МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное автономное образовательное учреждение
высшего образования
**«Южно-Уральский государственный университет
(национальный исследовательский университет)»**
Высшая школа электроники и компьютерных наук
Кафедра системного программирования

РАБОТА ПРОВЕРЕНА

Рецензент
Руководитель группы сопровождения ООО «ИТ Менеджмент»

_____А.А. Нусратуллин
“___”_____ 2017 г.

ДОПУСТИТЬ К ЗАЩИТЕ

Заведующий кафедрой,
д.ф.-м.н., профессор

_____ Л.Б. Соколинский

“___”_____ 2017 г.

РАЗРАБОТКА ПРОГРАММНОЙ СИСТЕМЫ ДЛЯ АВТОМАТИЗИРОВАННОГО МОНИТОРИНГА ПРЕДСТОЯЩИХ НАУЧНЫХ КОНФЕРЕНЦИЙ

ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА
ЮУрГУ – 02.03.02.2017.13-057-1401.ВКР

Научный руководитель
кандидат физ.-мат. наук
_____ С.А. Иванов

Автор работы,
студент группы КЭ-402
_____ В.А. Сухоруков

Ученый секретарь
(нормоконтролер)

_____ О.Н. Иванова

“___”_____ 2017 г.

Челябинск-2017

ОГЛАВЛЕНИЕ

ВВЕДЕНИЕ

Актуальность темы

Конференции являются важной и неотъемлемой частью для обнародования научных достижений, обменом опыта, поиск коллег для будущих научных исследований. Информация о конференциях обновляется постоянно, и в силу загруженности преподавателя или студента, не всегда есть время следить за сроками подачи документов и сроками провидения.

Часто интересующие конференции остаются не посещенными, в силу того, что вовремя не было произведено информирование о сроках подачи документов или о сроках проведения.

Автоматизированный мониторинг поможет быть в курсе о предстоящих конференциях, вовремя формировать документы для конференций. На настоящий момент нет специальных программ для автоматизированного мониторинга предстоящих научных конференций.

Таким образом, актуальной является задача разработки программной системы для автоматизированного мониторинга предстоящих научных конференций.

Цель и задачи

Целью работы является разработка программной системы для автоматизированного мониторинга предстоящих научных конференций.

Для достижения данной цели необходимо выполнить следующие задачи:

1) выполнить анализ предметной области;
2) выполнить проектирование системы;
3) реализовать систему;
4) провести тестирование системы.

Структура и объем работы

Работа состоит из введения, четырех глав, заключения, библиографического списка. Объем работы составляет 32 страницы, объем библиографии – 21 источник.

Содержание работы

В главе «Теоретическая часть» приведено описание постановки задачи, приведен сравнительный анализ существующих аналогов, а также представлено описание средств разработки.

Глава «Проектирование программной системы» посвящена определению функциональных требований к разрабатываемому приложению. В этой же главе рассматривается диаграмма прецедентов и диаграмма классов с описанием.

В главе «Реализация приложения» представлен перечень используемых средств и технологии разработки, представлены реальные интерфейсы форм приложения, а также некоторые фрагменты исходного кода системы.

В главе «Тестирование приложения» представлено тестирование системы.

В заключении сделаны выводы о проделанной работе.

1. ТЕОРЕТИЧЕСКАЯ ЧАСТЬ

1.1. Постановка задачи

Практика доказала, что посещение научных конференций оказывается полезным. Конечно же, конференции не могут полностью заменить научную работу, но они могут дополнить эту деятельность. Научные конференции полезны тем, что позволяют делиться опытом, общаться на научные темы и все это в неформальной обстановке, что, несомненно, влияет на качество научного процесса.

Научные конференции позволяют ознакомиться с актуальными проблемами науки без конкретного участия в научных процессах. В том случае, если есть какие-то идеи, будет полезно их обсудить с теми, кто обладают знаниями и заинтересованы темой. Таким образом, конференции помогут не только ознакомиться с чужими открытиями, а также прийти к своим открытиям, убедиться в их правильности.

Участие в конференциях позволяет развивать личные и профессиональные качества, способствует знакомству с интересными людьми, которые могут оказаться полезными для будущих научных открытий.

Таким образом, разработка программной системы для автоматизированного мониторинга предстоящих научных конференций позволит решить задачу своевременного и актуального информирования о предстоящих конференциях, чтобы вовремя подготовится к интересующей конференции.

1.2. Сравнительный анализ существующих аналогов

В настоящий момент имеется ряд продуктов, позволяющих следить за обновлением информацией на сайте. Ниже представлен сравнительный анализ их функциональных возможностей и обзор недостатков.

«WebSvodka Demo» [10] – бесплатная пробная версия профессионального сервиса для отслеживания и мониторингом изменений на выбранном сайте.

Преимущества:

1) отслеживает изменения на заданных интернет – страницах;

2) сообщает о изменениях на выбранных сайтах;

3) уведомляет о появлении новых страниц Поисковых систем;

4) контролирует изменения файлов в Интернете.

Недостатки:

1) отслеживание до 3 объектов;

2) платная версия;

3) внесения изменений не сохраняются в demo версии

4) частота обновления реже одного раза в сутки.

«http://bit.sp.susu.ru» [13] – сайт, на котором предоставляется информация о научных конференциях для кафедры СП.

Преимущества:

1) проверенные источники;

2) полная информация о конференции.

Недостатки:

1) нет оповещения о изменениях;

2) нет информирования о новых конференциях.

«Конференции.ru» [14] – информационная поддержка научных мероприятий. Сервис предоставляет список всех предстоящих конференций на различные тематики и по всему миру.

Преимущества:

1) большое количество конференций на различные темы;

2) проверенные источники;

3) расширенная сортировка поиска по интересующим темам.

Недостатки:

1) не работает информирование по Email о новых конференциях.

Таким образом, проведенный сравнительный анализ существующих аналогов показал, что они не могут быть использованы для автоматизированного мониторинга предстоящих научных конференций из-за наличия в постановке задачи специфичных требований, не реализуемых другими программными продуктами.

1.3. Обзор программных средств разработки

AngleSharp [1] – это библиотека .NET, которая дает вам возможность разбирать гипертекст, такой как HTML, SVG и MathML. XML без проверки также поддерживается библиотекой. Важным аспектом AngleSharp является то, что CSS также может быть проанализирован. Парсер [6] построен на официальной спецификации W3C. Это создает совершенно переносимое DOM-представление HTML5 данного исходного кода. Также текущие функции, такие как querySelector или querySelec-torAll, работают для обхода дерева.

Преимущества:

1) написан на C#;
2) включает в себя парсеры других языков;
3) постоянная развитие и поддержка проекта.

