МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

федеральное государственное бюджетное образовательное учреждение

высшего образования

**«УЛЬЯНОВСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»**

Ульяновский государственный технический университет

Кафедра вычислительной техники

**Отчет**

по индивидуальному заданию в рамках дисциплины

«Проектирование информационного обеспечения САПР»

Выполнил:

П.С.Кондратьев

Приняли:

Г.П.Токмаков

В.Н.Негода

Ульяновск – 2020

1. **Цели и содержание индивидуального задания**

Целями индивидуального задания являются:

* изучение основ учебной дисциплины в контексте проекта создания средств автоматизации формирования индивидуальных планов преподавателей;
* развитие средств автоматизации организационного управления кафедры ВТ и УлГТУ.

Содержание индивидуального задания приведено в следующей таблице:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Учебная дисциплина | Отчетность | Вид и объем учебной работы | Индивидуальное задание | Кто принимает результат |
| Проектирование информационного обеспечения САПР | Экзамен | Лек.: 32 Лаб.: 32 | Анализ методов и средств создания информационного обеспечения САПР. Разработка информационного обеспечения системы проектирования планов работ. | Г. П. Токмаков  В. Н. Негода |

1. **Анализ методов и средств создания информационного обеспечения САПР**

В рамках жизненного цикла промышленных изделий, САПР решает задачи автоматизации стадий проектирования и подготовки производства.

Основная цель создания САПР — повышение эффективности труда, включая в себя такие пункты как:

1. сокращения трудоёмкости проектирования и планирования;
2. сокращения себестоимости проектирования и изготовления, уменьшение затрат на эксплуатацию;
3. повышения качества и технико-экономического уровня результатов проектирования;

Достижение целей создания САПР обеспечивается путем:

1. автоматизации оформления документации;
2. информационной поддержки и автоматизации принятия решений;
3. использования технологий параллельного проектирования;
4. унификации проектных решений и процессов проектирования;
5. повторного использования проектных решений, данных и наработок;
   1. **Стадии проектирования ИО САПР**

Процесс проектирования рассматривается как начальный этап создания нового изделия и заключается в получении и преобразовании исходного описания объекта в окончательное описание на основе выполнения комплекса работ исследовательского, расчетного и конструкторского характера.

Проектирование, при котором проектные решения получают путем взаимодействия человека, ЭВМ и комплекса программных и других средств автоматизации его деятельности, в ходе которого создается модель предметной области.

Одна из основных целей моделирования данных в ходе проектирования информационных систем − это обеспечение естественного отражения объектов реального мира в терминах структур, ограничений и операций. Под естественностью в первую очередь понимается привычность для человека средств выражения информационных потребностей и простота их освоения.

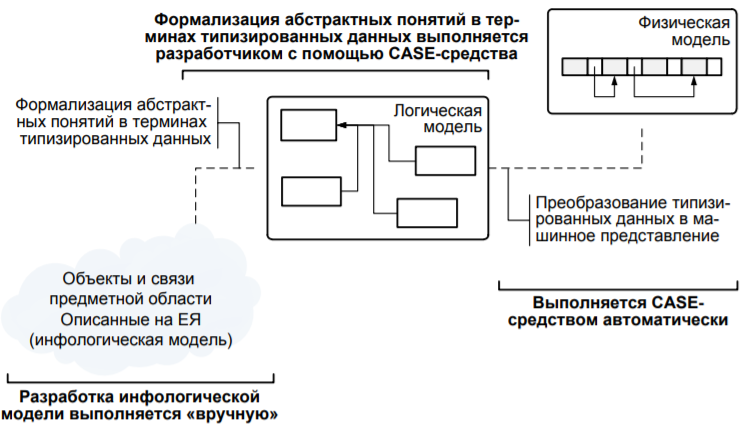


Рис. 1 Стадии и объекты процесса проектирования БД

На первом этапе предметная область описывается через некоторые естественные для человека базовые концепции. Полученное описание относится к концептуальной области моделирования данных. Это описание должно быть пригодно для компьютерной обработки и в то же время понятно специалисту в области автоматизируемой предметной области.

Далее это описание отображается в компьютерно-ориентированное представление, относящееся к логической области моделирования.

Третье отображение, как правило, получается из логического автоматически посредством использования либо средств СУБД, либо специально разработанных CASE-средств.

* 1. **Методы и средства реализации инфологических моделей баз данных**

Инфологическая модель представляет собой описание будущей базы данных, представленное с помощью естественного языка, формул, графиков, диаграмм, таблиц и других средств, понятных как разработчикам БД, так и обычным пользователям.

