

## РЕФЕРАТ

Пояснительная записка 62 с., 27 рис., 6 табл., 18 источн., прил.

### АВТОМАТИЗИРОВАННАЯ ИНФОРМАЦИОННАЯ СИСТЕМА, ПРОГРАММНОЕ ОБЕСПЕЧЕНИЕ, ЛИЦЕНЗИЯ, УПРАВЛЕНИЕ ИСПОЛЬЗОВАНИЕМ ЛИЦЕНЗИОННОГО ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ

Цель выпускной квалификационной работы – разработка проектных решений и макета приложения информационной системы (ИС), поддерживающей процессы учета состояния и использования лицензионного программного обеспечения (ПО) в университете.

Объектом исследования является лицензионное ПО.

Предметом исследования является автоматизация управления использованием лицензионного ПО университета.

Метод выполнения работы определяется решением следующей последовательности задач:

- описание предметной области (ПрО);
- разработка концептуальной модели ПрО;
- определение проблем и формирование концепции ИС;
- разработка концептуальной модели ИС;
- разработка логической модели макета приложения «Мониторинг использования ПО»;
- анализ возможностей информационных технологий и технических средств университета для целей разработки макета приложения для ИС и функционирования ИС;
- разработка макета приложения «Мониторинг использования ПО».

В результате выполнения работы получены проектные решения и разработан макет приложения ИС, поддерживающей процессы учета состояния и использования лицензионного ПО в университете.

## СОДЕРЖАНИЕ

ОПРЕДЕЛЕНИЯ, ОБОЗНАЧЕНИЯ И СОКРАЩЕНИЯ.....	5
ВВЕДЕНИЕ .....	6
1 Проектный раздел .....	8
1.1 Описание предметной области .....	8
1.1.1 Организационная структура.....	8
1.1.2 Процессы.....	10
1.1.3 Основные знания предметной области.....	12
1.2 Разработка концептуальной модели предметной области .....	13
1.3 Проблемы предметной области и концепция информационной системы .....	18
1.3.1 Проблемы предметной области .....	18
1.3.2 Концепция информационной системы .....	19
1.3.2.1 Основные понятия .....	19
1.3.2.2 Функциональные требования.....	20
1.3.2.3 Нефункциональные требования.....	20
1.4 Разработка концептуальной модели информационной системы.....	21
1.5 Разработка логической модели макета приложения «Мониторинг использования ПО».....	28
1.5.1 Модель поведения .....	28
1.5.2 Модель структуры .....	29
2 Технологический раздел .....	31
2.1 Применение ИТ-активов университета для целей разработки макета приложения и функционирования ИС .....	31
2.2 Разработка макета приложения «Мониторинг использования ПО» (функции пользовательского интерфейса) .....	34
2.2.1 База данных.....	34
2.2.2 Язык программирования .....	39
2.2.3 Среда разработки .....	39
2.2.4 Разработка экранных форм пользовательского интерфейса ...	39

2.2.5 Обобщенная схема информационной системы.....	48
ЗАКЛЮЧЕНИЕ .....	50
СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ .....	51
ПРИЛОЖЕНИЕ А .....	53

## ОПРЕДЕЛЕНИЯ, ОБОЗНАЧЕНИЯ И СОКРАЩЕНИЯ

В настоящей пояснительной записке применяют следующие сокращения и обозначения:

Актор – связанный набор ролей, исполняемый пользователем при взаимодействии с вариантом использования.

БД – база данных.

ИС – информационная система.

ИТ-актив – информационно-технологический актив.

Конфигурация лицензионного ПО – структурированная совокупность ПО, обеспечивающая информационную поддержку учебного процесса, управления, исследований и работ поддержки физической инфраструктуры университета. ПО размещается с учетом его архитектуры на ИТ-актив.

Объект конфигурации – составная часть парка ИТ-средств университета, значимая для выполнения установленных требований и рассматриваемая в процедурах управления конфигурацией как единое целое [1].

Изменение конфигурации – изменение структурированной совокупности лицензионного ПО, размещенного на ИТ-активе, а также документации (данных) в соответствии с проводимыми изменениями.

ОС – операционная система.

ПО – программное обеспечение.

ПрО – предметная область.

ПС – программное средство.

ПП – программный продукт.

Состояние ПО – состояние лицензионного ПО. Определяется наличием актуальной лицензии, значением признака «использование по назначению», количеством установок на сервера (серверные части), рабочие станции (клиентские части), отдельные компьютеры.

СУБД – система управления базами данных.

УИТ – управление информационными технологиями.

UML — унифицированный язык моделирования).

## **ВВЕДЕНИЕ**

Применение в обучении компьютерных средств ставит перед руководством университета задачу эффективного использования как технических средств, так и лицензионного программного обеспечения (ПО). Компьютерные средства применяются в учебном процессе, в управлении университетом, исследовательской деятельности, то есть, во всех ключевых видах деятельности университета. Поэтому с увеличением количества технических средств и лицензионного ПО и их важности в процессах университета появляется необходимость в их рациональном использовании и, как следствие, в автоматизации процессов количественного учета, а также процессов учета состояния и, в частности, состояния лицензионного ПО. Для этих целей рассматривается возможность применения информационной системы (ИС), поддерживающей процессы мониторинга использования лицензионного ПО в университете.