Недостатки:

1) низкая скорость обработки таблиц.

CsQuery [3] – это порт jQuery для .NET 4. Он реализует все селектора CSS2 и CSS3, все методы манипуляции DOM в jQuery и некоторые из вспомогательных методов.

Преимущества:

1) схожесть с jQuery.

Недостатки:

1) на данный момент разработка CsQuery в пассивной стадии;
2) низкая скорость обработки страниц.

Fizzler [4] – библиотека .NET для выбора элементов из дерева узлов на основе селектора CSS. Реализация по умолчанию основана на HTMLAgilityPack.

Преимущества:

1) позволяет использовать селекторы CSS;
2) удобнее HtmlAgilityPack.

Недостатки:

1) разработка и поддержка приостановлена.

HtmlAgilityPack [7] – это быстрый HTML – парсер, который создает DOM для чтения / записи и поддерживает простые XPATH или XSLT. Это библиотека .NET, которая позволяет анализировать HTML-файлы вне Интернета. Парсер очень толерантен с «неправильным» HTML-кодом. Объектная модель очень похожа на то, что предлагает System.Xml, но для документов HTML (или потоков).

Преимущества:

1) большое количество документации.

Недостатки:

1) разработка и поддержка приостановлена.

Regex [9] – регулярные выражения предоставляют мощный, гибкий и эффективный способ обработки текста. Комплексная нотация сопоставления шаблонов регулярных выражений позволяет быстро анализировать большие объемы текста для поиска определенных шаблонов символов; проверять текст на соответствие предопределенному шаблону (например, адресу электронной почты); чтобы извлекать, изменять, заменять и удалять текстовые подстроки; а также чтобы добавлять извлеченные строки в коллекцию для создания отчета. Для многих приложений, которые работают со строками или анализируют большие блоки текста, регулярные выражения - незаменимый инструмент.

Преимущества:

1) быстрая обработка таблиц;

2) минимальное потребление процессорного времени и памяти.

Недостатки:

1) сложный для понимания.

Сравнение выше рассмотренных парсеров производилось по нескольким критериям: простота установки, количество документации, удобство использования, масштаб обычно разрабатываемых проектов с использованием данных средств (табл. 1).

Табл. 1. Обзор возможных для использования парсеров

№ п/п	Критерий	AngleSharp	CsQuery	Fizzler	Html Agility Pack	Regex
1.	Простота установки	5/5	4/5	4/5	3/5	5/5
2.	Количество документации	4/5	3/5	4/5	4/5	3/5
3.	Лицензия	Open Source	Open Source	Open Source	Open Source	Open Source
5.	Обработка CSS	ДА	ДА	ДА	НЕТ	НЕТ
6.	Среднее время обработки HTML	27.4 ms	42.2 ms	21.7 ms	20.6 ms	42.2 ms

На основании рассмотренных выше парсеров по нескольким критериям можно сделать вывод о том, что оптимальным выбором будет AngleSharp, так как он активно разрабатывается, обладает интуитивным API и показывает хорошо время обработки. AngleSharp наиболее подход для использования в разработке программной системы автоматизированного мониторинга предстоящих научных конференций. Установка AngleSharp показаны на рис. 1.

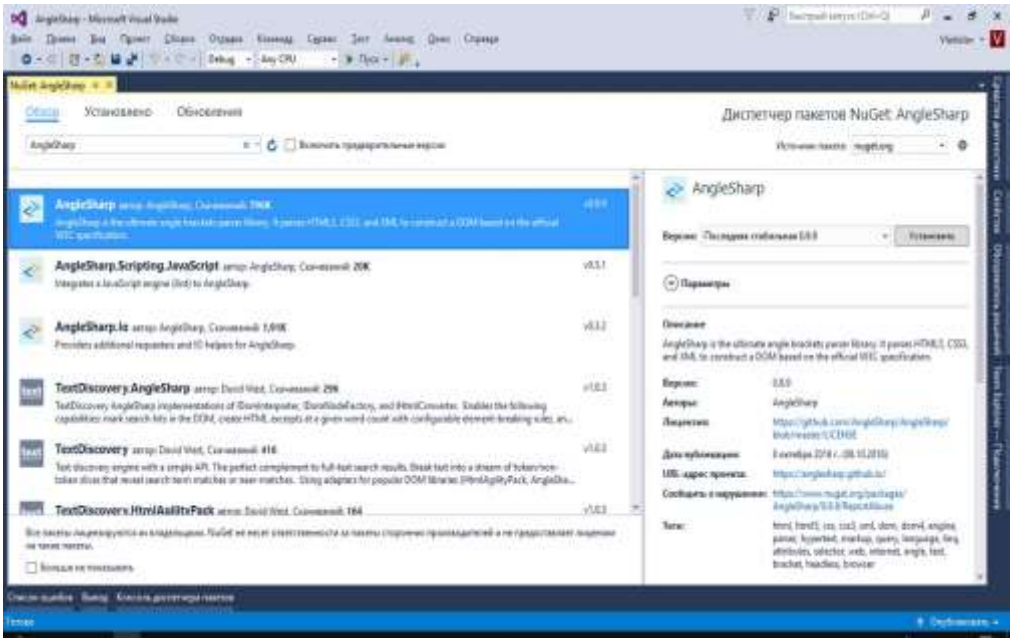

Рис. 1. Установка AngleSharp

1.4. Вывод

В данной главе был сделан обзор основных существующих решений для разработки программной системы автоматизированного мониторинга предстоящих научных конференций и выбран основной инструмент разработки.

2. ПРОЕКТИРОВАНИЕ ПРОГРАММНОЙ СИСТЕМЫ

В данном разделе описано проектирование создаваемой программной системы, выявлены основные требования к ней, варианты использования системы. Представлена диаграмма основных классов приложения, их взаимодействие между собой.

2.1. Назначение

Программная система для автоматизированного мониторинга предстоящих научных конференций предназначена для упрощения отслеживания научных конференций.

2.2. Определение функциональных требований

Требование можно определить, как подробное описание того, что должно быть реализовано.

В ходе проектирования приложения были определены функциональные возможности, предоставляемые пользователю.

При загрузке программной системы, пользователь должен увидеть главную форму с логотипом «FQW» и кнопками, которые позволят пользователю обновить список конференций, включить мониторинг, настроить программу, кнопки управления программой.

При нажатии на кнопку «Меню» пользователь должен увидеть появляющееся окно, в котором можно выбрать ресурс, за которым будет вестись наблюдение, по умолчанию это ресурс «konferencii.ru».

При нажатии на кнопку «Обновить» пользователь должен увидеть полный список всех предстоящих конференций.

