МИНОБРНАУКИ РОССИИ
Федеральное государственное бюджетное образовательное учреждение
высшего образования
**«САРАТОВСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ
ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
ИМЕНИ Н. Г. ЧЕРНЫШЕВСКОГО»**
Кафедра математической кибернетики и компьютерных наук

РАЗРАБОТКА ПАРСЕРА УЧЕБНОГО ПЛАНА

АВТОРЕФЕРАТ БАКАЛАВРСКОЙ РАБОТЫ

Студентки 4 курса 411 группы
направления 02.03.02 — Фундаментальная информатика и информационные технологии
факультета КНиИТ
Суховой Надежды Валентиновны

Научный руководитель
зав. каф, к. ф.-м. н., доцент _____ С. В. Миронов

Заведующий кафедрой
к. ф.-м. н., доцент _____ С. В. Миронов

Саратов 2021

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

Актуальность данной работы обусловлена приоритетом цифровизации образования. Цифровизация, являясь новым этапом развития отечественного образования, охватывает все сферы деятельности общества, внося множество изменений в рабочий процесс и образовательную систему.

Приоритет цифровизации также отражен в указе президента России от 09.05.2017 №203 ББО «Стратегии развития информационного общества в РФ на 2017-2030 годы», где отмечается, что процесс цифровизации учебных процессов является приоритетной задачей.

Цифровизация в образовании подразумевает создание и внедрение электронных систем обучения. Подобный подход повышает доступность и удобство обучения. Сложность и глобальность цифровизации подразумевает целый комплекс мероприятий. Кроме обновления материальной инфраструктуры необходимо развивать цифровые программы и системы управления обучением.

Очередным этапом цифровизации учебного процесса Саратовского национального исследовательского государственного университета имени Н.Г. Чернышевского является создание электронных учебных программ на основе имеющейся базы данных. В этом должен помочь парсер учебных планов, который облегчит и сделает быстрее работу преподавателей по обработке учебных планов.

Для успешного внедрения парсера необходимо ознакомиться с уже имеющимися технологиями разработки и изучить их как с теоретической, так и с практической стороны.

Целью настоящей работы является создание парсера для учебного плана для использования его сайтом ipsilon.sgu.ru.

Были поставлены задачи:

- создать парсер, который из указанного файла с учебным планом в формате XLS будет находить необходимые данные, а именно: наименование дисциплины, название группы дисциплины, вид отчётности, количество контрольных работ по данной дисциплине и соответствующее ей количество зачётных единиц;
- добавить тестирование, чтобы проверить правильность работы парсера;
- ввести парсер в эксплуатацию.

1 Постановка задачи

1.1 Ipsilon

IpsilonUni — это система дистанционного обучения, разработанная для Саратовского Государственного Университета.

IpsilonUni был создан для выполнения следующих функций [1]:

- работа с учебными планами;
- фиксация балльно-рейтинговой системы;
- работа с учебно-методическими курсами и тестами;
- формирование портфолио обучающегося.

Цель данной работы — создание парсера, который необходим для внесения в структуру базы данных Ipsilon необходимых данных из предоставляемых ему учебных планов.

1.2 Стандарты учебных планов

Для учебных планов, как для любого официального документа, существуют свои определённые стандарты. На данный момент существует несколько государственных стандартов для образовательных учреждений: ФГОС, ФГОС3+, ФГОС3++.

1.2.1 ФГОС

ФГОС (федеральные государственные образовательные стандарты) включают в себя совокупность требований к образовательным учреждениям большинства уровней (от детских садов до курсов повышения квалификации) [2]. Эти требования представляют из себя следующие критерии, которые обязательны для применения:

- требования к соотношению объёма образовательной программы к её частям, соотношению обязательной части программы к части, которую формируют участники образовательной программы и т. д.;
- требования к условиям обучения по основным образовательным программам, в том числе к финансовым условиям, техническим и т. д.;
- требования к результатам обучения по основным образовательным программам.

1.2.2 ФГОС3+

ФГОС3+ — это стандарт для высших учебных заведений, требования которого действуют для бакалавриата, магистратуры и специалитета, и аспирантуры. Для бакалавриата предусмотрены два типа учебных программ: программа для академического и прикладного бакалавриата. При этом учитывается дистанционный вариант обучения и дистанционные образовательные технологии.