ИС – взаимосвязанная совокупность баз данных (БД), информационных технологий, технических средств и персонала, используемых для хранения, обработки и выдачи информации при принятии решений по достижению целей управления.

Как правило, цель создания ИС для университета – обеспечить руководство и подразделения эффективным инструментарием информационной поддержки формирования состава лицензионного ПО, учета, контроля применения и замещения ПО университета.

ИС управления лицензионным ПО позволяет: повысить эффективность деятельности университета; анализировать потребности подразделений университета (кафедры, исследовательские лаборатории и т.д.) в лицензионном ПО; удаленно контролировать состояние ПО, а также отслеживать состояние ПО на протяжении его жизненного цикла.

Цель выпускной квалификационной работы – разработка проектных решений и макета приложения ИС, поддерживающей процессы учета состояния и использования лицензионного ПО в университете.

Для достижения поставленной цели решены следующие задачи:

- описана предметная область (ПрО);
- разработана модель предметной области и концепция ИС;
- разработана концептуальная модель ИС;
- разработана логическая модель макета приложения ИС (экранные формы пользовательского интерфейса);
- проанализированы возможности информационных технологий и технических средств университета для размещения приложения мониторинга использования ПО;
- разработан макет приложения «Мониторинг использования ПО».

Полученные решения перечисленных задач позволяют разработать в дальнейшем ИС управления, которая обеспечит возможность мониторинга состояния лицензионного ПО университета, планировать приобретение лицензий и формировать заказы на продление существующих лицензий, то есть принимать обоснованные и своевременные управленческие решения.

## **1 Проектный раздел**

### **1.1 Описание предметной области**

В данном разделе приводится описание ПрО, которая представлена процессами управления использованием лицензионного ПО Балтийского государственного технического университета «ВОЕНМЕХ» им. Д. Ф. Устинова (БГТУ «ВОЕНМЕХ» им. Д. Ф. Устинова).

Университет является ярким примером высшей технической школы России, где смогли не только сберечь, но и увеличить достижения отечественного и мирового инженерно-технического образования.

БГТУ «Военмех» им. Д. Ф. Устинова занимает одно из первых мест в России из числа университетов по успешности выпускников. За долгое время существования Военмех выпустил огромное количество квалифицированных специалистов. Из стен университета вышли главные конструкторы систем вооружения, оружия, боеприпасов, директора оборонных организаций, видные партийные и государственные деятели, летчики-космонавты [2].

#### **1.1.1 Организационная структура**

Управление университетом осуществляется в соответствии с законодательством Российской Федерации, Типовым положением об образовательном учреждении высшего профессионального образования (высшем учебном заведении) и уставом на принципах сочетания единоначалия и коллегиальности и обеспечивает оптимальное функционирование всех структурных подразделений.

Университет обладает сложной структурой управления, которая включает большое число отделов. Во главе университета стоит ректор, заместителями по разным сферам деятельности являются проректоры, которые решают оперативные и тактические вопросы работы высшего учебного заведения. Стратегические вопросы развития решает Учёный совет, а отдел кадров занимается вопросами управления персоналом [3].

Примерная организационная структура изображена на рисунке 1.

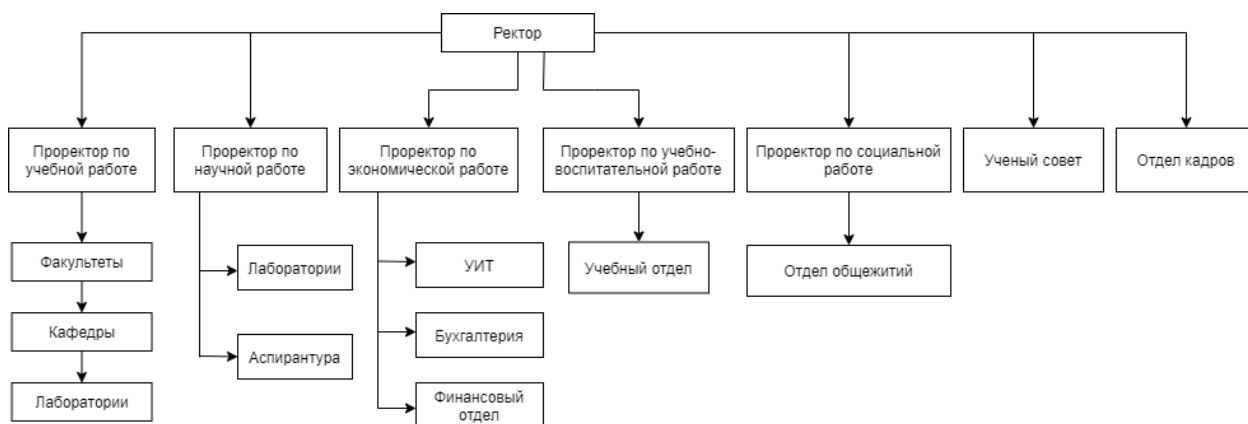


Рисунок 1 – Организационная структура

За состояние информационно-технических активов (ИТ-актив) отвечает сотрудник управления информационных технологий (УИТ), который управляет использованием лицензионного ПО в университете. В состав заинтересованных лиц предметной области (ПрО) входят: заказчики ПО, сотрудники по его установке, ответственные за техническую поддержку состояния ПО, а также сотрудники, занимающиеся покупкой лицензий и мониторингом их эффективного использования.