При нажатии на кнопку «Мониторинг» пользователь должен увидеть окно мониторинга, в котором ведется отслеживание за обновлением информации о конференциях на выбранном ресурсе.

При нажатии на кнопку «Календарь» пользователь должен увидеть окно с текущей датой, количество предстоящих конференций.

Пользователь должен иметь возможность вернуться с каждой формы на предыдущую или главную форму.

2.3. Диаграмма прецедентов

Диаграммы прецедентов представляют собой один из пяти типов диаграмм, применяемых в UML [11, 16] для моделирования динамических аспектов системы (остальные четыре типа – это диаграммы деятельности, состояний, последовательностей и кооперации). Диаграммы прецедентов играют основную роль в моделировании поведения системы, подсистемы или класса. Каждая из таких диаграмм показывает множество прецедентов, актеров и отношения между ними.

Диаграммы прецедентов применяются для моделирования вида системы с точки зрения прецедентов (вариантов использования). Чаще всего это предполагает моделирование контекста системы, подсистемы или класса либо моделирование требований, предъявляемых к поведению указанных элементов.

Диаграммы прецедентов имеют большое значение для визуализации, специфицирования и документирования поведения элемента. Они облегчают понимание систем, подсистем или классов, представляя взгляд извне на то, как данные элементы могут быть использованы в соответствующем контексте. Кроме того, такие диаграммы важны для тестирования исполняемых систем в процессе прямого проектирования и понимания их внутреннего устройства при обратном проектировании.

Диаграммой прецедентов (use case diagram), называется диаграмма, на которой показана совокупность прецедентов и актеров, а также отношения между ними. Диаграммы прецедентов обладают стандартными свойствами, присущими любой диаграмме, именем и графическим содержанием, которое представляет собой одну из проекций модели. Диаграмма прецедентов отличается от прочих своим конкретным содержанием. Диаграммы прецедентов обычно включают в себя:

- прецеденты;
- актеров;
- отношения зависимости, обобщения и ассоциации;

- как и все остальные диаграммы, они могут содержать примечания и ограничения.

Иногда диаграммы прецедентов помещают в пакеты, применяемые для группирования элементов модели в более крупные блоки, в ряде случаев и экземпляры прецедентов, особенно если надо визуализировать конкретную исполняемую систему. Диаграммы прецедентов, или вариантов использования, применяют для моделирования статического вида системы с точки зрения прецедентов. Это вид охватывает главным образом поведение системы, то есть видимые извне сервисы, предоставляемые системой в контексте ее окружения. При моделировании статического вида системы с точки зрения прецедентов диаграммы использования обычно применяют двумя способами: для моделирования контекста системы и для моделирования требований.

Для описания работы системы необходимо рассмотреть диаграмму прецедентов, выявляющую основных пользователей системы и задачи, которые данная система должна решать (рис. 2).

Рис. 2. Диаграмма вариантов использования

В системе актером является пользователь программы.

Пользователь может выполнить *обновление списка конференций*, т.е. загрузить информацию о конференциях.

Пользователь может выполнить *просмотр всех конференций,* т.е. провести обзор всех предстоящих конференций.

Пользователь может выполнить *поиск по конференциям,* т.е. в списке конференций найти интересующею.

Пользователь может выполнить *настройку программы,* т.е. выбрать внешний вид отображения программы, и частоту обновления списка конференций.

2.4. Диаграмма классов

Диаграмма классов носит структурный характер. Она предназначена для отображения классов разрабатываемого приложения и их взаимосвязей.

На рис. 3 изображены основные классы, используемые при проектировании и реализации программной системы.

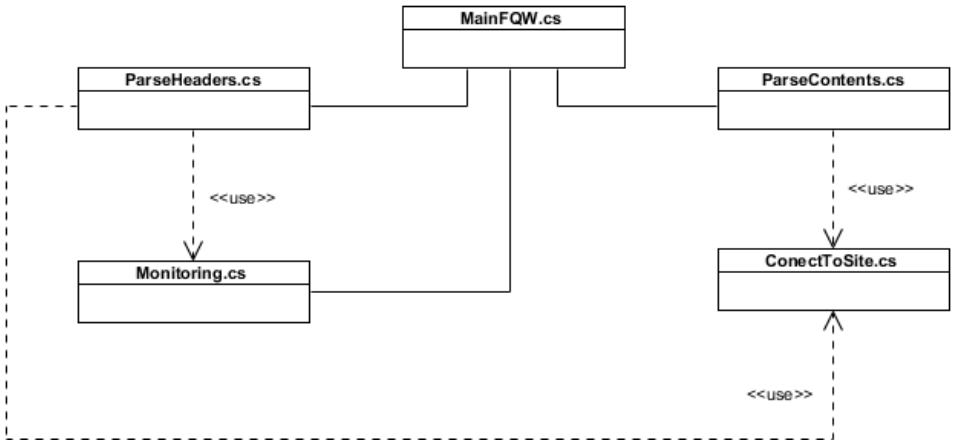

Рис. 3. Диаграмма классов

2.5. Вывод

Во второй главе были описаны основные требования к разрабатываемой программной системе, и в соответствии с этим, была спроектирована его диаграмма классов, и описаны основные варианты использования.

3. РЕАЛИЗАЦИЯ ПРИЛОЖЕНИЯ

В данном разделе описаны особенности разработки программной системы автоматизированного мониторинга предстоящих научных конференций.

Описаны основные инструменты, используемые при реализации, компоненты, из которых состоит приложение.

3.1. Реализация приложения

Для реализации был выбран язык программирования C# [21], и среда Microsoft Visual Studio Community 2015 [8]. Для реализации была выбрана технология Windows Presentation Foundation (WPF) [17].

Технология Windows Presentation Foundation (WPF) является часть экосистемы платформы .NET [20] и представляет собой подсистему для построения графических интерфейсов.

Если при создании традиционных приложений на основе WinForms за отрисовку элементов управления и графики отвечали такие части ОС Windows, как User32 и GDI+, то приложения WPF основаны на DirectX [19]. В этом состоит ключевая особенность рендеринга графики в WPF: используя WPF, значительная часть работы по отрисовке графики, как простейших кнопок, так и сложных 3D-моделей, ложиться на графический процессор на видеокарте, что также позволяет воспользоваться аппаратным ускорением графики.

Одной из важных особенностей является использование языка декларативной разметки интерфейса XAML, основанного на XML: можно создавать насыщенный графический интерфейс, используя или декларативное объявление интерфейса, или код на управляемых языках C# [20] и VB.NET, либо совмещать и то, и другое [18].

