Министерство образования и науки Российской Федерации
ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ
ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
«САРАТОВСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ
ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ ИМЕНИ Н.Г.ЧЕРНЫШЕВСКОГО»

Кафедра компьютерной физики
и метаматериалов на базе Саратовского филиала
Института радиотехники и электроники
им. В.А. Котельникова РАН

Система автоматического сбора и обработки информации в сети Интернет

АВТОРЕФЕРАТ БАКАЛАВРСКОЙ ДИПЛОМНОЙ РАБОТЫ

студента 4 курса 431 группы

направления 03.03.02 «Физика» физического факультета

Ковина Станислава Сергеевича

Научный руководитель

к. ф.-м. н., доцент _____ А. С. Ремизов

Зав. кафедрой

д. ф.-м. н., профессор _____ В. М. Аникин

Саратов 2016 год

Актуальность работы. Системы автоматизации используются очень давно в различных отраслях и сферах деятельности. Такие системы позволяют увеличить скорость выполнения того или иного процесса.

Технологии не стоят на месте, все постоянно совершенствуется, оптимизируется. Автоматизированные системы позволяют увеличивать производительность, уменьшать трудозатраты, и много других плюсов.

Актуальность использования описанных в данной работе технологий, сложно переоценить. В наше время, в области интернет технологий наблюдается стремительный прогресс. Увеличивается скорость соединения, сокращаются потери при передаче данных до минимального, а самое главное, безопасность в сети интернет достигла высокого уровня. В связи с этими критериями, мы можем большую часть видов деятельности и процессов, перенести в интернет. Это позволит избавиться от многих повседневных проблем, в числе которых необходимость перемещения из дома, сбора информации по той или иной деятельности в бумажном виде. Находясь дома, достаточно открыть браузер, собрать всю необходимую информацию в сети интернет, и совершить любую операцию необходимую нам, будь то оплата услуг или получение выписок.

В этой работе мы рассмотрим технологии с помощью которых разрабатывают веб-приложения в сети интернет. Так же рассмотрим конкретный пример сервиса по автоматическому сбору и обработки информации.

Целью данной работы ставится изучение методов автоматизации процессов в сети интернет. А также, создание веб-приложения в качестве примера, для наглядного рассмотрения применения описанных в данной работе технологий.

В **задачи** работы входит:

1) Изучение основных технологий, применяемых на практике при реализации веб-приложений;

2) Рассмотрение принципов работы синтаксических анализаторов данных;
3) Реализация приложения сбора и обработки информации в сети интернет;

Структура и объем работы. Выпускная квалификационная работа изложена на 41 странице, состоит из введения, 3 разделов (1. Форматы и протоколы обмена информацией; 2. Используемые технологии; 3. Система автоматического сбора и обработки информации) и заключения. Библиографический список включает 11 наименований. Текст иллюстрирован 8 рисунками и 3 таблицами, демонстрирующие основные технологии.

Основное содержание работы

Структура веб приложения

Веб-приложение состоит из клиентской и серверной частей, тем самым реализуя технологию «клиент-сервер» .

Клиентская часть реализует пользовательский интерфейс, формирует запросы к серверу и обрабатывает ответы от него.

Серверная часть получает запрос от клиента, выполняет вычисления, после этого формирует веб-страницу и отправляет её клиенту по сети с использованием протокола HTTP.

Само веб-приложение может выступать в качестве клиента других служб, например, базы данных или другого веб-приложения, расположенного на другом сервере. Ярким примером веб-приложения является система управления содержимым статей Википедии: множество её участников могут принимать участие в создании сетевой энциклопедии, используя для этого браузеры своих операционных систем (будь то Microsoft Windows, GNU/Linux или любая другая операционная система) и не загружая дополнительных исполняемых модулей для работы с базой данных статей.

В настоящее время подход к разработке веб-приложений, основанный на асинхронных Ajax-запросах, стал практически стандартом. При использовании Ajax страницы веб-приложения не перезагружаются целиком, а лишь догружают необходимые данные с сервера, что делает их более интерактивными и производительными.

Также в последнее время набирает большую популярность технология WebSocket, которая не требует постоянных запросов от клиента к серверу, а создает двунаправленное соединение, при котором сервер может отправлять данные клиенту, без запроса от последнего. Таким образом появляется возможность динамически управлять контентом в режиме реального времени.

Для создания веб-приложений на стороне сервера используются разнообразные технологии и любые языки программирования, способные осуществлять вывод в стандартную консоль.

Клиентская часть

Клиентская часть веб приложения - это графический интерфейс. Это то, что вы видите на странице. Графический интерфейс отображается в браузере. Пользователь взаимодействует с веб-приложением именно через браузер, кликая по ссылкам и кнопкам.