1.2.3 ФГОС3++

ФГОС3++ является дополнением стандартов ФГОС 3+, основным отличием стандартов является модернизация. В данном стандарте расписываются определённые предметы, которые должны преподаваться для всех направлений. Следовательно, стандарт устанавливал требования таким образом, что во всех ВУЗах на одинаковых направлениях преподавались одни и те же дисциплины.

1.3 Имеющаяся среда

Так как система разрабатывалась на языке Ruby с использованием фреймворка для работы с веб-приложениями Ruby on Rails, то парсер, как модуль, входящий в эту систему, так же будет написан на языке Ruby.

В шапке страницы можно увидеть кнопки стандартного функционала, доступного администраторам сайта, а ниже располагается основная информация учебного плана, в том числе и та, которую можно отредактировать: номера групп, сроки обучения и тип учебного плана. Так же присутствует опциональный шаблон для балльно-рейтинговой системы, который выставляет баллы для системы БАРС.

Под шапкой в описании учебного плана расположена кнопка «Обновить из файла», функционалом для которой и послужит в будущем парсер. Здесь можно отправить системе необходимый файл формата XLS, выбрать семестры, которые необходимо обновить из файла, а так же выбрать опцию очистки выбранных семестров. Ниже следует список дисциплин учебного плана.

В этой вкладке можно удалить как одну дисциплину, так и сам учебный период. К тому же, предоставляется возможность создать новую группу дисциплин в учебном плане и, если необходимо, скачать ведомости по дисциплинам данного учебного периода.

1.4 Существующие парсеры

На данный момент существуют следующие парсеры файлов Excel.

1.4.1 SAX2

Simple API for XML (SAX) — это программный интерфейс, который позволяет создавать парсеры для чтения файлов XML [3].

При использовании этого интерфейса файл разбивается на отдельные объекты при чтении, а дальше, используя структуру «дерево», строится структура документа. Таким образом можно вести работу с каждым узлом отдельно. Это удобно, но в то же время для реализации этого необходимы большие затраты памяти.

Также современные браузеры не имеют встроенной поддержки SAX, что усложняет работу с ними.

1.4.2 DOM parser

Как упоминалось ранее, DOM parser (Document Object Model) — парсер, схожий с SAX, но имеющий другую, отличную от его, модель (древовидную [4], в то время, как у SAX — событийная). В DOM для работы с каким-либо файлом, в отличие от SAX, необходимо загрузить весь исходный файл в единый объект с иерархией в виде дерева, с которым в будущем и предстоит работать.

Звучит удобно, ведь все необходимые данные можно легко и быстро достать из объекта, который уже есть под рукой, однако не стоит забывать, что данный подход будет удобен при работе с относительно небольшими файлами. В противном случае работа с файлом займет большое количество как времени, так и памяти [5].

1.4.3 StAX

StAX (Streaming API for XML) — потоковый синтаксический анализатор для чтения и записи XML файлов, который отличается от описанного выше SAX и основан на DOM.

Например, в отличие от SAX, этот парсер извлекает необходимые данные из xml файла, в то время как SAX — отправляет их [6]. К тому же, если в SAX можно сохранять данные только по мере их прочтения, то в StAX существует

такое понятие как «курсор», что значит, что пользователь может обратиться к необходимым ему данным в любой момент времени [7].

При этом в StAX присутствует та же древовидная структура, что и в DOM, которая предоставляет свободный доступ к данным, но при этом StAX основан на событиях. Это значит, что данные сохраняются в нем по мере прочтения документа.

2 Анализ технологий обработки данных для языка Ruby

2.1 Ruby

Ruby — это высокоуровневый язык программирования, на котором написана логика текущего проекта [8]. Удобство данного языка состоит в том, что он динамичен и прост в изучении. Его легко и приятно читать, в нём есть реализация многопоточности (независимо от операционной системы), сборщик мусора, строгая динамическая типизация и другие особенности. Язык обладает некоторыми чертами языков Python, Lisp, Dylan и Clu, а по объектно-ориентированному подходу он схож с языком Smalltalk [9].

Поскольку данный язык разрабатывался на Linux, то и работать с ним, соответственно, удобнее в данной операционной системе [10].

Для работы с проектом был необходим фреймворк для работы с веб-приложениями — Ruby on Rails [11]. Это программное обеспечение с открытым исходным кодом, написанным на языке Ruby с архитектурой MVC (Model-View-Controller). При этом Rails предоставляет множество библиотек, которые помогут в работе с парсингом XLM файлов.