Преимущества WPF:

- использование традиционных языков .NET-платформы – C# и VB.NET для создания логики приложения;

- возможность декларативного определения графического интерфейса с помощью специального языка разметки XAML, основанном на xml и представляющем альтернативу программному созданию графики и элементов управления, а также возможность комбинировать XAML и C#/VB.NET;
- независимость от разрешения экрана: поскольку в WPF все элементы измеряются в независимых от устройства единицах, приложения на WPF легко масштабируются под разные экраны с разным разрешением;
- новые возможности, которых сложно было достичь в WinForms, например, создание трехмерных моделей, привязка данных, использование таких элементов, как стили, шаблоны, темы и др.;
- хорошее взаимодействие с WinForms, благодаря чему, например, в приложениях WPF можно использовать традиционные элементы управления из WinForms;
- богатые возможности по созданию различных приложений: это и мультимедиа, и двухмерная и трехмерная графика, и богатый набор встроенных элементов управления, а также возможность самим создавать новые элементы, создание анимации, привязка данных, стили, шаблоны;
- аппаратное ускорение графики - вне зависимости от того, работаете ли вы с 2D или 3D, графикой или текстом, все компоненты приложения транслируются в объекты, понятные Direct3D, и затем визуализируются с помощью процессора на видеокарте, что повышает производительность, делает графику более плавной;
- создание приложений под множество ОС семейства Windows - от Windows XP до Windows 10.

Архитектура WPF

Схематически архитектуру WPF представлена на рис. 4.

Как видно на схеме, WPF разбивается на два уровня: managed API и unmanaged API (уровень интеграции с DirectX).

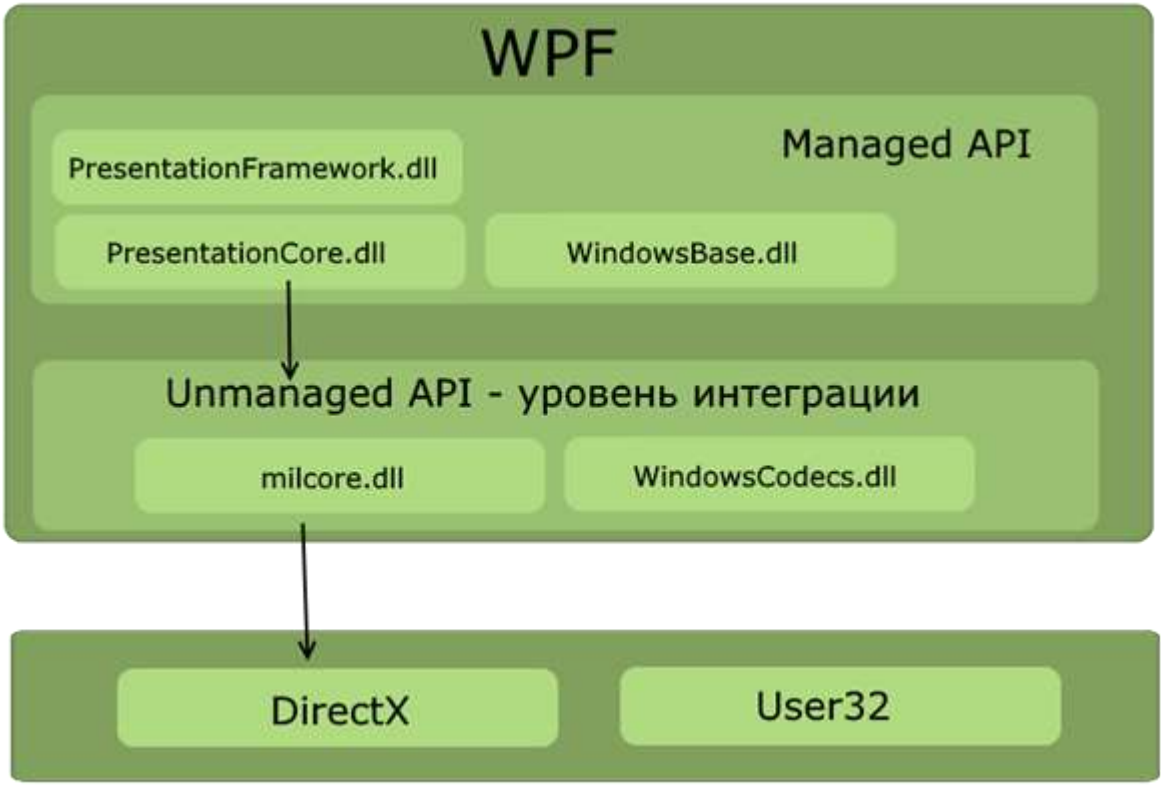

Рис. 4. Архитектуру WPF

Managed API (управляемый API-интерфейс) содержит код, исполняемый под управлением общеязыковой среды выполнения .NET - Common Language Runtime. Этот API описывает основной функционал платформы WPF и состоит из следующих компонентов:

PresentationFramework.dll: содержит все основные реализации компонентов и элементов управления, которые можно используются при построении графического интерфейса

PresentationCore.dll: содержит все базовые типы для большинства классов из PresentationFramework.dll

WindowsBase.dll: содержит ряд вспомогательных классов, которые применяются в WPF.

Unmanaged API используется для интеграции вышележащего уровня с DirectX. milcore.dll: обеспечивает интеграцию компонентов WPF с DirectX. Данный компонент написан на неуправляемом коде (C/C++) для взаимодействия с DirectX.

WindowsCodecs.dll: библиотека, которая предоставляет низкоуровневую поддержку для изображений в WPF.

Еще ниже собственно находятся компоненты операционной системы и DirectX, которые производят визуализацию компонентов приложения, либо выполняют прочую низкоуровневую обработку. В частности, с помощью низкоуровневого интерфейса Direct3D, который входит в состав DirectX, происходит трансляция.

Здесь также на одном уровне находится библиотека user32.dll. И хотя выше говорилось, что WPF не использует эту библиотеку для рендеринга и визуализации, однако для ряда вычислительных задач (не включающих визуализацию) данная библиотека продолжает использоваться.

XAML (eXtensible Application Markup Language) – язык разметки, используемый для инициализации объектов в технологиях на платформе .NET. Применительно к WPF (а также к Silverlight) данный язык используется прежде всего для создания пользовательского интерфейса декларативным путем. Хотя функциональность XAML только графическими интерфейсами не ограничивается: данный язык также используется в технологиях WCF и WF, где он никак не связан с графическим интерфейсом. То есть его область шире. Применительно к WPF мы будем говорить о нем чаще всего именно как о языке разметки, который позволяет создавать декларативным путем интерфейса, наподобие HTML в веб-программировании.