Клиентская часть состоит из стека таких технологий, как HTML, JavaScript, CSS. HTML и CSS позволяют создать красиво оформленное, удобное для пользователя веб приложения. Тогда как JavaScript добавляет динамичности интерфейсу.

Серверная часть

Серверная часть веб-приложения - это программа или скрипт на сервере, обрабатывающая запросы пользователя (точнее, запросы браузера). При каждом переходе пользователя по ссылке браузер отправляет запрос к серверу. Сервер обрабатывает этот запрос, вызывая скрипт, который

формирует веб-страницу, описанную языком HTML, и отсылает клиенту по сети. Браузер тут же отображает полученный результат в виде очередной веб-страницы.

Самые популярные веб-сервера - Apache и Nginx. Одно из самых существенных отличий между Apache и Nginx состоит в том, как они обрабатывают соединения и отвечают на различные виды трафика.

Apache предоставляет несколько модулей мультипроцессинга (multi-processing modules, MPM), которые отвечают за то как запрос клиента будет обработан. Это позволяет администраторам определять политику обработки соединений.

Nginx изначально был спроектирован на базе асинхронных неблокирующих event-driven алгоритмов. Nginx создает процессы-воркеры каждый из которых может обслуживать тысячи соединений. Каждое соединение, обрабатываемое воркером, помещается в event loop вместе с другими соединениями. В этом цикле события обрабатываются асинхронно, позволяя обрабатывать задачи в неблокирующей манере. Когда соединение закрывается оно удаляется из цикла. Этот подход к обработке соединений позволяет Nginx'у невероятно масштабироваться при ограниченных ресурсах.

Так же распространенной схемой использования является размещение Nginx перед Apache в качестве реверс-прокси. В такой конфигурации Nginx называют фронтендом, а Apache — бэкендом. При таком подходе Nginx будет обслуживать все входящие запросы клиентов, и мы получим выигрыш из-за его возможности обрабатывать множество конкурентных запросов.

Nginx будет самостоятельно обслуживать статический контент, а для динамического контента, например, для запросов к PHP-страницам, будет передавать запрос к Apache, который будет рендерить страницу, возвращать ее Nginx'у, а тот в свою очередь будет передавать ее пользователю.

Эта конфигурация позволяет горизонтально масштабировать приложение: вы можете установить несколько бэкенд серверов за одним

фронтендом и Nginx будет распределять нагрузку между ними, увеличивая тем самым отказоустойчивость приложения.

Скрипт-серверные языки программирования

В качестве серверных языков в основном используются PHP, Python, Ruby, CGI-скрипты.

CGI - Common Gateway Interface является стандартом интерфейса, который служит для связи внешней программы с веб-сервером. Интерфейс разработан таким образом, чтобы можно было использовать любой язык программирования, который может работать со стандартными устройствами ввода/вывода.

PHP – скриптовый язык предназначенный для разработки веб-приложений. В области веб-программирования, в частности серверной части, PHP — один из популярных сценарных языков (наряду с JSP, Perl и языками, используемыми в ASP.NET). Популярность в области построения веб-сайтов определяется наличием большого набора встроенных средств для разработки веб-приложений.

Python — высокоуровневый язык программирования общего назначения, ориентированный на повышение производительности разработчика и читаемости кода. Синтаксис ядра Python минималистичен. Python поддерживает несколько парадигм программирования, в том числе структурное, объектно-ориентированное, функциональное, императивное и аспектно-ориентированное. Основные архитектурные черты — динамическая типизация, автоматическое управление памятью, полная интроспекция, механизм обработки исключений, поддержка многопоточных вычислений и удобные высокоуровневые структуры данных.

Ruby — динамический, рефлективный, интерпретируемый высокоуровневый язык программирования для быстрого и удобного объектно-ориентированного программирования. Язык обладает независимой от операционной системы реализацией многопоточности, строгой

динамической типизацией, сборщиком мусора и многими другими возможностями. По особенностям синтаксиса он близок к языкам Perl и Eiffel, по объектно-ориентированному подходу — к Smalltalk. Также некоторые черты языка взяты из Python, Lisp, Dylan и Клу.

Архитектура REST

REST (Representational state transfer) – это стиль архитектуры программного обеспечения для распределенных систем, который, как правило, используется для построения веб-служб. Термин REST был введен в 2000 году Роем Филдингом, одним из авторов HTTP-протокола. Системы, поддерживающие REST, называются RESTful-системами.

В общем случае REST является очень простым интерфейсом управления информацией без использования каких-то дополнительных внутренних прослоек. Каждая единица информации однозначно определяется URL – это значит, что URL по сути является первичным ключом для единицы данных. Т.е., например, третья книга с книжной полки будет иметь вид /book/3, а 35 страница в этой книге — /book/3/page/35. Отсюда и получается строго заданный формат. Причем совершенно не имеет значения, в каком формате находятся данные по адресу /book/3/page/35 – это может быть и HTML, и отсканированная копия в виде jpeg-файла, и документ Microsoft Word.