Лицензионное ПО входит в состав ИТ-актив и характеризуется областью применения, функциональными возможностями, правовым статусом, состоянием программного продукта, а также информацией о производителе.

Структурная схема представлена на рисунке 2.

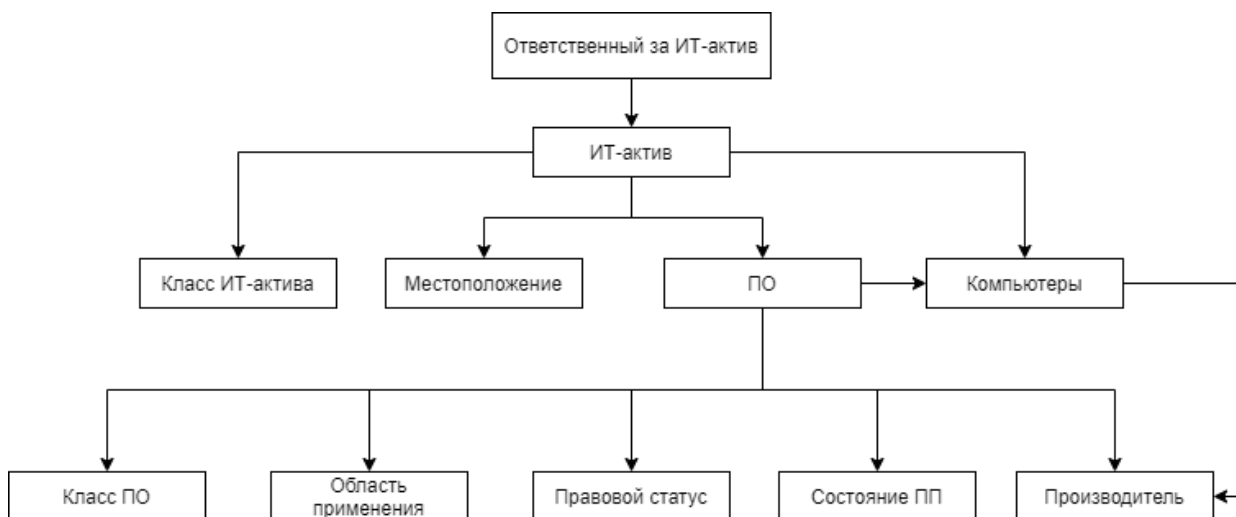


Рисунок 2 – Структурная схема ИТ-актива



### **1.1.2 Процессы**

Процесс управления ИТ-актив – это управленческий процесс по планированию, учету и отслеживанию состояния ИТ-актив и, соответственно, его компонента – лицензионного ПО.

Руководство управления информационных технологий совместно с руководством университета составляют план использования ИТ-актив, в который входит планирование приобретения, установка, использование, замена и утилизации компонентов ИТ-актива, включая и компоненты лицензионного ПО.

Основными процессами предметной области можно считать:

#### **1. Мониторинг состояния ИТ-актива**

Мониторинг заключается в проверке ПО на то, являются ли они работоспособными, актуальными и используются ли в той области деятельности, для которой приобретены. Процесс мониторинга является конституирующим процессом предметной области. Поэтому для университета важно следить за состоянием ИТ-актива. Результаты мониторинга способствуют принятию решений о дальнейшем использовании или, наоборот, неиспользовании ПО, чтобы в будущем грамотно использовать финансовые средства университета.

#### **2. Планирование приобретения и использования ПО**

Для того чтобы грамотно использовать финансовые средства университета, а также грамотного использования ПО необходимо планировать приобретение ПО. К планированию можно отнести составления планов-графиков закупок, а также планирование финансовых расходов на будущие покупки ПО. Такие планы-графики составляет сотрудник УИТ и передает руководству университета.

#### **3. Покупка и установка ИТ-актива**

Покупка и установка ИТ-актива происходит в зависимости от потребностей подразделений университета (кафедры, исследовательские лаборатории и т. д.), его финансовых ресурсов и необходимости данных активов. Ис-

ходя из этого университет может выбрать только те ИТ-активы, которые необходимы.

В начале каждого семестра сотрудник УИТ составляет план-график закупки на приобретение нового ПО, включая лицензии к нему, а также на продление лицензии некоторых устаревших ПО, в которых нуждается университет. План-график закупок включается в отчет, где прописаны состояния лицензий ПО за прошедший семестр. Далее отчет передается руководству университета, где он либо одобряется, либо передается обратно в УИТ с целью внесения изменений.

#### 4. Конфигурация ПО под условия и требования применения ИТ-актива

Университет может нуждаться в дополнении ПО или же, наоборот, в его сокращении из-за изменений в учебном процессе или исследовательских работах.

Конфигурация ПО заключается в том, что количество установленных продуктов ПО может быть расширено, уменьшено или переустановлено, если в том или ином случае оно покупалось модулями.