XAML [12] – не является обязательной частью приложения, можно обходиться без него, создавая все элементы в файле связанного с ним кода на языке C#. Однако использование XAML все-таки несет некоторые преимущества:

- возможность отделить графический интерфейс от логики приложения, благодаря чему над разными частями приложения могут относительно автономно работать разные специалисты: над интерфейсом – дизайнеры, над кодом логики – программисты;

- компактность, понятность, код на XAML относительно легко поддерживать.

При компиляции приложения в Visual Studio код в xaml-файлах также компилируется в бинарное представление кода xaml, которое называется BAML (Binary Application Markup Language). И затем код baml встраивается в финальную сборку приложения - exe или dll-файл.

AngleSharp [1] – это библиотека .NET, которая дает вам возможность разбирать гипертекст, такой как HTML, SVG и MathML. XML без проверки также поддерживается библиотекой. Важным аспектом AngleSharp является то, что CSS также может быть проанализирован. Парсер построен на официальной спецификации W3C. Это создает совершенно переносимое DOM-представление HTML5 [15] данного исходного кода. Также текущие функции, такие как querySelector или querySelec-torAll, работают для обхода дерева.

3.2. Система управления версиями

Для хранения исходного кода используется система управления версиями (VCS, англ. Version Control System) [2]. Данная система предназначена для удобства работы с меняющейся информацией, помогая осуществлять хранение нескольких версий одного документа. GitHub [5] – сервис бесплатного хранилища кода.

3.3. Взаимодействие компонентов

При запуске приложения, инициализируется страница Main-Window.xaml (рис. 5 и рис. 6). На данной странице отображаются все основные элементы для взаимодействия с программой:

- кнопка меню;
- кнопка обновить;
- кнопка календарь;
- кнопка мониторинг;
- кнопка закрыть;
- список конференций.

```
<Window x:Name="MainWindowFQW" x:Class="FQW.MainWindow"
        xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
        xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
        xmlns:d="http://schemas.microsoft.com/expression/blend/2008"
        xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"
        xmlns:local="clr-namespace:FQW"
        mc:Ignorable="d"
        Title="FQW" Height="600" Width="800" WindowStartupLocation="CenterScreen"
Foreground="{x:Null}" BorderThickness="1,1,1,5" BorderBrush="#FF293955" Back-
ground="White" ShowInTaskbar="False" WindowStyle="None" Margin="1" Resize-
Mode="NoResize">
    <Grid x:Name="MainBackGround" Background="#FFF6F6F6" Margin="0">
        <Grid x:Name="TopPanel" MouseLeftButtonDown="MainWindowFQW_MouseLeftBut-
tonDown" Background="#FF293955" Margin="0,0,0,499">
            <Button x:Name="showSettings" HorizontalAlignment="Left" Mar-
gin="19,35,0,35" Width="25" BorderThickness="0" Click="button_Click_1" Border-
Brush="{x:Null}" Foreground="{x:Null}" ToolTip="Настройки">
                <Button.Background>
                    <ImageBrush ImageSource="Sprites\menuMain.png" Stretch="Uni-
form"/>
                </Button.Background>
            </Button>
            <Label x:Name="LogoName" Content="FQW" FontFamily="Ebrima" FontSize="24"
FontWeight="Bold" Foreground="White" Margin="55,29,0,29" HorizontalAlignment="Left"
Width="70" VerticalContentAlignment="Center" HorizontalContentAlignment="Center"
Padding="0"/>
            <Button x:Name="Refresh" Content="" Margin="0,21,125,21" BorderThick-
ness="0" Click="Refresh_Click" HorizontalAlignment="Right" Width="35" Border-
Brush="{x:Null}" Foreground="White" FontSize="8" FontFamily="Arial Black"
ToolTip="Обновить" Height="35">
                <Button.Background>
                    <ImageBrush ImageSource="Sprites\refresh.png" Stretch="Uni-
form"/>
                </Button.Background>
            </Button>
            <Button x:Name="GoDate" Content="" Margin="0,29,30,31" HorizontalAlign-
ment="Right" Width="35" Click="GoDate_Click" Padding="0" BorderBrush="{x:Null}"
Foreground="White" BorderThickness="0" FontSize="8" FontFamily="Arial Black"
ToolTip="Календарь" Height="35">
                <Button.Background>
                    <ImageBrush ImageSource="Sprites\kalendar.png" Stretch="Uni-
formToFill"/>
                </Button.Background>
            </Button>
            <Button x:Name="closeButton" Content="" HorizontalAlignment="Right"
Height="15" VerticalAlignment="Top" Width="15" BorderBrush="{x:Null}" Fore-
ground="{x:Null}" HorizontalContentAlignment="Center" VerticalContentAlignment="Cen-
ter" Padding="0" BorderThickness="0" Click="closeButton_Click" RenderTrans-
formOrigin="1.5,1.5" Margin="0" ToolTip="Закрыть">
                <Button.Background>
                    <ImageBrush ImageSource="Sprites\close.png" Stretch="Uniform"/>
                </Button.Background>
            </Button>
            <Button x:Name="monitoring" Content="" Margin="0,30,80,30" BorderThick-
ness="0" HorizontalAlignment="Right" Width="35" Click="monitoring_Click">
```

