МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ПРОФЕССИОНАЛЬНОГО ОБРАЗОВАНИЯ

*«САРАТОВСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ ИМЕНИ ГАГАРИНА Ю.А.»*

МЕЖДУНАРОДНЫЙ ФАКУЛЬТЕТ ПРИКЛАДНЫХ ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ

Кафедра «Информационные системы и технологии»

Отчет по научно-исследовательской работе\_\_\_\_\_\_\_\_\_\_\_\_

|  |  |
| --- | --- |
| Место прохождения | СГТУ имени Гагарина Ю.А. |
| Время прохождения |  |

|  |  |  |  |
| --- | --- | --- | --- |
| Выполнил студент группы |  |  |  |
|  | ФИО | Подпись | Дата |
| Руководитель от СГТУ имени Гагарина Ю.А. |  |  |  |
|  | ФИО | Подпись | Дата |
| Руководитель |  |  |  |
|  | ФИО | Подпись | Дата |

|  |  |
| --- | --- |
| Оценка руководителя |  |
| Итоговая оценка по защите результатов работы |  |

Саратов 2015

# **Содержание**

[Содержание 2](#_Toc450607211)

[Введение 3](#_Toc450607212)

[1. Выбор программных и языковых средств 4](#_Toc450607213)

[2. Выбор методологии разработки програмного обеспечения 7](#_Toc450607214)

[Заключение 13](#_Toc450607215)

[Список использованных источников 14](#_Toc450607216)

# **Введение**

На сегодняшний день объем информации в Web колоссален и с каждым днем он становится все больше и больше, поэтому необходимо проводить исследования в области методов и подходов к созданию, предоставлению и обработке информации. Один из этих подходов – применение технологий Семантического Веба (Semantic Web). Концепция Semantic Web была принята и активно продвигается Консорциумом W3C (World Wide Web Consortium), который внедряет и разрабатывает стандарты для World Wide Web. Данная концепция является стеком технологий, существующих над Всемирной паутиной (World Wide Web), который позволяет размещать информацию в сети Интернет в виде, пригодном для машинной обработки[1].

В настоящее время технологии Семантического Веба, также известного как Web 3.0, практически не используются в коммерческих проектах[2]. Это связано, в первую очередь, со значительным увеличением стоимости и времени разработки.

В выпускной квалификационной работе планируется частично решить эту проблему путем создания библиотеки для генерации семантических данных. Целью данной работы является выполнение проектирования библиотеки. Для выполнения этой цели требуется выполнить следующие задачи:

* Выбор программных и языковых средств;
* Выбрать методологию и организовать процесс разработки.

# **Выбор программных и языковых средств**

При разработке в процессе выполнения выпускной квалификационной работы программного продукта была использована платформа .NET Framework версии 4.6.2 (CLR 4.0). .NET Framework – это программная платформа, которая была выпущена Microsoft в 2002 году. Принцип данной платформы заключается в том, что программа, написанная на любом поддерживаемом языке программирования сохраняется в сборку (assembly), содержащаую специальный, независимый от используемого аппаратного обеспечения IL-код (Intermediate Language). Сборка уже выполняется виртуальной машиной CLR (Common Language Runtime). Это позволяет программистам разрабатывать программы на любом языке программирования, а также не заботиться об особенностях аппаратной части.

Платформа .NET была выбрана в связи с рядом значимых преимуществ перед конкурентами, в частности Java:

* «Синтаксический сахар» C#, который значительно увеличивает скорость разработки приложения;
* Удобство компиляции и внедрения компонентов в операционную систему Windows;
* Улучшенная скорость выполнения кода в связи с тем, что CLR оптимизирует IL код под различные архитектуры процессоров;
* Самый удобный на рынке фреймворк для разработки Web-приложений ASP.NET (в частности использовался ASP.NET WebAPI 2);
* Удобство отладки, предоставленное IDE Microsoft Visual Studio 2015;
* Набор технологий LINQ (LINQ to SQL, LINQ.Expressions), которые предоставляют не только богатые возможности работы с коллекциями C#, но и возможности работы с внешними источниками данных, не углубляясь в детали их реализации;
* Наличие библиотеки dotNetRdf с открытым исходным кодом.