Конфигурация системы ИТ-актива (системы  $S_{ITA}$ ) определяется кортежем из следующих взаимосвязанных понятий:

$S_{ITA} = (L, LT, ltype, D, DT, dtype, SN, lmap, A, AT, atype, fmap,)$ , где:

- $D$  – конечное непустое множество устройств ИТ-актива, входящих в систему  $S$ ;
- $DT$  – конечное, возможно пустое, множество типов ИТ-актива, используемых в системе  $S$ ;
- $dtype: D \rightarrow DT$  – функция, назначающая отдельным устройствам ИТ-актива тип;
- $A$  – конечное множество приложений системы  $S$ ;
- $AT$  – конечное множество типов приложений (и их составных частей), используемых в системе  $S_{ITA}$
- $atype: A \rightarrow AT$  – функция, назначающая отдельным  $A$  тип  $A$ ;

- $fmap: FA \rightarrow FD$  – отображение (с учетом значений ресурсов) функций приложений (ПО) на функции и ресурсы ИТ-актива;
- $lmap: D \rightarrow LD$  – размещение (с учетом пространственных характеристик) устройств ИТ-актива на места их эксплуатации;
- $SN$  – множество сегментов сети, на которых выполняются приложения  $A$ .

Функции  $*map$  и  $*type$  рассматриваются как функции управления конфигурацией (типизировать объект, создать конфигурацию объектов).

### 5. Модернизация, замена и утилизация ИТ-актива

В результате мониторинга состояния ИТ-актив аи лицензионного ПО могут быть приняты решения об их дальнейшем использовании. Модернизация ИТ-активов происходит для обновления активов, приведения их в соответствии с новыми требованиями и нормами, техническими условиями, показателями качества. Замена ИТ-активов используется в том случае, если активы устарели и не заявлены для использования в учебном процессе. Кроме того, в процессе замены выявляются активы, которые давно не использовались или использовались не по назначению.

#### 1.1.3 Основные знания предметной области

На основании приведенного выше описания процессов выделим основные знания, связанные с управлением использованием лицензионного ПО и представим их в таблице.

Основные знания о предметной области представлены в таблице 1.

Таблица 1 – Основные знания предметной области

Данные	Первоначальные данные. Правовой статус. Состояние лицензионного ПО. Лицензия ПО. Оплата лицензии. План закупки ПО. Отчет.
Функции	Мониторинг состояния ИТ-активов. Планирование и использования

	Покупка и установка ПО, а также модернизация, замена и утилизация ИТ-активов
--	--

Продолжение таблицы 1

Рабочие места	Местоположение ИТ-активов в университете.
Люди	Ректорат, сотрудники университета, ответственный за ИТ-актив
Процессы	Управление состоянием лицензионного ПО.
Цели	Рациональное использование ИТ-активов.

## 1.2 Разработка концептуальной модели предметной области

Для разработки модели предметной области исходными данными будет являться информация, полученная из предыдущего подраздела.

Конституирующим процессом для данной предметной области является процесс управления использованием лицензионного ПО. Он включает в себя процессы планирования и мониторинга состояния ПО.

Университет заинтересован в использовании для моделирования ПрО языка визуального моделирования – UML, который помогает представить абстрактные модели системы в виде графических диаграмм, а также описать основные объекты предметной области и их поведения.

Определим перечень необходимых высказываний, которые определяют основные отношения ПрО и является устойчивым на протяжении длительного времени его существования. Такой перечень представлен в таблице 2.

Таблица 2 – Перечень необходимых высказываний о ПрО

№	Необходимые высказывания
1.	Предметная область определяется процессом управления лицензионным ПО университета.
2.	Сотрудник УИТ заносит информацию о текущем состоянии лицензионного ПО университета.
3.	Периодически сотрудник УИТ составляет подробный отчет об изменении состояния ПО (события и даты покупки, проектирования, разработки, эксплуатации и утилизации ПО)

#### 4. Руководство университета запрашивает данные о состояниях ПО

Диаграмма вариантов использования позволяет отразить отношения между акторами и прецедентами. С ее помощью заказчику, разработчику и конечному пользователю будет удобно обсуждать проектируемую систему [4].

На рисунке 1 представлена диаграмма вариантов использования, моделирующая основное отношение и его основные процессы.