Рис. 5. Страница MainWindowFQW.xaml

```
                <Button.Background>
                    <ImageBrush ImageSource="Sprites\monitoring.png"/>
                </Button.Background>
            </Button>
            <ComboBox x:Name="comboBox" HorizontalAlignment="Left" Mar-
gin="449,30,0,30" Width="170" BorderThickness="1,1,1,5" BorderBrush="White" Back-
ground="White" Foreground="White" SelectedIndex="0"/>
        </Grid>
        <Grid x:Name="mainContentPanel" HorizontalAlignment="Left" Height="485" Mar-
gin="4,104,0,0" VerticalAlignment="Top" Width="790" Background="#FFF6F6F6" Visibil-
ity="Hidden">
            <Label x:Name="label" Content="" Margin="5,10,5,0" VerticalAlign-
ment="Top" Background="#FF293A56" Foreground="White" Height="45" VerticalCon-
tentAlignment="Center" FontFamily="Arial Black" FontSize="16"/>
            <RichTextBox x:Name="richTextBox" Margin="30,65,30,25" ScrollViewer.Can-
ContentScroll="True" IsReadOnly="True" BorderThickness="0,5,0,0" Background="White"
BorderBrush="#FF293955" FontFamily="Arial" FontSize="18">
                <FlowDocument>
                    <Paragraph Padding="1.25,2,0,2">
                        <Run Text="RichTextBox"/>
                    </Paragraph>
                </FlowDocument>
            </RichTextBox>
            <Label x:Name="label1" Content=" " Margin="245,461,245,-2" VerticalA-
lignment="Center" HorizontalAlignment="Center" Width="300" HorizontalContentAlign-
ment="Center" VerticalContentAlignment="Center"/>
        </Grid>
        <Grid x:Name="settingPanel" HorizontalAlignment="Left" Height="594" Verti-
calAlignment="Top" Width="264" Background="#CC293955" Visibility="Hidden">
            <Label x:Name="LogoName_Copy" Content="FQW" FontFamily="Ebrima" Font-
Size="24" FontWeight="Bold" Foreground="White" Margin="58,26,0,528" Back-
ground="{x:Null}" HorizontalAlignment="Left" Width="99"/>
            <Button x:Name="hideSetting" Content="" HorizontalAlignment="Left" Mar-
gin="19,35,0,0" VerticalAlignment="Top" Width="25" BorderThickness="0" Border-
Brush="{x:Null}" Foreground="Black" Height="25" Click="HideSetting_Click" Horizon-
talContentAlignment="Center" VerticalContentAlignment="Center">
                <Button.Background>
                    <ImageBrush ImageSource="Sprites\menuMain.png"/>
                </Button.Background>
            </Button>
            <Button x:Name="settingOne" Content="Стили" Margin="10,120,10,0"
Height="45" BorderThickness="1,1,1,5" FontSize="16" FontFamily="Arial Black" Verti-
calAlignment="Top" Foreground="White" Padding="1" BorderBrush="White" Back-
ground="#FF525F75"/>
            <Button x:Name="settingTwo" Content="Частота обновления" Mar-
gin="10,170,10,0" Height="45" BorderThickness="1,1,1,5" FontSize="16" FontFam-
ily="Arial Black" VerticalAlignment="Top" Foreground="White" Padding="1" Border-
Brush="White" Background="#FF525F75"/>
        </Grid>
    </Grid>
</Window>
```

Рис. 6. Страница MainWindowFQW.xaml

3.4. Интерфейс

Макет главного экрана с названием всех элементов на нем, показан на рис. 7.

Макет экрана «Календарь» представлен на рис. 8.

Рис. 7. Макет главного экрана

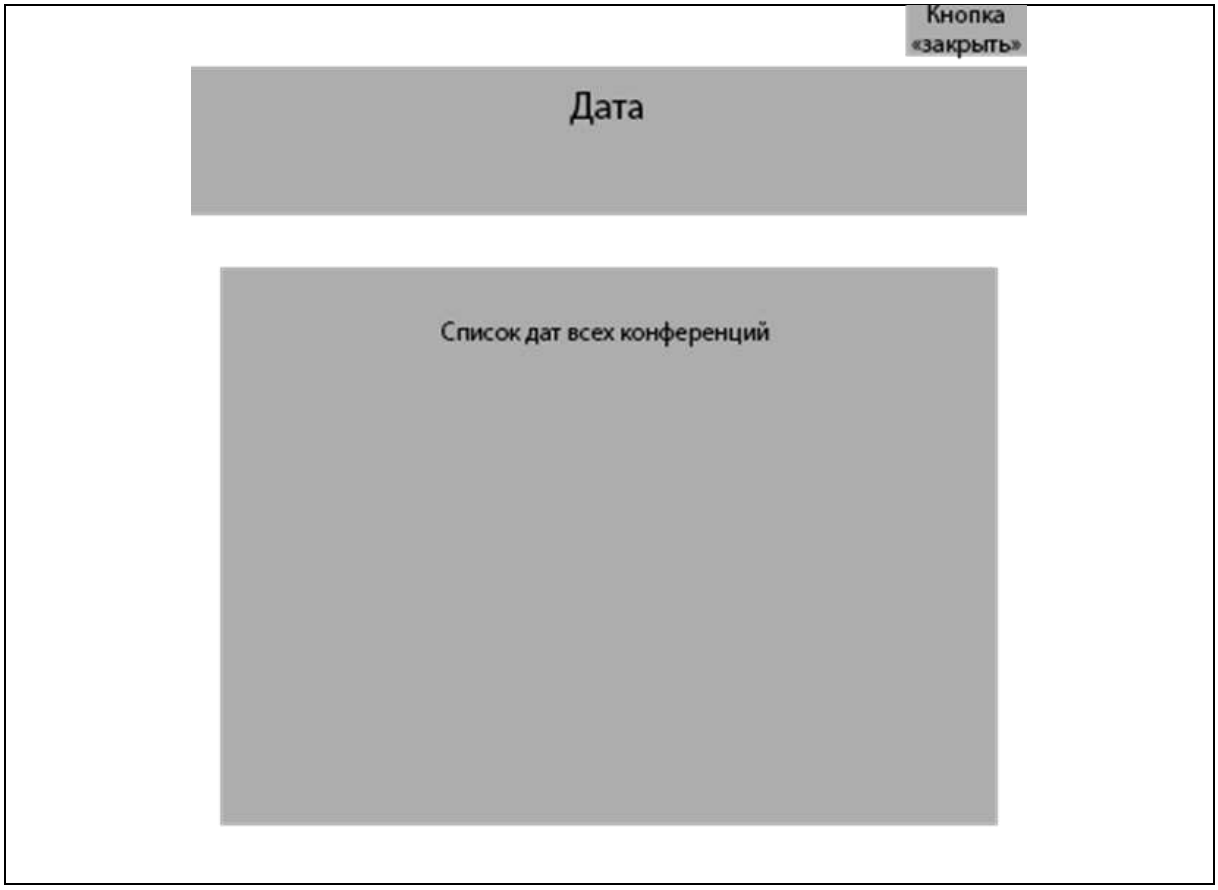

Рис. 8. Макет экрана «Календарь»

3.5. Вывод

В данном разделе были описаны особенности разработки программной системы для автоматизированного мониторинга предстоящих научных конференций, рассмотрены основные инструменты, используемые при реализации, а также представлены компоненты, из которых состоит программная система.

4. ТЕСТИРОВАНИЕ ПРИЛОЖЕНИЯ

Тестирование программного обеспечения – проверка соответствия между реальным и ожидаемым поведением программы, осуществляемая на конечном наборе тестов, выбранном определенным образом. В более широком смысле, тестирование – это одна из техник контроля качества, включающая в себя активности по планированию работ, проектированию тестов, выполнению тестирования и анализу полученных результатов.

Тестовый случай – это артефакт, описывающий совокупность шагов, конкретных условий и параметров, необходимых для проверки реализации тестируемой функции или ее части.

Разрабатываемая программная система прошла функциональное тестирование, т.е. проверка способности ПО в определенных условиях решать задачи, нужные пользователям.

Для тестирования программной системы был создан локальный сервер (рис. 9).