По состоянию на декабрь 19 года схема бд прототипа для системы поддержки формирования индивидуальных планов преподавателей имеет следующий вид:

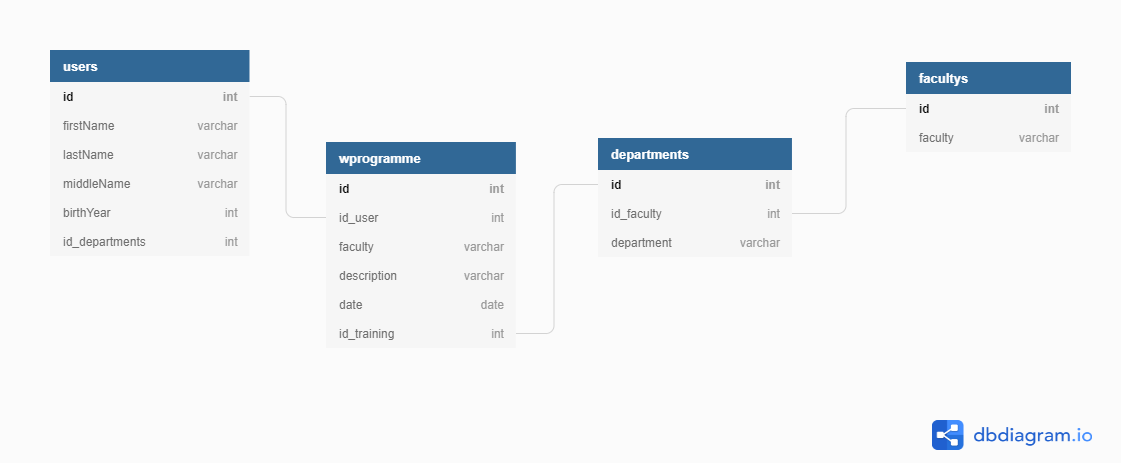


Рис. 2. ER диаграмма схемы бд прототипа

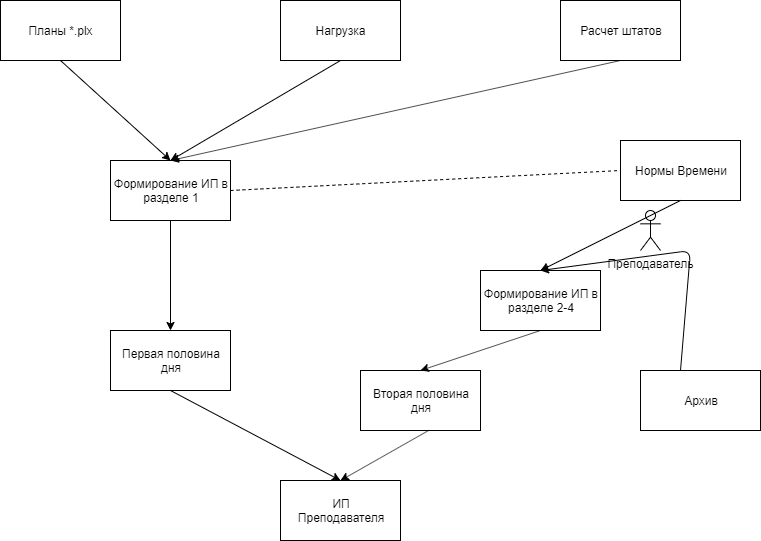


Рис. 3.1. Концепция автоматизированной системы поддержки рабочих программ

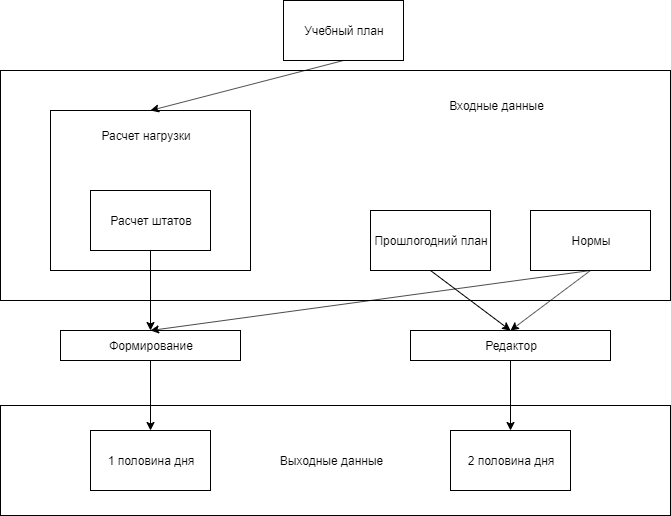


Рис. 3.2. Концепция автоматизированной системы поддержки рабочих программ

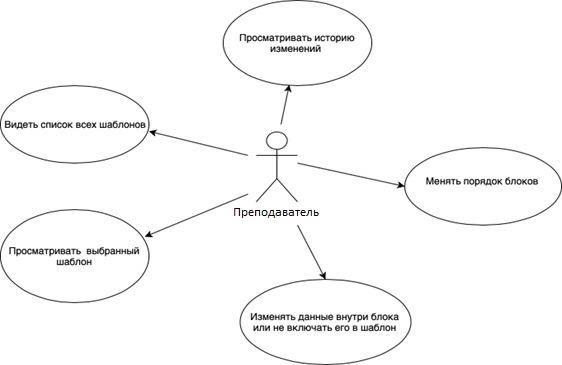


Рис. 4. Пример Use-case диаграммы в системы поддержки рабочих программ

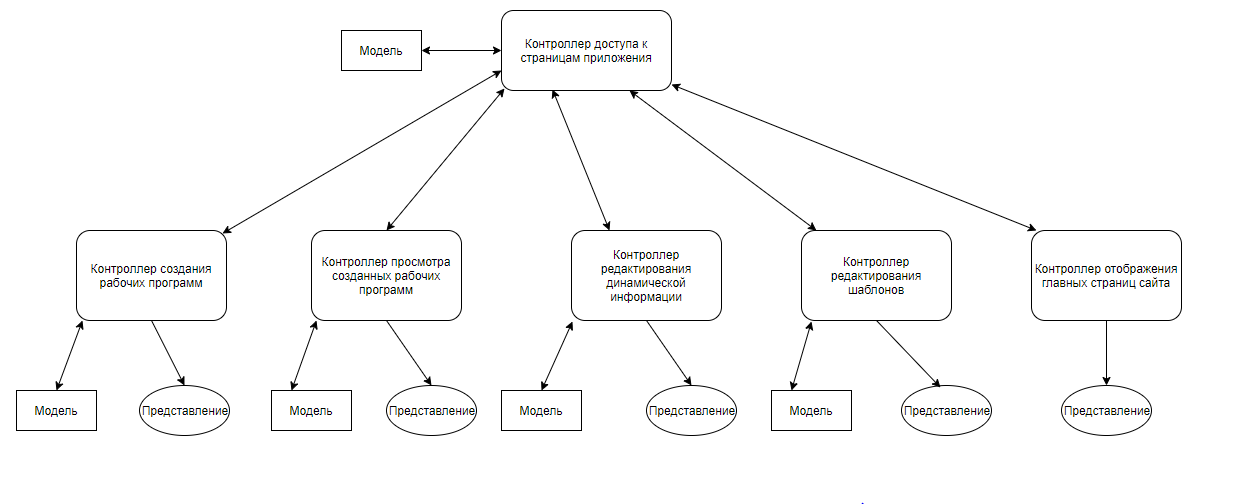


Рис. 5. Схема на основе MVC (Model, View, Controller)

* + 1. **Средства поддержки программирования на SQL в Postgresql**

PostgreSQL не просто реляционная, а объектно-реляционная СУБД. Это даёт ему некоторые преимущества над другими SQL базами данных с открытым исходным кодом.

**Структуры и типы данных**

Существует обширный список типов данных, которые поддерживает Постгрес. Кроме числовых, с плавающей точкой, текстовых, булевых и других ожидаемых типов данных (а также множества их вариаций), PostgreSQL может похвастаться поддержкой uuid, денежного, перечисляемого, геометрического, бинарного типов, сетевых адресов, битовых строк, текстового поиска, xml, json, массивов, композитных типов и диапазонов, а также некоторых внутренних типов для идентификации объектов и местоположения логов.

**Поддержка JSON**

Поддержка JSON в PostgreSQL позволяет вам перейти к хранению schema-less данных в SQL базе данных. Это может быть полезно, когда структура данных требует определённой гибкости: например, если в процессе разработки структура всё ещё меняется или неизвестно, какие поля будет содержать объект данных.

Тип данных JSON обеспечивает проверку корректности JSON, который позволяет использовать специализированные JSON операторы и функции, встроенные в Постгрес для выполнения запросов и манипулирования данными. Также доступен тип JSONB — двоичная разновидность формата JSON, у которой пробелы удаляются, сортировка объектов не сохраняется, вместо этого они хранятся наиболее оптимальным образом, и сохраняется только последнее значение для ключей-дубликатов. JSONB обычно является предпочтительным форматом, поскольку требует меньше места для объектов, может быть проиндексирован и обрабатывается быстрее, так как не требует повторного синтаксического анализа.