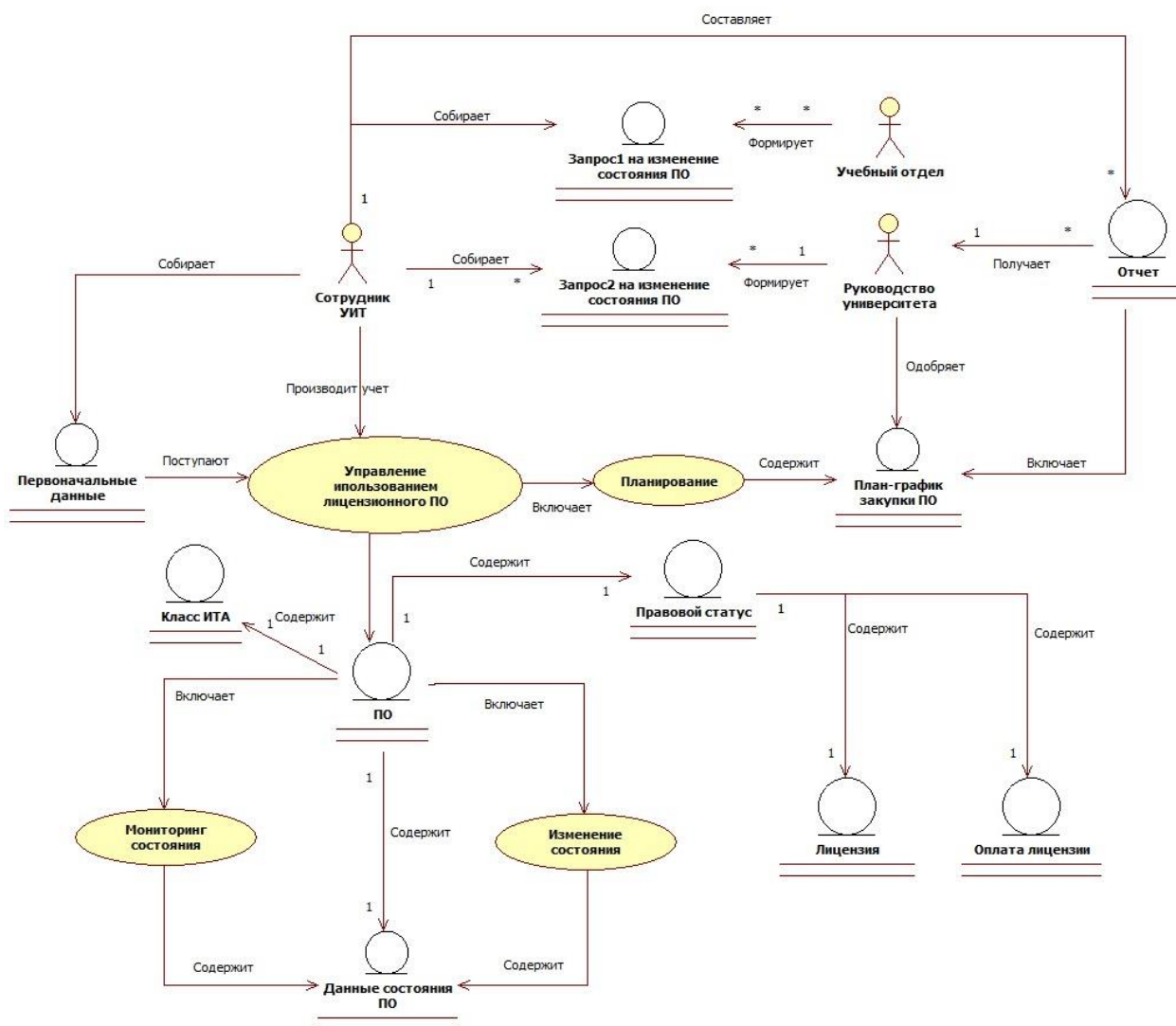


Рисунок 1 – Диаграмма вариантов использования, моделирующая основные функциональные отношения предметной области

Для того чтобы детально визуализировать процесс управления использованием лицензионного ПО университета можно прибегнуть к использова-

С помощью диаграммы активности можно увидеть виды деятельности или каких-либо действий, которые соединены между собой потоками, идущими от выхода одного узла к входу другого [5].

Диаграммы активности представлены на рисунках 2-3.