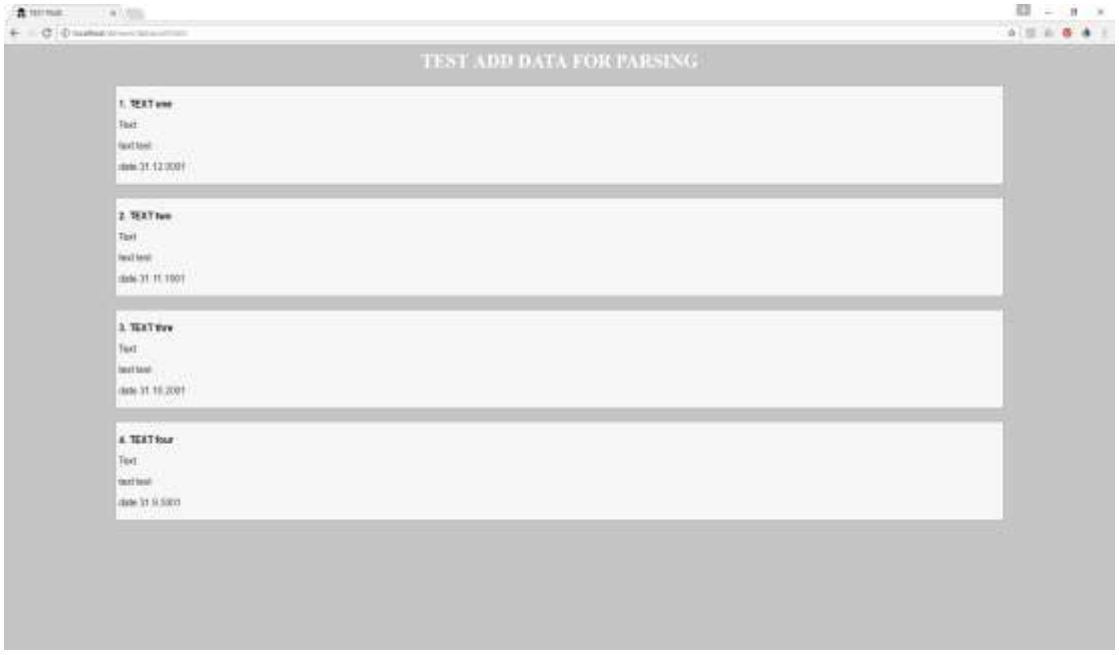

Рис. 9. Тестовая страница локального сервера

При нажатии на кнопку «Обновить» программа загружает данные с тестовой станицы, и отображает эти данные на главном экране программы (рис. 10).

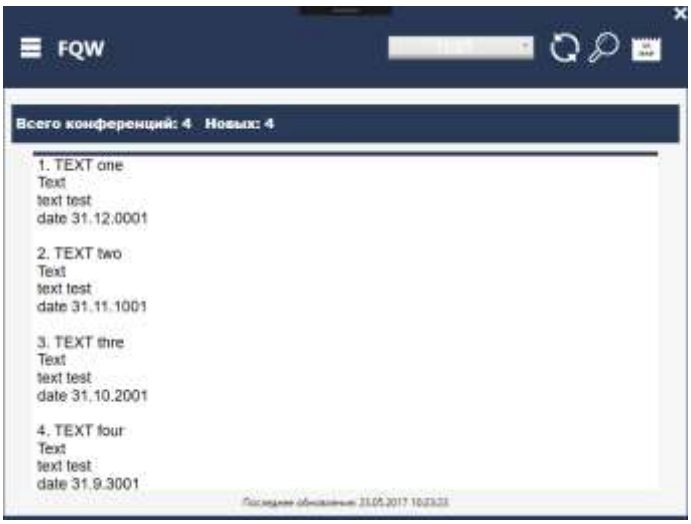

Рис. 10. Главный экран

При добавлении новых данных на тестовую страницу данные на главном экране программы обновляются (рис. 11, рис. 12).

Рис. 11. Тестовая страница с добавленными данными

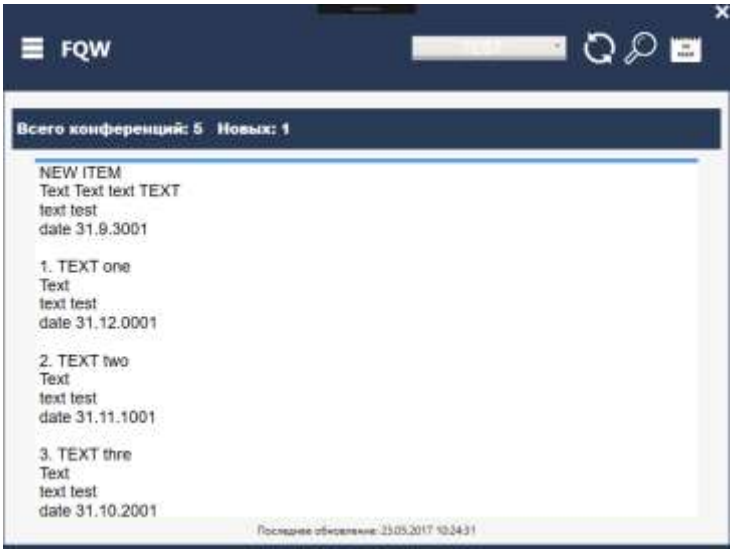

Рис. 12. Главный экран с обновленными данными

Таким образом, при занесении новых данных, данные также обновляются в программной системе.

Так же было проведено тестирование на корректность загрузки данных с одного из источников, с сайта konferencii.ru. Результаты представлены на рис. 13 и рис. 14.