В качестве отрицательной стороны данной платформы можно отметить отсутствие кроссплатформенности, однако этот минус на сегодняшний день является незначительным в связи, во-первых, со стандартизацией технологий Microsoft. Это значит, что приложение можно запустить и на остальных операционных системах, таких как GNU/Linux, однако это потребует значительно больших затрат ресурсов и времени. Во-вторых, приложение в перспективе планируется перенести на платформу .NET Core, которая является кроссплатформенной и использует язык C# и инфраструктуру NuGET. В данный момент технология является нестабильной и разрабатывать библиотеку на ней сегодня нецелесообразно в связи с большими рисками.

Приложение было написано на языке C#. Это объектно-ориентированный язык, реализующий принципы инкапсуляции, наследования и полиморфизма. Он является наиболее используемым языком в .NET Framework. C# является одним из наиболее используемых языков программирования для разработки прикладного программного обеспечения в промышленной среде.

Для разработки программного обеспечения потребовалось использовать следующие программные продукты:

* IDE Microsoft Visual Studio 2015 Enterprise;
* JetBrains ReSharper 10 Ultimate;
* MS SQL Management Studio 2014;
* Библиотека dotNetRdf для работы с семантическими данными;
* ASP.NET WebAPI 2;
* Entity Framework 6.0.

Рассмотрим подробнее данные программные средства.

В качестве интегрированной среды разработки была выбрана Microsoft Visual Studio 2015 Ultimate. Это наилучший из вариантов IDE для разработки на C# для Windows. В качестве альтернатив можно было использовать Visual Studio Code или Mono, однако они значительно уступают в функциональности, поэтому даже не рассматривались на роль IDE для разработки[3].

Линейка продуктов JetBrains ReSharper Ultimate использовалась для ускорения процесса разработки. Это набор надстроек над Visual Studio, который автоматизирует часто повторяющиеся действия, выполняет статический анализ кода, следит за соблюдением стилистики кода, содержит средства для декомпиляции, запуска тестов и анализа покрытия тестами. В общем случае, данное средство ускоряет процесс разработки в 2-3 раза[4].

MS SQL Management Studio использовалась для работы с базой данных. В частности, для отладки процесса создания базы данных Entity Framework. Сам Entity Framework используется для управления базой данных. Он полностью создает базу данных из классов C#, генерируя к ней запросы «на лету».

Для работы с RDF-данными была использована библиотека dotNetRdf[5]. Библиотека была внедрена в проект как NuGet-пакет.

# **Выбор методологии разработки програмного обеспечения**

Современная разработка программного обеспечения стремится к тому, чтобы минимизировать затраты ресурсов, затрачиваемых на создание и поддержку программных продуктов. Под ресурсами в данном случае понимается цена и время. Также значительным фактором является минимизация количества ошибок и повышения качества продукта в целом.

Для управления вышеизложенными параметрами в коммерческих проектах применяются методологии разработки программного обеспечения. В общем случае, методология – это система принципов, совокупность понятий, идей, способов и средств, определяющих стиль разработки программного обеспечения[6]. Методология – это реализация старндарта. Сами стандарты указывают на то, как должно быть, методология оставляет свободу выбора и адаптации.

Конкретные процессы разработки программного обеспечения реализуются через выбранную методологию. Именно она определяет как именно выполняется разработка. Существует большое количество методологий программного обеспечения, таких как каскадная, итеративная, спиральная. Все они имеют свои положительные и отрицательные стороны. В большинстве коммерческих проектов используется группа методологий Agile (гибкие методологии разработки). Эта группа методологий получила широкое распространение в связи с тем, что именно методологии этой группы позволяют быстрее всего реагировать на изменение требований с минимальными потерями времени, денег (в случае, если проект коммерческий) и потерями качества. Также эти методологии обладают дополнительными положительными сторонами, которые являются следствием гибкости разработки.

Группа методологий Agile имеет манифест, который обьединяет все методологии внутри него[7]:

* Люди и взаимодействие важнее процессов и инструментов;
* Работающий продукт важнее исчерпывающей документации;
* Сотрудничество с заказчиком важнее согласования условий контракта;
* Готовность к изменениям важнее следования первоначальному плану.

Так как проект планируется поддерживать и постоянно дорабатывать, то очевидно, что для разработки будет выбрана одна из методологий Agile.

Самой известной методологией разработки Agile является Scrum[8]. Дословно, с английского название переводится как «толкучка». Основной акцент делается на качественном контроле за процессом разработки. Разработка поделена на итерации, которые называются спринтами. Выполнение спринта, все роли, обсужения и артефакты четко регламентированы. Их исполнение и применение контролируется специальным человеком – Scrum-мастером.