2.1.1 Парсеры Ruby

Перед тем, как выбрать одну из библиотек Ruby, которая поможет в парсинге файлов, необходимо определить тип файлов, с которыми, собственно, предстоит работать. Файлы Excel разделяются на три типа: XLS, XSLM и XLT [12]. Каждый из них отличается от другого, в нашем случае парсинг будет работать с XLS файлами.

Для определённой работы можно выбрать как одну, так и несколько библиотек. В нашем случае идеально подходит библиотека Spreadsheet.

3 Разработка парсера учебного плана для сайта ipsilon.sgu.ru

3.1 Ubuntu

Linux — это название, объединяющее операционные системы, который образованы под влиянием Unix и основаны на одноименном ядре, и скомпилированные для них системных программ и библиотек, которые были разработаны в рамках проекта GNU [13].

Для более комфортной работы было необходимо установить в качестве второй ОС Ubuntu, наиболее известный дистрибутив Linux.

3.2 Подготовка системы к работе

После установки системы, необходимо подготовить ряд компонентов для дальнейшей работы над проектом. Как упоминалось выше, все необходимые программы и приложения находятся в репозитории, а значит всё, что необходимо сделать пользователю — ввести соответствующие команды в терминал.

В первую очередь необходимо установить утилиту sudo, которая обеспечивает контроль доступа [14]. Далее, с помощью установленной утилиты sudo, подготавливается конфигурация среды, а именно — устанавливаются все необходимые компоненты. В нашем случае, это система управления базами данных (PostgreSQL), система управления базами данных класса NoSQL (Redis) и язык программирования, на котором будет написан парсер (Ruby). В дальнейшей работе для включения парсера в работу самого сайта Ipsilon мне так же потребовалась nodejs — программная платформа для работы с JavaScript как с языком общего назначения.

После установки всех необходимых пакетов можно начинать работу с сайтом. Для этого потребуется утилита для работы с пакетами Ruby (или gem-ами) — bundler.

С помощью команды gem с репозитория RubyGems можно скачать все необходимые пакеты, что невероятно облегчает работу с их установкой.

Далее с помощью менеджера пакетов (npm, который входит в состав Node.js) устанавливается bower (менеджер для разработки с использованием JavaSctript) и копируется используемый для разработки пример базы данных в необходимую папку и, установив все необходимые базы данных, запускается проект.

3.3 Разработка парсера учебных планов

Работа с учебным планом является актуальной для продвижения функционала сайта ipsilon.sgu.ru. Проблема состояла в отсутствии парсера, поскольку преподавателям было необходимо вручную вбивать данные учебных планов.

Для облегчения работы преподавателей с сайтом был разработан парсер, который предоставляет возможность не расписывать учебный план вручную, а предоставить системе XLS файл, из которого в базу данных будет загружена вся необходимая информация, которая и будет отражена на сайте.

Необходимая информация загружается в хэш [15], ключи которого будут отображать информацию, хранящуюся в определённом элементе.

Для каждого семестра необходимо находить следующую информацию по каждой дисциплине:

- название дисциплины;
- название группы, к которой эта дисциплина принадлежит;
- вид отчётности по данной дисциплине;
- количество контрольных работ по данной дисциплине;
- количество зачётных единиц для дисциплины.

Чтобы загрузить необходимую информацию, производилась работа непосредственно с файлом учебного плана в формате XLS. Из него было необходимо загрузить данные из соотетствующих колонок. Возникла проблема в том, что учебные планы для очного и для заочного обучения отличались. В связи с этим, было решено создать два парсера и файл, определяющий форму обучения, который по титульному листу файла определял форму обучения и вызывал соответствующий парсер для работы с файлом.

При этом задача несколько осложнялась тем, что позиция дисциплин и необходимых колонок не указывалась чётко, поэтому было решено искать нужные ячейки путём сканирования всего учебного плана. Таким образом, алгоритм состоял в проходе по всем необходимым ячейкам в цикле и сохранении найденной в этих ячейках информации в нужном виде для сохранения в базу данных.

3.4 Файл, определяющий форму обучения

Файл проверяет две возможные ячейки, в которых может быть прописана форма обучения, а далее либо вызывается парсер для соответствующей формы

обучения и возвращается хэш, который является результатом работы парсера, либо возникает сообщение об ошибке, если файл невозможно обработать как стандартный учебный план.

Информация из хэша будет сохранена в базе данных и отображена на сайте [16].