Рис. 13. Новые конференции на сайте konferencii.ru

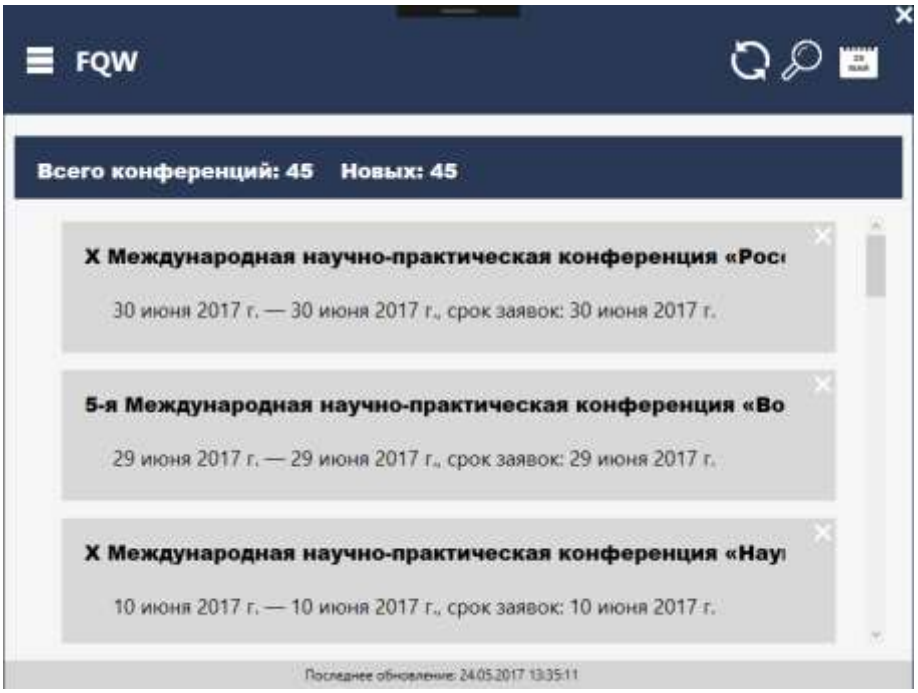

Рис. 14. Обновленный список конференций в программе

В результате проведенного тестирования было установлено соответствие разработанного приложения исходным функциональным требованиям.

4.1. Функциональное тестирование

Результаты функционального тестирования полностью совпали с ожидаемыми результатами и приведены в табл. 2.

Табл. 2. Результат функционального тестирования

№	Название теста	Шаги	Ожидаемый результат	Тест пройден?
1.	Обновить список конференций	1. Пользователь нажимает кнопку «Обновить»	Данные конференций обновлены, и выведены на главный экран	Да
2.	Просмотреть календарь дат	1. Пользователь нажимает на кнопку «Календарь»	Автоматически открывается окно календаря, в котором показаны даты предстоящих конференций	Да
3.	Изменить частоту обновления списка конференций	1. Пользователь нажимает кнопку «Меню» 2. Нажимает на кнопку «Частота обновления» 3. Выбирает пункт «каждые 10 мин.»	Каждые 10 минут программа автоматически обновляет список конференций	Да
4.	Изменить частоту обновления списка конференций	1. Пользователь нажимает кнопку «Меню» 2. Нажимает на кнопку «Частота обновления» 3. Выбирает пункт «каждые 30 мин.»	Каждые 30 минут программа автоматически обновляет список конференций	Да
5.	Переход на web страницу конференции	1. Пользователь выбирает из списка конференцию 2. Нажимает на выбранную конференцию	Автоматически открывается браузер, в котором загружена страница выбранной конференции	Да
6.	Изменить источник мониторинга	1. Пользователь нажимает на кнопку «Меню» 2. Нажимает на кнопку источник 3. Выбирает источник Konferencii.ru	Автоматически загрузится информация о конференциях с выбранного источника	Да

ЗАКЛЮЧЕНИЕ

Разрабатываемая программная система для автоматизированного мониторинга предстоящих научных конференций является актуальной и перспективной.

В рамках работы была разработана программная система для автоматизированного мониторинга предстоящих научных конференций.

Эта программная система поможет:

- быть в курсе предстоящих конференций;
- не пропустить важную конференцию;
- вовремя оформить нужные документы для конференции;
- следить за интересующими конференциями.

Для достижения поставленной цели были решены следующие задачи:

- выполнен анализ предметной области;
- выполнено проектирование системы;
- реализована система;
- проведено тестирование системы.

ЛИТЕРАТУРА

1. AngleSharp. [Электронный ресурс] URL: https://github.com/FlorianRappl/AngleSharp (дата обращения: 01.03.2017).

2. Chacon S, Straub B. Pro Git, 2nd ed. – USA: Apress, 2014. – 608 p.

3. CsQuery - C# jQuery Port for .NET 4. [Электронный ресурс] URL: https://github.com/jamietre/CsQuery (дата обращения: 01.03.2017).

4. Fizzler. [Электронный ресурс] URL: https://code.google.com/archive/p/fizzler/ (дата обращения: 01.03.2017).

5. Git - GitHub is a development platform inspired by the way you work. [Электронный ресурс] URL: https://github.com/ (дата обращения: 01.03.2017).

6. Grune D. Parsing Techniques: A Practical Guide 2nd ed. – USA: Springer, 2007. – 662 p.

7. Html Agility Pack. [Электронный ресурс] URL: https://htmlagilitypack.codeplex.com/ (дата обращения: 01.03.2017).

8. Johnson B. Professional Visual Studio 2015, 1st ed. – USA: Wiley / Wrox, 2015. – 104 p.

9. Regex. [Электронный ресурс] URL: https://msdn.microsoft.com/library/hs600312.aspx (дата обращения: 01.03.2017).

10. WebSvodka - это инструмент автоматического контроля изменений интернет-страниц, который позволяет первым узнавать важную информацию и экономит время. [Электронный ресурс] URL: http://www.websvodka.ru/ (дата обращения: 20.05.2017).

11. Арлоу Д., Нейштадт А. UML 2 и Унифицированный процесс. Практический объектно-ориентированный анализ и проектирование. – М.: Символ- Плюс, 2007. – 624 с.

12. Арчер Т. Основы C#. – М.: Русская редакция, 2001. – 448 с.

13. Конференции. [Электронный ресурс] URL: http://bit.sp.susu.ru (дата обращения: 01.03.2017).

14. Конференции.ru - информационная поддержка научных мероприятий. [Электронный ресурс] URL: http://konferencii.ru/ (дата обращения: 01.03.2017).

15. Лабберс П., Олберс Б., Салим Ф. HTML5 для профессионалов: мощные инструменты для разработки современных веб-приложений – Pro HTML5 Programming: Powerful APIs for Richer Internet Application Development. - М.: «Вильямс», 2011. – С. 272.

16. Леоненков А. Самоучитель UML 2. – СПб.: БХВ-Петербург, 2007. - 576 с.

17. Мак-Дональд М. WPF: Windows Presentation Foundation в .NET 4.5 с примерами на C# 5.0 для профессионалов, 4-е изд. – М.: «Вильямс», 2013. – 1024 с.

18. Руководство по программированию на C#. [Электронный ресурс] URL: https://msdn.microsoft.com/ru-ru/library/67ef8sbd.aspx (дата обращения: 20.05.2017).

19. Скит Д. C# для профессионалов: тонкости программирования, 3-е издание, новый перевод. – М.: «Вильямс», 2014. – 608 с.

20. Фримен А. ASP.NET MVC 4 с примерами на C# 5.0 для профессионалов, 4-е изд. – Pro ASP.NET MVC 4, 4th ed. – USA: «Вильямс», 2013. – 688 с.

21. Шилдт Г. C# 4.0. Полное руководство. – М.: Издательский дом «Вильямс», 2015. – 1056 с.