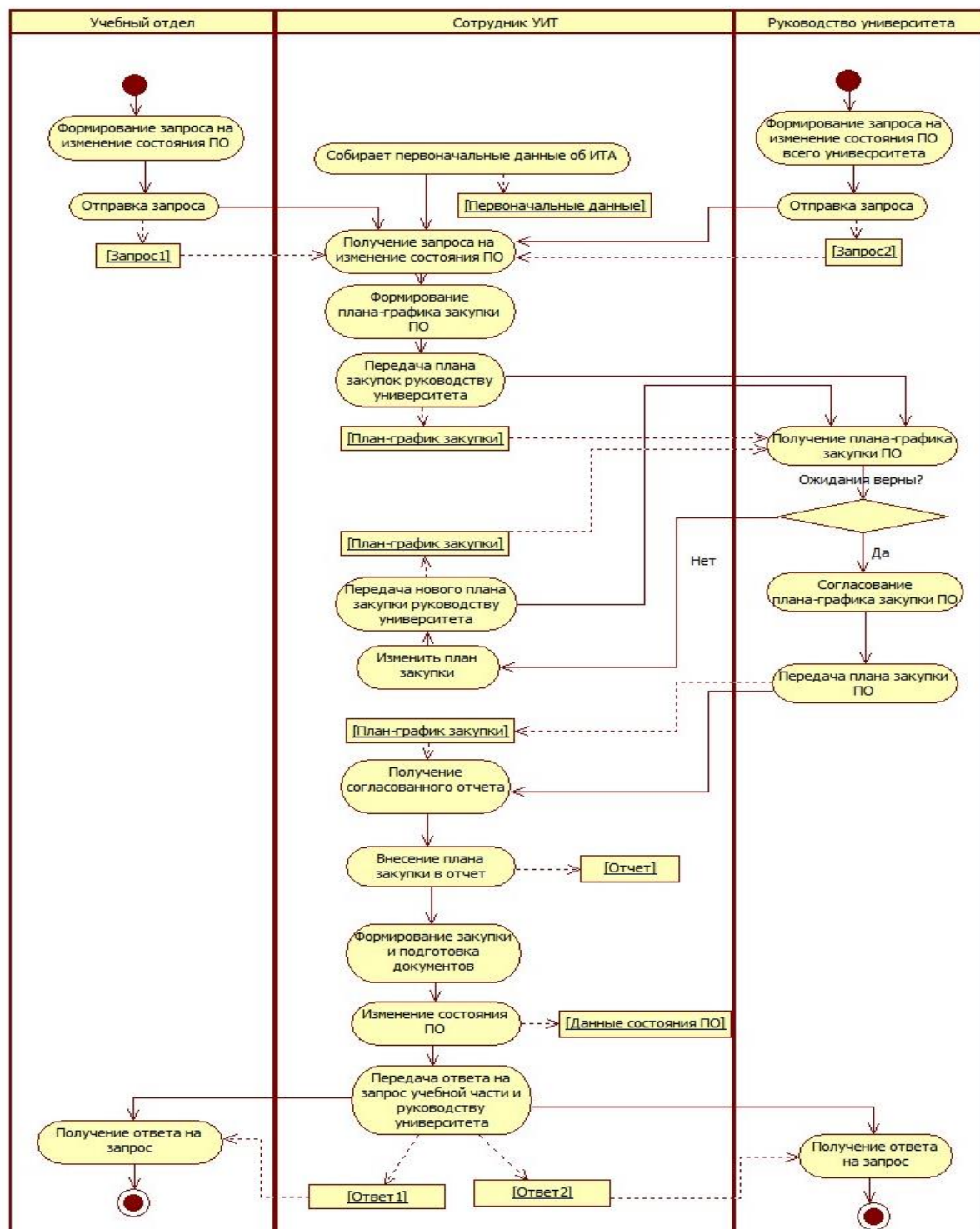


Рисунок 2 – Диаграмма активности, моделирующая процесс управления использованием лицензионного ПО

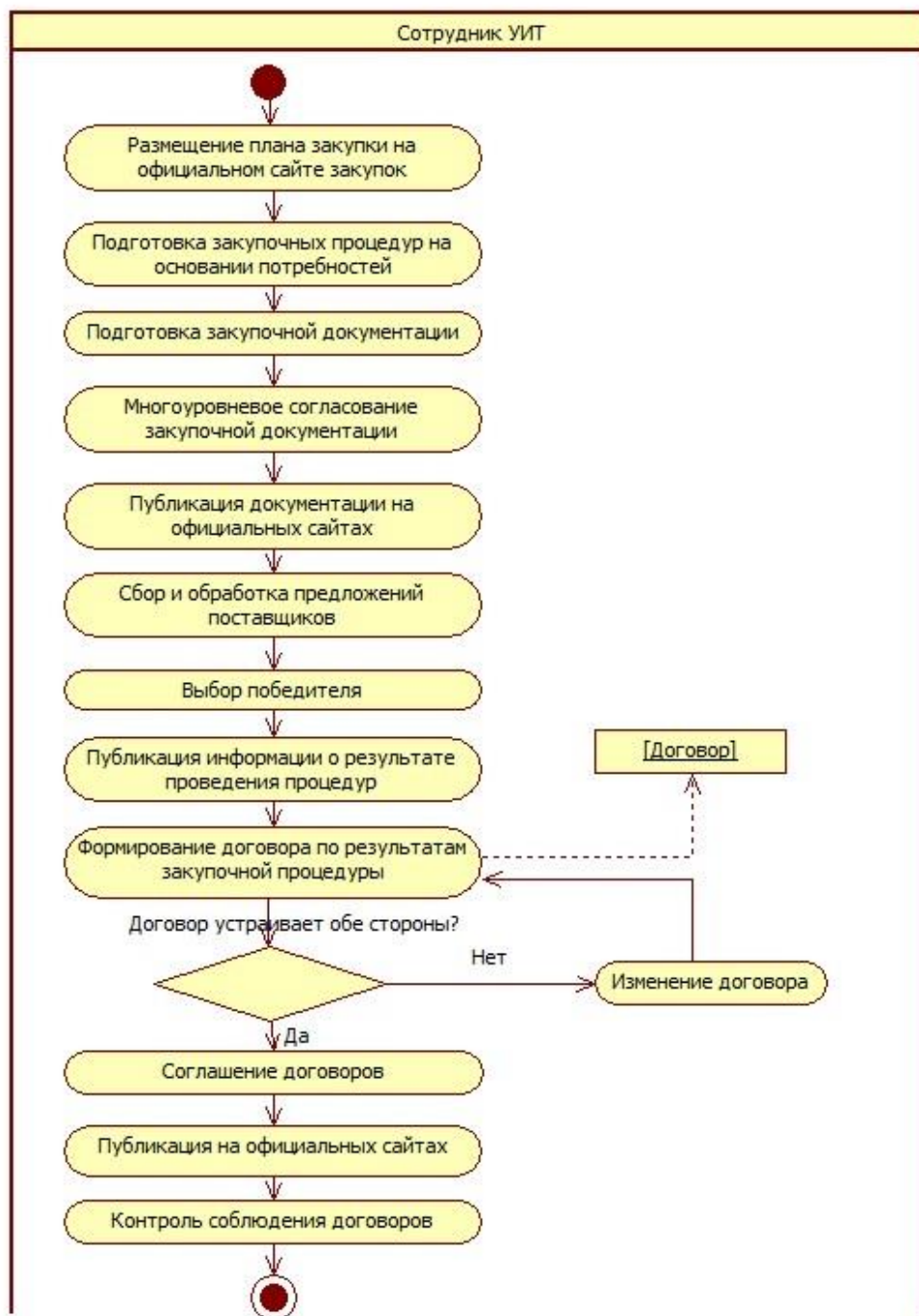


Рисунок 3 – Диаграмма активности, моделирующая процесс формирования плана (спецификации) закупки

Моделирование отношений ключевых объектов ПрО выполнено посредством диаграммы классов. Диаграмма классов – диаграмма, позволяю-

шая продемонстрировать взаимосвязи между сущностями предметной области, а также описать внутреннюю структуру и типы отношений.