У Postgres множество возможностей. Созданный с использованием объектно-реляционной модели, он поддерживает сложные структуры и широкий спектр встроенных и определяемых пользователем типов данных. Он обеспечивает расширенную ёмкость данных и заслужил доверие бережным отношением к целостности данных.

1. **Разработка информационного обеспечения системы поддержки формирования индивидуальных планов преподавателей**

Суть проектирования заключается в реализации 2 систем: серверной (Back-end на Фреймворке NestJs с поддержкой TypeScript) и пользовательского интерфейса (Front-end на Фреймворке Angular с поддержкой TypeScript) для автоматизированной системы поддержки рабочих программ.

* 1. **Анализ представление учебных планов образовательных программ и разработка механизмов извлечения данных из них.**

Учебный план – это документ, устанавливающий список предметов, объем рабочего времени, порядок следования и отдельные этапы подготовки учебных дисциплин, стажировок, факультативных занятий, учебных модулей и других видов практической образовательной деятельности. Также в нем указываются положения о промежуточной оценке студентов и учеников, если этого не определяет Федеральный закон об образовании.

Учебный план представлен в формате .plx и содержит в себе XML разметку, содержащую в себе необходимую информацию для создания шаблона рабочей программы в автоматическом режиме.

Благодаря тому, что XML является универсальным форматом для обмена информацией, взаимодействие с данным языком разметки имеет простую структуру. Для работы с ним используются так называемые XML-анализаторы, которые работают с xml-разметкой и позволяют находить по тегам необходимую информацию, которую нужно выделить из файла.

Чтобы работа с xml файлом была удобней, принято то что его нужно распарить в формате json, для удобства чтения и простоты использования данных внутри проекта. Парсинг происходит с помощью библиотеки xml2js, взятой из менеджера пакетов JS.

Листинг 1. Загрузка и парсинг xml-файла.

let { parseString, Builder} = require('xml2js');  
let xml = fs.readFileSync("09.03.01\_02-2019.plx", "utf8");

parseString(xml, { explicitArray : false }, (error, result) => {

if(error) throw error;

fs.writeFileSync("json", JSON.stringify(result, null, 4));

console.log(result.Документ);

});



Рис. 6. Файл json

* 1. **Анализ представления расчета штатов и разработка механизмов извлечения данных из них.**

Расчет штатов (нагрузка преподавателя) представляет собой файл расширения .xlsm, но также часы по какому-либо предмету можно найти в в файле учебного плана с расширением .plx (был рассмотрен ранее в П.3.2). Так как заведомо в автоматизированной системе входные данные неявно заданы (могут храниться в xlm или excel), то разработка будет происходить не сверху вниз, а снизу-вверх (отталкиваемся от выходного файла, который должен получиться после использования расчетов штатов – 1 половина дня).

