлежности слова к лексической цепочке. Если свойство name объекта Word совпадает с каким либо из слов в цепочке, то функция возвращает true, иначе false. Входным параметром функции являет объект.

Для данной функции было разработано несколько блочных тестов:

- Сопоставление ожидаемого и реального значения результата:

№	Значение свойства name объекта word	Значение свойства me объекта находящегося в цепочке	Ожидаемый результат	Тип теста
1.8	"впечатление"	"способность живого существа воспринимать внешние впечатления, ощущать, испытывать что-нибудь"	true	Общий
1.9	"вертолет"	"способность живого существа воспринимать внешние впечатления, ощущать, испытывать что-нибудь"	false	Негативный
1.10	Пустая строка	"способность живого	false	Негативный

		существа воспринимать внешние впечатления, ощущать, испытывать что-нибудь"		
1.11	Свойста объекта word не определены	"способность живого существа воспринимать внешние впечатления, ощущать, испытывать что-нибудь"	false	Негативный

Метод getMeaningsArrayWithLemma класса Word ищет употребление слова lemma в значениях других слов контекста.

№	**Входной параметр**	**Массив имеющихся значений**	**Ожидаемый результат**	**Тип теста**
1.12	"способность"	"способности живого существа воспринимать внешние впечатления"	array("способности живого существа воспринимать внешние впечатления")	Общий
1.13	"способность"	"способности живого существа	array("способности живого существа воспринимать	Общий

		воспринимать внешние впечатления", "живого существа воспринимать внешние впечатления", "живого существа воспринимать способностей внешние впечатления"	внешние впечатления", "живого существа воспринимать способностей внешние впечатления")	
1.14	Пустая строка	"способность живого существа воспринимать внешние впечатления"	Array()	Негативный
1.15	"способность"	"живого существа воспринимать внешние впечатления"	Array()	Негативный
1.16	"способность"	NULL	Array()	Негативный

Пример тестов:

```
class SplitterTest extends PHPUnit_Framework_TestCase{
```

```
	/**
	* @dataProvider providerDivideText
	*/
	public function testDivideText($a,$b){
		$test = new Splitter();
		$this->assertEquals($a, $test->DivideText($b));
	}
	public function providerDivideText(){
		return array (
				array (array('яблоко'), 'яблоко'),
				array (array(array()), ""),
				array (array('как-никак'), "как-никак"),
				array (array('microsoft', 'windows'), "Microsoft windows"),
				array (array("яблоко", "лимон", "банан"), "ЯБЛОКО, БаНан, лимон!!!")
				);
	}
}
```

Интеграционное тестирование

Интеграция между классами Meaning и Word осуществлена с помощью метода getMeaningsArrayWithLemma класса Word, который ищет употребление слова lemma в значениях других слов контекста. Данный метод в свою очередь использует метод hasLemma класса Meaning, который проверяет, совпадает ли целевое слово с каким либо из слов в конкретном значении.

Тесты:

№	Входной параметр	Массив имеющихся значений	Ожидаемый результат	Тип теста
2.1	"способность"	"способности живого существа воспринимать внешние впечатления"	array("способности живого существа воспринимать внешние впечатления")	Общий

2.2	"способность"	"способности живого существа воспринимать внешние впечатления", "живого существа воспринимать внешние впечатления", "живого существа воспринимать способностей внешние впечатления"	array("способности живого существа воспринимать внешние впечатления", "живого существа воспринимать способностей внешние впечатления")	Общий
2.3	Пустая строка	"способность живого существа воспринимать внешние впечатления"	Array()	Негативный
2.4	"способность"	"живого существа воспринимать внешние впечатления"	Array()	Негативный
2.5	"способность"	NULL	Array()	Негативный

Интеграция между модулями Word и Chains осуществлена с помощью метода HasWord класса Chains. Метод HasWord проверяет принадлежность словарного объекта конкретной лексической цепочке путем сравнения имени слова в объекте со словами, используемыми в значениях слов цепочки.

№	**Входной параметр**	**Массив имеющихся значений**	**Ожидаемый результат**	**Тип теста**
2.6	Word1 (объект, содержащий в имени слово «впечатление»)	Word (объект, содержащий в значение с употреблением данного слова)	true	Общий
2.7	Word1 (объект, содержащий в имени слово «вертолет»)	Word (объект, не содержащий в значение с употреблением данного слова)	false	Негативный
2.8	Word1 (объект, содержащий в имени пустую строку	Объект Word	false	Негативный
2.9	Объект NULL	Объект Word	false	Негативный

Интеграция между модулями Splitter и Meaning осуществлена с помощью метода setArrLemmasLemmatizeText класса Meaning. Метод setArrLemmasLemmatizeText устанавливает объекту Meaning свойство $arr_lemmas, которое получается путем взаимодействия с методом DivideText класса Splitter.