3.5 Парсер учебного плана для очной формы обучения

Учебные планы у очного и заочного отделения различаются, но несмотря на это парсинг учебных планов строился по схожему алгоритму для этих двух форм обучения.

Алгоритм будет идти по колонкам семестров, сохраняя всю необходимую информацию в хэш. Чтобы идти от семестра к семестру, необходимо вычислить количество ячеек, которые будут содержаться между соседними семестрами, эта разница сохраняется в отдельную переменную.

Далее для очного формата необходимо сохранить номера колонок, в которых хранятся названия дисциплин, списки семестров, в которых предусмотрен экзамен, зачёт или зачёт с оценкой, списки семестров, в которых предусмотрены контрольные работы, а так же список семестров, в которых предусмотрена сдача реферата. Всё это повлияет на информацию, которая будет записана в форме отчётности по данной дисциплине.

Помимо формы отчётности, для каждой дисциплины необходимо сохранять количество зачётных единиц. Колонка со значениями з. е. ищется схожим образом.

В завершении, так как дисциплина «Курсовая работа» отсутствует, списки семестров, в котором необходима её сдача, сохраняются в отдельной колонке «КР», номер которой так же находится и сохраняется в отдельную переменную.

Отчётность складывается из последовательности букв:

— «э» для экзамена;
— «з» для зачёта;
— «о» для для зачёта с оценкой;
— «к» для контрольной работы;
— «ф» для реферата.

Стоит помнить, что если в одном семестре по одной дисциплине предусмотрен и зачёт, и экзамен, то в ведомость записываются две одноименные

дисциплины с разными формами отчётности.

Таким образом, сначала производится проверка формы отчётности. Если в ней присутствует и экзамен, и зачёт, тогда в хэш добавляются две отдельные дисциплины, у одно из которых будет форма отчётности «экзамен», а у другой — «зачёт».

После добавления всех дисциплин позиция смещается для работы со следующим семестром. Когда все дисциплины были сохранены, в хэш добавляются курсовые работы в соответствии со списком, в которых их сдача должна присутствовать.

После этого метод возвращает хэш как результат выполнения алгоритма.

3.6 Парсер учебного плана для заочной формы обучения

Для заочной формы обучения парсер работает схожим образом, за исключением ряда отличий, которые были необходимы, поскольку планы отличаются по шаблонам.

Первое отличие состоит в том, что у заочной формы обучения в шаблоне нет чёткой нумерации по семестрам. Во время одного курса у студентов заочной формы обучения проходит три сессии: установочная, зимняя и летняя. Первая не конвертируется в учебный план на портале Ipsilon, однако остальные две добавляются в учебный план.

Так, необходимо несколько переменных для определения количества ячеек между семестрами: одна для определения расстояния между зимней и летней сессией, а другая — для определения расстояния между летней и зимней сессиями через установочную.

В отличие от плана обучения для очного формата, здесь есть отдельная колонка, в которой расписана форма отчётности, так что в данном парсере её прописывать нет нужды.

При этом, в плане заочного обучения у дисциплин нет отдельной колонки, выделенной под з. е. Чтобы рассчитать зачётные единицы для дисциплины во время зимней сессии необходимо посчитать количество её часов во время зимней и установочной сессии. Каждые 36 часов — это одна з. е. Для рассчёта з. е. для дисциплины во время летней сессии достаточно поделить количество её часов на 36.

Количество контрольных работ считается, исходя из указанной формы отчётности. Если в семестре предусмотрено несколько контрольных работ по

данной дисциплине, то в форме отчётности указывается подстрока, состоящая из символов «к», чья длина равна количеству контрольных в семестре. Таким образом, чтобы подсчитать количество контрольных, нужно посчитать длину этой подстроки.

В остальном работа парсеров идёт схожим образом. Так же, как и в случае с очной формой обучения, алгоритм идёт по семестрам, записывая для каждой дисциплины, количество контрольных работ, группу и форму отчётности.

3.7 Введение парсера в эксплуатацию

Когда парсер был написан, всё, что осталось — это внедрение его в работу сайта. Для этого используются различные модели.

Для дисциплин используется модель Curriculum::Discipline. При этом дисциплина в данной модели не привязывается к семестру. Для связи дисциплины и семестра существует отдельная модель Curriculum::DisciplineSemester.