Как происходит управление информацией сервиса – это целиком и полностью основывается на протоколе передачи данных. Наиболее распространенный протокол конечно же HTTP. Для HTTP действие над данными задается с помощью методов: GET (получить), PUT (добавить, заменить), POST (добавить, изменить, удалить), DELETE (удалить). Таким образом, действия CRUD (Create-Read-Updtae-Delete) могут выполняться как со всеми 4-мя методами, так и только с помощью GET и POST.

Вот как это будет выглядеть на примере:

GET /book/ — получить список всех книг

GET /book/3/ — получить книгу номер 3

PUT /book/ — добавить книгу (данные в теле запроса)

POST /book/3 – изменить книгу (данные в теле запроса)

DELETE /book/3 – удалить книгу

Как видно, в архитектура REST очень проста в плане использования. По виду пришедшего запроса сразу можно определить, что он делает, не разбираясь в форматах (в отличие от SOAP, XML-RPC). Данные передаются без применения дополнительных слоев, поэтому REST считается менее ресурсоемким, поскольку не надо анализировать запрос чтобы понять, что он должен сделать и не надо переводить данные из одного формата в другой.

Синтаксический анализ данных

Синтаксический анализ, или "парсинг" (жарг., от англ. parsing) - процесс сопоставления линейной последовательности лексем естественного или формального языка с его формальной грамматикой. Результатом как правило является дерево разбора (синтаксическое дерево).

Синтаксический анализатор (жарг. па́рсер) — это программа или часть программы, выполняющая синтаксический анализ.

Все, что имеет синтаксис, поддается автоматизации. В нашей задаче автоматизируется анализ содержимого интернет-страниц. Исходный текст анализируется, превращаясь в структурированные информационные блоки, которые сохраняются в базе данных, каждый блок обрабатываются как отдельная сущность.

Используется связка следующих технологий:

- **Регулярные выражения** –инструмент синтаксического анализа выражений, соответствующих небольшим формальным грамматикам. Задачи, связанные с синтаксическим разбором в прикладном программировании встречаются довольно часто, и решать некоторые из них без использования сервисов поддержки регулярных выражений крайне трудоемко;

- **PHPQuery** – библиотека для парсинга, аналог jQuery на PHP. Автор – польский программист Тобиас Судник. Первоначально разработал небольшой парсер для своих нужд - это был сборщик рецензий на фильмы. Несложный поисковый робот со временем преобразовался в одно из самых лучших средств синтаксического анализа. Основанная на DOM, библиотека является одной из самых быстрых. PHPQuery работает с селекторами, атрибутами, Ajax-ом, событиями и пр.;

- **cURL** - свободная, кроссплатформенная служебная программа командной строки, позволяющая взаимодействовать с множеством различных серверов по множеству различных протоколов с синтаксисом URL. cURL дает возможность отправлять запросы, получать ответы от сервера.

Приложение для автоматического сбора информации

Рассмотренные технологии я применяю по месту работы, изученные принципы автоматизации пригодились при решении одной практических из задач. А именно, при разработке системы автоматического получения банковских гарантий, не посещая банки.

Сервис банковских гарантий - уникальный финансовый инструмент для юридических лиц, позволяющий участникам закупок, проводимых в соответствии с Федеральным законом №44-ФЗ, получать для предоставления государственным и муниципальным заказчикам обеспечение исполнения контракта в виде банковской гарантии.

Сервис призван облегчить для них процедуру получения гарантийных документов для обеспечения заявок на участие в конкурсах и тендерах.

Формат электронных торгов позволяет участникам в полной мере оценить преимущества продукта: возможность удобной и качественной работы в удаленном режиме и передачи в электронном виде данных, необходимых для получения гарантии банка. Это снижает издержки компаний и ускоряет процесс получения необходимых документов.

Таких сервисов достаточное количество в России, главное преимущество нашей системы – автоматизация процессов.

ВЫВОДЫ

Автоматизированные системы в данное время очень актуальны. Любой процесс можно автоматизировать. Нужно лишь знать, как работает система изнутри и с помощью каких технологий можно реализовать автоматизацию.

В данной работе я описал доступные средства, с помощью которых можно автоматизировать процессы в сети интернет. В теоретической части работы рассмотрены базовые принципы функционирования веб-приложений, форматы и протоколы передачи данных, архитектура построения распределенных систем и технологии синтаксического анализа данных. В практической части приведено описание разработанной системы сбора и обработки информации.