№	Условия	Ожидаемый результат	Тип теста

2.10	Пустая строка	Array()	Негативный
2.11	'арбуз'	Array('арбуз')	Общий
2.12	"12234"	array()	Негативный
2.13	"Разрешение лексической многозначности"	array("разрешение", "лексической", "многозначности")	Общий

Пример теста:

```
class ChainTest extends PHPUnit_Framework_TestCase{
        public function testHasWord(){
                        $word = new Word;
                        $word->name = "способность";
                        $word->name_related_word = "чувство";
                        $word->me = "способность живого существа воспринимать внешние
                                впечатления, ощущать, испытывать что-нибудь";
                        $word->arr_me = Array ('что-
                                        нибудь','существо','способность','ощущать',
                                        'испытывать','воспринимать','живой','впечатле
                                        ние','внешний');

                        $word1 = new Word;
                        $word1->name = "впечатление";
                        $word1->name_related_word = "чувство";
                        $word1->me = "способность живого существа воспринимать внешние
                                впечатления, ощущать, испытывать что-нибудь";
                        $word1->arr_me = Array ('что-нибудь','существо',
                                        'способность','ощущать','испытывать',
                                        'воспринимать','живой','впечатление',
                                        'внешний');

                $test = new Chain;
                        $test->number=1;
                        $test->words=Array($word);
                        $this->assertEquals(true, $test->hasWord($word1));
                }
```

Аттестационное тестирование

3.1 Сценарий: Отправка пустой формы

Тестируемый модуль: wsd.php

Ожидаемый результат: Вывод сообщения «Введите интересующее слово и контекст!»

Тип теста: Негативный

3.2 Сценарий: Ввод слова без контекста

Тестируемый модуль: wsd.php

Ожидаемый результат: Вывод сообщения «Введите контекст!»

Тип теста: Негативный

3.3 Сценарий: Ввод контекста без слова

Тестируемый модуль: wsd.php

Ожидаемый результат: Вывод сообщения «Введите интересующее слово!»

Тип теста: Негативный

3.4 Сценарий: Ввод слова «дом» и контекста «Любовь к Родине – одно из самых мощных, возвышенных чувств. Она в полной мере проявилась в братской поддержке жителей Крыма и Севастополя, когда они твердо решили вернуться в свой родной дом»

Тестируемый модуль: chains.php

Ожидаемый результат: Дом - место, где кто-либо постоянно проживает

Тип теста: Общий

Нагрузочное тестирование

В процессе тестирования выполнялась следующая последовательность действий:

1. запуск программы

2. задание критических значений

3. оценка времени работы программы

4. завершение программы

Количество слов	**Время обработки**
3	< 5 сек

5	< 5 сек
10	< 10 сек
15	< 10 сек
20	< 10 сек
30	< 25 сек
35	<30 сек
40	<30 сек
50	<40 сек

Покрытие кода тестами

Расчет тестового покрытия относительно исполняемого кода программного обеспечения проводится по формуле:

$$T_{cov} = \frac{L_{tc}}{L_{code}} \times 100\%$$

Где:

T_{cov} - тестовое покрытие;

L_{tc} - количество строк кода, покрытых тестами;

L_{code} – общее количество строк кода;

Тогда:

$$T_{cov} = \frac{L_{tc}}{L_{code}} \times 100\% = \frac{193}{250} \times 100\% = 77.2\%$$

Результаты тестирования:

№	Результат
1.1	пройден
1.2	пройден
1.3	пройден
1.4	**ошибка**
1.5	пройден
1.6	пройден
1.7	пройден
1.8	пройден

1.9	пройден
1.10	пройден
1.11	пройден
1.12	пройден
1.13	пройден
1.14	пройден
1.5	пройден
1.16	пройден
2.1	**ошибка**
2.2	пройден
2.3	пройден
2.4	пройден
2.5	пройден
2.6	пройден
2.7	пройден
2.8	пройден
2.9	пройден
2.10	пройден
2.11	пройден
2.12	пройден
2.13	пройден
3.1	пройден
3.2	пройден
3.3	пройден
3.4	пройден

Найденные ошибки:

Тест 1.4 (блочный)

Ошибка: invalid argument supplied for foreach()

Причина: отсутствие в базе данных phpmorphy англоязычных слов

Статус: не исправлена, так как на данном этапе разработки нет необходимости расширения программы для разрешения многозначности с использованием других языков кроме русского

Тест 2.1 (интеграционный)

Ошибка: failed asserting that false matches expected true

Причина: возвращение false внутри основного цикла

```
foreach($ws->arr_me as $arr){
    if($arr==$w->name){
        return true;
}return false;}}
```

Статус: исправлена

```
foreach($ws->arr_me as $arr){
    if($arr==$w->name){
        return true;
}}}return false;
```

Результаты повторного тестирования:

№	Результат
1.1	пройден
1.2	пройден
1.3	пройден
1.4	**ошибка**
1.5	пройден
1.6	пройден
1.7	пройден

1.8	пройден
1.9	пройден
1.10	пройден
1.11	пройден
1.12	пройден
1.13	пройден
1.14	пройден
1.5	пройден
1.16	пройден
2.1	пройден
2.2	пройден
2.3	пройден
2.4	пройден
2.5	пройден
2.6	пройден
2.7	пройден
2.8	пройден
2.9	пройден
2.10	пройден
2.11	пройден
2.12	пройден
2.13	пройден
3.1	пройден
3.2	пройден
3.3	пройден
3.4	пройден