ЗАКЛЮЧЕНИЕ

В ходе написания данной работы был создан парсер учебного плана для системы дистанционного обучения ipsilon.sgu.ru.

Были выполнены задачи:

- создать парсер, который из указанного файла с учебным планом в формате XLS будет извлекать необходимые данные, а именно: наименование дисциплины, название группы дисциплины, вид отчётности, количество контрольных работ по данной дисциплине и соответствующее ей количество зачётных единиц;
- провести тестирование, чтобы проверить правильность работы парсера;
- ввести парсер в эксплуатацию.

Также были изучены библиотеки Ruby для работы с форматом XLS, продемонстрированы результаты их работы, а также была изучена среда разработки Ruby on Rails.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1 Образовательный портал [Электронный ресурс]. — URL: https://ipsilon.sgu.ru/users/sign_in (Дата обращения 12.05.2021). Загл. с экр. Яз. рус.

2 Что такое ФГОС и почему на них все равняются [Электронный ресурс]. — URL: https://externat.foxford.ru/polezno-znat/fgos (Дата обращения 20.05.2021). Загл. с экр. Яз. рус.

3 Типы парсеров XML [Электронный ресурс]. — URL: http://bourabai.kz/xml/parsers.htm (Дата обращения 12.05.2021). Загл. с экр. Яз. рус.

4 *Тидуэлл, Д.* XSLT. 2-Е ИЗДАНИЕ / Д. Тидуэлл. — Москва: Символ-Плюс, 2009.

5 SAX vs DOM [Электронный ресурс]. — URL: https://sites.google.com/site/sureshdevang/sax-vs-p (Дата обращения 17.05.2021). Загл. с экр. Яз. англ.

6 Parsing an XML File Using StAX [Электронный ресурс]. — URL: https://www.baeldung.com/java-stax (Дата обращения 18.05.2021). Загл. с экр. Яз. англ.

7 StAX Parser [Электронный ресурс]. — URL: https://www.examclouds.com/ru/java/web-services/stax (Дата обращения 18.05.2021). Загл. с экр. Яз. англ.

8 Ruby Installation [Электронный ресурс]. — URL: http://rubylearning.com/satishtalim/ruby_installation.html (Дата обращения 12.05.2021). Загл. с экр. Яз. англ.

9 Ruby on Rails [Электронный ресурс]. — URL: http://www.rubyonrails.ru/ (Дата обращения 12.05.2021). Загл. с экр. Яз. рус.

10 Official Ruby FAQ [Электронный ресурс]. — URL: https://www.ruby-lang.org/en/documentation/faq/3/ (Дата обращения 12.05.2021). Загл. с экр. Яз. англ.

11 Rails для начинающих [Электронный ресурс]. — URL: http://rusrails.ru/getting-started-with-rails (Дата обращения 12.05.2021). Загл. с экр. Яз. англ.

12 Поддерживаемые Excel форматы файлов [Электронный ресурс]. — URL: https://support.microsoft.com/ru-ru/office/%D0%BF%D0%BE%D0%B4%D0%B4%D0%B5%D1%80%D0%B6%D0%B8%D0%B2%D0%B0%D0%B5%D0%BC%D1%8B%D0%B5-excel-%D1%84%D0%BE%D1%80%D0%BC%D0%B0%D1%82%D1%8B-%D1%84%D0%B0%D0%B9%D0%BB%D0%BE%D0%B2-0943ff2c-6014-4e8d-aaea-b83d51d46247 (Дата обращения 12.05.2021). Загл. с экр. Яз. рус.

13 Что такое операционная система Linux: плюсы и минусы свободной платформы [Электронный ресурс]. — URL: https://te-st.ru/2017/11/01/linux-advantages-and-disadvantages/ (Дата обращения 15.05.2021). Загл. с экр. Яз. рус.

14 Пользователь root и sudo [Электронный ресурс]. — URL: https://help.ubuntu.com/kubuntu/desktopguide/ru/root-and-sudo.html (Дата обращения 16.05.2021). Загл. с экр. Яз. рус.

15 *Клеппман, М.* Высоконагруженные приложения. Программирование, масштабирование, поддержка / М. Клеппман. — Санкт-Петербург: Издательский Дом ПИТЕР, 2018.

16 Архитектура «клиент-сервер» [Электронный ресурс]. — URL: http://www.mstu.edu.ru/study/materials/zelenkov/ch_7_1.html (Дата обращения 12.05.2021). Загл. с экр. Яз. рус.