Основные виды символов в диаграммы активности:

- изображение человека используется для обозначения актора (действующего лица), выполняющего свои функции в предметной области;
- круг с нижним подчеркиванием используется для обозначения класса сущности.

Диаграмма классов представлена на рисунке 4.

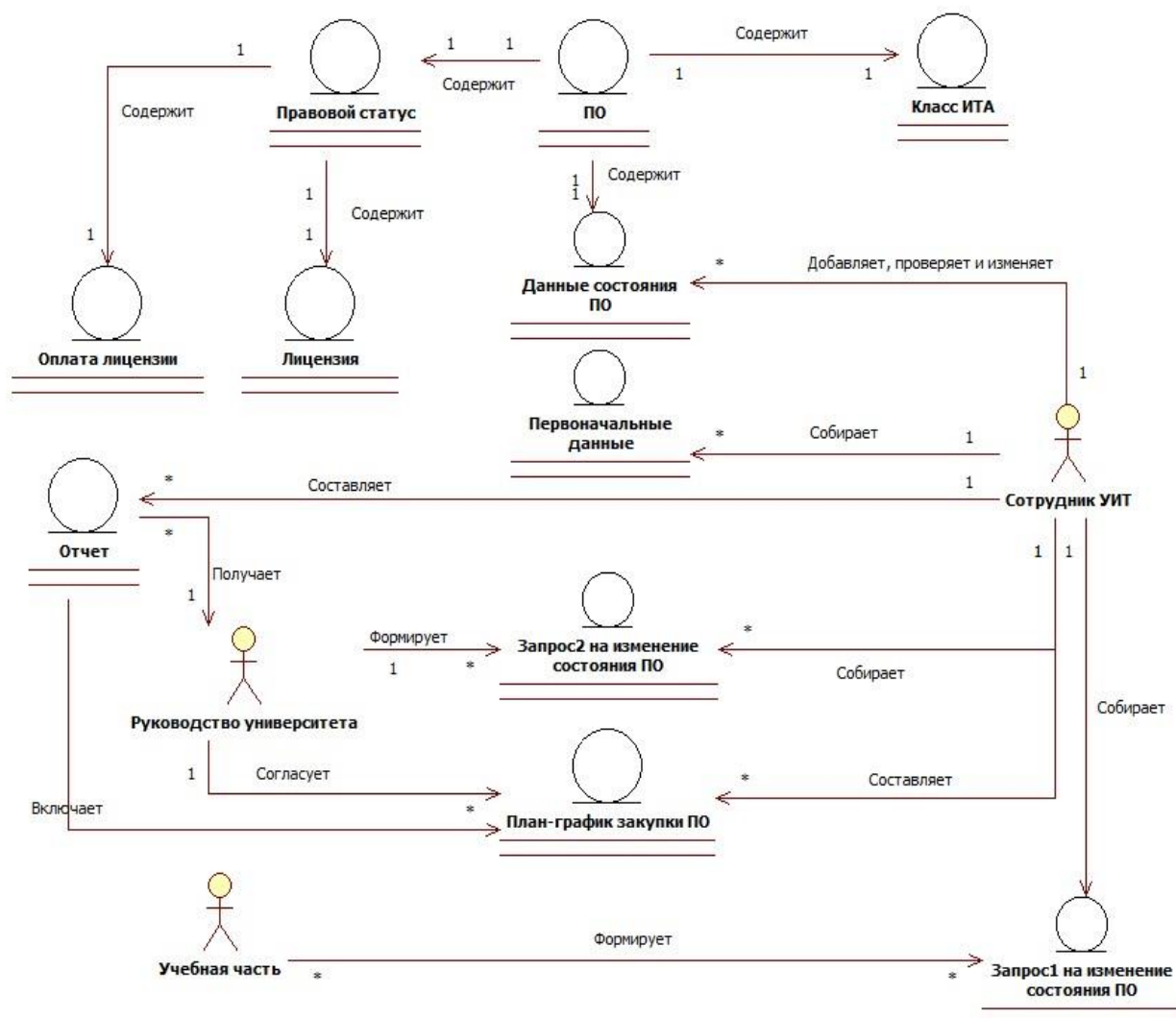


Рисунок 4 – Диаграмма классов, моделирующая основные отношения между ключевыми объектами предметной области



### **1.3 Проблемы предметной области и концепция информационной системы**

Данный раздел отражает результаты проблемного анализа ПрО и связывает проблемы и функциональные требования к ИС посредством отношения «решения проблем».

#### **1.3.1 Проблемы предметной области**

Университет организационно состоит из факультетов. В свою очередь каждый факультет включает несколько кафедр. На каждой кафедре работают преподаватели, главные инженеры, лаборанты.