По методологии Scrum все задачи с прикрепленным к ним приоритетом складываются в специальный список задач владельцем продукта (список задач называется Backlog).

Спринты занимают 1-4 недели и являются неизменяемыми единицами выполненной работы с точки зрения владельца продукта. Спринт начинается с планирования, в рамках которого команда выбирает из Backlog-а задачи, которые будет выполнять в текущий спринт. Задачи, которые не входят в спринт целиком декомпозируются. Каждый день во время спринта команда собирается и каждый ее участник сообщает о том, что он сделал, что собирается делать и какие вопросы у него есть. Это называется ежедневный митинг, позволяет отслеживать общий прогресс команды и принимать быстрые решения вовремя в случае экстренных ситуаций. По окончании спринта для владельца продукта производится демонстрация новой функциональности рабочего продукта. После сдачи спринта проводится ретроспектива с целью просуммировать командный опыт по результатам спринта.

Положительные стороны Scrum:

* Прозрачность для всех участников процесса;
* Короткие спринты позволяют быстро применять изменения;
* Возможность учиться на своих ошибках как команда (благодаря ретроспективам);
* Возможность применять качественные решения;

Однако, у Scrum есть и отрицательные стороны, например постоянный контроль над командой. Это, во-первых, незначительно снижает мотивацию команды. Во-вторых, на процессы уходит значительное количество времени. Также Scrum вынуждает команду по результатам спринта создавать ветки. В случае обнаружения критических ошибок приходится разрабатывать продукт в нескольких ветках, а затем сливать. Это неудобно и требует дополнительных ресурсов.

Следующей методологией разработки ПО является канбан (Kanban). Принято считать, что эта методология разработки наиболее удачно подходит для проектов находящихся на стадии поддержки. Основная идея методологии – минимизация количества одновременно выполняющихся задач. Для этого создается специальная таблица c задачами, на которой видно сколько задач в текущий момент выполняет команда.

Положительные стороны Kanban:

* Отсутствие документации;
* Прозрачность процесса;
* Отсутствие сроков;
* Быстрое применение изменений;

Отрицательные стороны:

* Отсутствие сроков может повлечь за собой снижение производительности команды, если команда не мотивирована;
* Не подходит для разработки новых продуктов.

Рассмотрим следующую методологию разработки – Extreme Programming (XP)[9]. Данная методология предусматривает максимальное упрощение организации процесса разработки в условиях постоянно меняющихся требований. Основной целью методологии является повышение качества продукта в минимальные сроки. Общий смысл методологии в применении двенадцати практик:

* Планирование процесса;
* Тесное взаимодействие с заказчиком;
* Общесистемные правила именования;
* Простая архитектура;
* Рефакторинг;
* Парное программирование;
* 40-часовая рабочая неделя;
* Коллективное владение кодом;
* Единые стандарты кодирования;
* Небольшие релизы;
* Непрерывная интеграция;
* Разработка через тестирование (TDD).

Процесс XP является неформальным и требует высокого уровня самодисциплины каждого их разработчиков. В XP каждый член команды считается профессиональным самомотивированным работником.

Положительные стороны XP:

* Полное отсутствие как документации, так и артефактов;
* Отсутствие потерь времени на процессы;
* Быстрое применение изменений;
* Выполнение задач быстрее, чем в рамках других методологий;ъ
* Гарантия качества выполнения задач за счет некоторых практик (TDD, парное программирование, коллективное владение кодом).

Стоит отметить, что данная методология имеет ограничения, которые одновременно и являются ее отрицательной стороной. XP применим только для небольших и средних команд. Для большой команды невозможно применить некоторые из ключевых практик и процессы на проекте превращаются в хаос. Также методология применима только для самодисциплинируемых программистов. В противном случае методологию применять бессмыслено в связи с отсутствием контроля как со стороны менеджмента, так и со стороны программиста. Эта причина является ключевым моментом в том, что самой популярной методологией является Scrum, а не XP, несмотря на то, что XP потенциально предоставляет больший уровень производительности команды.