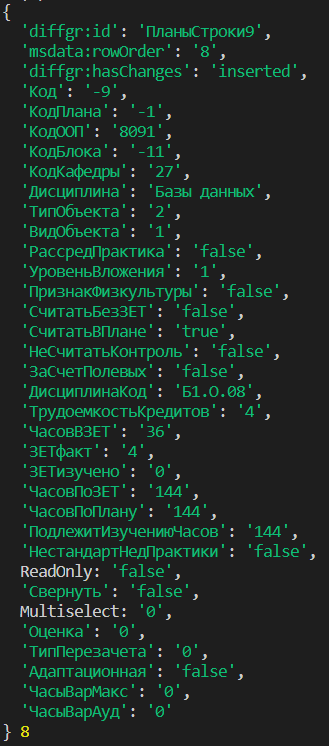


Рис. 7. Пример объекта дисциплины Базы данных

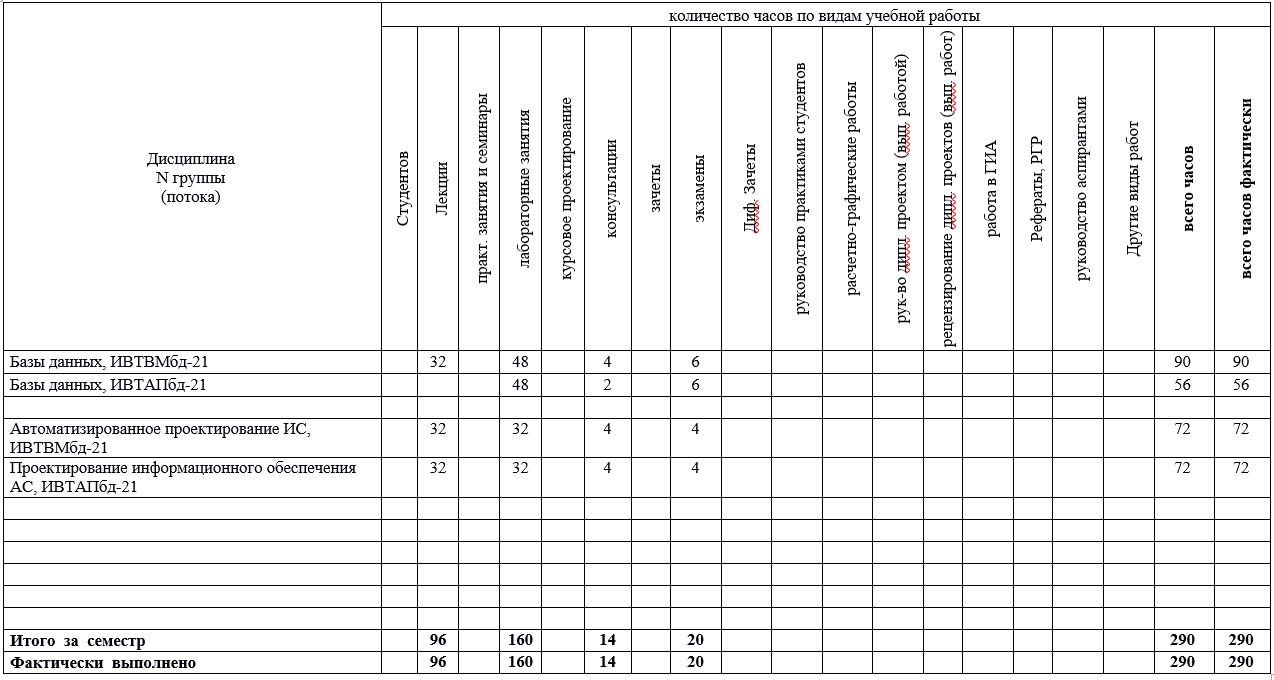


Рис. 6. Нагрузка преподавателя по программам высшего образования

* 1. **Анализ представления индивидуальных планов в части учебных занятий и разработка механизмов автоматизации формирования этой части.**

Прототип индивидуальных планов в части учебно-методической, организационно-методической и научно-исследовательской работы и разработка механизмов автоматизации формирования будет представлять собой:

* Парсинг данных
* Сохранение в базе данных
* Создание шаблона документа
* Заполнение шаблона
* Запись данных в шаблон

Листинг 2. Создание и сохранение данных в документ по шаблону.

let content = fs.readFileSync('template.docx', 'binary');

let zip = new PizZip(content);

let doc = new Docxtemplater();

doc.loadZip(zip);

doc.setData({

faculty : faculty,

nameDepartment : nameDepartment,

firstName : "Токмаков",

lastName : "Геннадий",

middleName : "Петрович",

position : "профессор",

birthYear : 1951

});

doc.render()

let buf = doc.getZip().generate({type: 'nodebuffer'});

fs.writeFileSync('output.docx', buf);

Пример прототипа, создание документа:

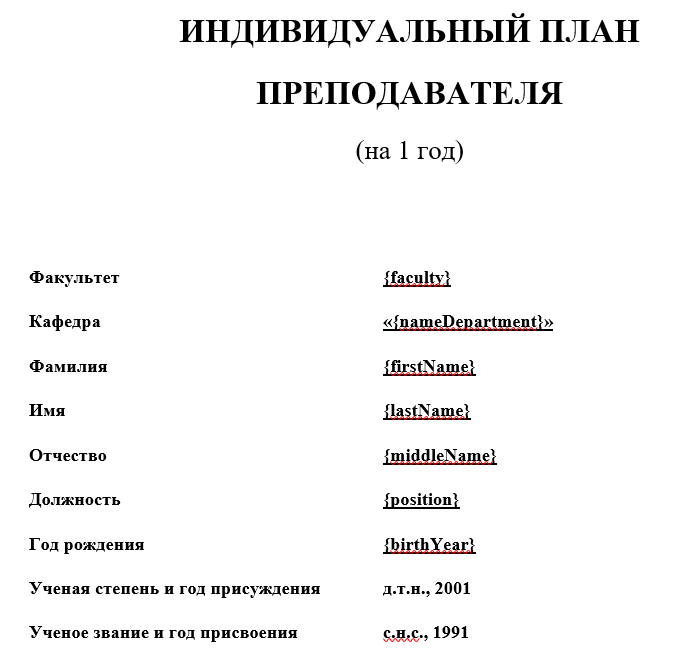
****

Рис. 8. Шаблон документа template.docx