Все вышеизложенные методологии объединяет отсутствие документации. Однако, не все методологии Agile предполагают ее отсутствие. Например, существует методология RUP (Rational Unified Process)[10]. Это методология разработки программного обеспечения, которая была разработана компанией Rational Software, с целью создать методологию, которая позволит скомбинировать положительные стороны водопадной модели и Agile. Аналогично водопаду, процесс разработки разделен на фазы, результатами которых является соответствующая документация. Однако, в данной методологии требования могут меняться на любой стадии.

Положительные стороны:

* Этапы могут выполнять разные команды;
* Предсказуемость;
* Качество продукта;
* Прозрачность процесса, которая не теряет актуальность для команд большого размера.

Отрицательные стороны:

* Колоссальные потери времени на процессы.

Для разработки библиотеки генерации семантических данных была выбрана методология Extreme Programming. RUP и Scrum были отсеяны ввиду требования документации и соблюдения процессов, времени на написание которой нет. Kanban может использоваться в поддержке фреймворка, однако для его разработки рационально использовать XP. Ограничения методологии Extreme Programming на исполнителя не распространяются, в связи с тем, что команда состоит из одного человека, который достаточно мотивирован, чтобы методология работала.

Также, ключевым моментом является техника написания тестов TDD, которая входит в практики Extreme Programming. Данный подход заключается в том, чтобы сначала писать тесты, а потом уже код. Такая практика позволяет значительно повысить качество кода и снизить время на исправление ошибок ценой небольшого увеличения сроков разработки.

# **Заключение**

В данной работе было выполнено проектирование фреймворка для генерации семантических данных. В процессе проектирования были сделаны следующие выводы:

* Для создания программного продукта наиболее перспективной является платформа .NET Framework 4.6.2;
* Для автоматизации работы с базами данных наиболее рационально использовать EntityFramework 6;
* Для работы с Web 2.0 лучшим выбором является использование ASP.NET WebAPI 2, а для Web 3.0 – dotNetRDF;
* Для ускорения процесса разработки следует использовать JetBrains ReSharper 10 Ultimate;

Также, в данной работе были исследованы методологии разработки программного обеспечения. По результатам была выбрана методология Extreme Programming. В условиях написания приложения в качестве дипломной работы с возможностью дальнейшего развития эта методология выглядит наиболее выгодной.

Ключевым моментов в выборе Extreme Programming является техника написания тестов Test-Driven Development (TDD).

Предполагается, что библиотека будет иметь востребованность среди проектов на платформе .NET. В любом случае, реализация данной библиотеки является одним шагом вперед к повсеместному использованию технологий Семантического Веб.

# **Список использованных источников**

1. Berners-Lee, T. The Semantic Web. A new form of Web content that is meaningful to computers will unleash a revolution of new possibilities [Электронный ресурс] / T. Berners-Lee, J. Hendler, O. Lassila. Режим доступа: http://jeckle.de/files/tblSW.pdf (дата обращения 11.07.2015).
2. Kashyap V. Real World Semantic Web Applications / V. Kashyap, L. Shklar. – IOS Press, 2002. – 197 c.
3. Официальный сайт IDE Microsoft Visual Studio 2015 [Электронный ресурс]. – Режим доступа: http://www.visualstudio.com (дата обращения 10.05.2016).
4. Официальный сайт JetBrains [Электронный ресурс]. – Режим доступа: https://www.jetbrains.com (дата обращения 10.05.2016).
5. Официальный сайт библиотеки dotNetRdf [Электронный ресурс]. – Режим доступа: http://www.dotnetrdf.org (дата обращения 10.05.2016)
6. Методологии разработки программного обеспечения [Электронный ресурс]. Режим доступа: https://habrahabr.ru/sandbox/43802/ (дата обращения 10.05.2016).
7. Agile-манифест [Электронный ресурс]. – Режим доступа: http://www.agilemanifesto.org/iso/en/ (дата обращения 10.05.2016).
8. The Official Scrum Rulebook [Электронный ресурс]. – Режим доступа: https://www.scrum.org (дата обращения 10.05.2016).
9. Don Wells. Extreme Programming: A gentle introduction [Электронный ресурс] / D. Wells. Режим доступа: http://www.extremeprogramming.org (дата обращения 10.05.2016).
10. Rational Software. Rational Unified Process. Best Practices for Software Development Teams [Электронный ресурс] / Rational Software. Режим доступа: https://www.ibm.com/developerworks/rational/library/content/03July/1000/1251/1251\_bestpractices\_TP026B.pdf (дата обращения 10.05.2016).