Процесс обучения осуществляется на основе учебных планов, составленных на выпускающих кафедрах, которые затем утверждаются деканом и начальником учебного управления.

Для комфортного проведения занятий в университете существуют аудитории. Каждая аудитория университета обладает некоторым количеством компьютеров, на каждом установлено различное программное обеспечение. У ПО имеются лицензии, документы о праве использования ПО, в котором указан срок действия лицензий, после истечения которого университету нужно оплатить продление лицензии или же отказаться от использования ПО.

В настоящее время в университете отсутствует ИС, которая отслеживает состояния лицензионного ПО, хранит отчеты и планы дальнейших закупок ПО, вносит и изменяет данные удаленно. В результате чего существует ряд проблем в управлении использованием ПО.

Таким образом, управление использованием лицензионного ПО – сложная задача, требующая большой обработки данных. Использование ИС позволит значительно сократить временные и трудовые затраты на выполнение решения ряда проблем. Следовательно, необходимо автоматизировать процесс управления использованием ПО посредством разработки и внедрения ИС соответствующего назначения.

В результате анализа были выявлены следующие проблемы предметной области:

1. Трудоемкость процесса обнаружения устаревшей информации или внесения новой информации об изменениях состояния ПО.
2. Возможность внесения ошибочной информации о состоянии ПО.
3. Сложность и трудоемкость построения прогнозов по составу и стоимости закупаемых лицензий ПО.

### **1.3.2 Концепция информационной системы**

#### **1.3.2.1 Основные понятия**

Основные понятия, используемые для описания функционирования ИС:

Лицензия – документ, дающий право (или права) на осуществление конкретного вида действий.

Запрос на изменение состояния ПО – документ, содержащий запрос на обновление/закупку того или иного ПО.

Данные состояния – данные, которые сотрудник УИТ собирает о ПО.

Оплата лицензии – документ, подтверждающий оплату коммерческой лицензии.

Сотрудник УИТ – лицо, работающее в университете и отвечающий за какие-либо функции.

Руководство университета – люди, которые принимают решения и осуществляют контроль.

Учебный отдел – подразделение университета, занимающееся вопросами организации, планировании и контролем образовательной деятельности.

Отчет – документ, включающий в себя состояние лицензионного ПО и план закупки лицензий на планируемый период.

Первоначальные данные – данные, которые необходимо собрать до разработки ИС.

План-график закупки ПО – документ, содержащий в себе перечень ПО, которое необходимо закупить, включая их стоимость.

Правовой статус ПО – статус, который позволяет на правовой основе выполнять определенные действия в отношении ПО.

#### **1.3.2.2 Функциональные требования**

Разработка функциональных требований к ИС для управления использованием лицензионного ПО является неотъемлемой частью разработки любой ИС.

Функциональные требования – это описание ожидаемого поведения системы, которая описывает все возможные ситуации.

Поэтому ИС должна обладать следующими функциональными возможностями:

- проверка актуальности предоставляемой информации об ИТ-активах;
- поддержка планирования размещения ПО на ИТ-активах;
- поддержка планирования закупок ПО с учетом выделенного бюджета;
- добавление, редактирование, удаление информации о состоянии лицензионного ПО;
- обработка электронных документов, связанных с управлением лицензионным ПО.

Существует ряд обеспечивающих функциональных требований, к которым относятся следующие функции:

- защита информации от несанкционированного доступа и изменений;
- обеспечение возможности резервного копирования данных;
- хранение отчетов о состоянии лицензионного ПО.

#### **1.3.2.3 Нефункциональные требования**

Нефункциональные требования определяют атрибуты качества разрабатываемой системы. К таким требованиям относятся характеристики,

важные для пользователей. Это может быть производительность, простота использования, минимальное количество сбоев, надежность.

Поэтому ИС должна работать в рабочее время университета, производительность системы должна обеспечивать формирование отчетов без задержки, одновременную работу до 20 человек.

#### **1.4 Разработка концептуальной модели информационной системы**

Проектирование ИС, способной решать проблемы предметной области, начинают с разработки UML-диаграмм, моделирующих функциональные возможности и структуру ИС ПО на концептуальном уровне.

Для разработки концептуальной модели ИС исходными данными будет является модель функциональных требований, в которой содержится перечень функций, которым должна обладать ИС.

Для того что наглядно показать жизненный цикл управления лицензионным ПО университета, а также взаимодействие актора ИС с объектами используется диаграмма последовательности.

Диаграмма последовательности является одним из видов диаграмм языка UML, которая описывает отношения объектов.

К основным элементам данной диаграммы относятся:

- прямоугольники с названиями объектов;
- вертикальные «линии жизни», которые отображают течение времени;
- прямоугольники, отражающие деятельность объекта или исполнение ими каких-либо функций
- стрелки, показывающие обмен сигналами и сообщениями между самими объектами [6].

Диаграммы последовательностей, представленные на рисунке 5 и рисунке 6, моделируют взаимодействие актора и ИС при управлении лицензионным ПО университета. Кроме того, эти диаграммы, моделируя взаимодействие по принципу «черного ящика», определяют внутренние функции ИС, посредством которых обрабатываются запросы акторов к системе.

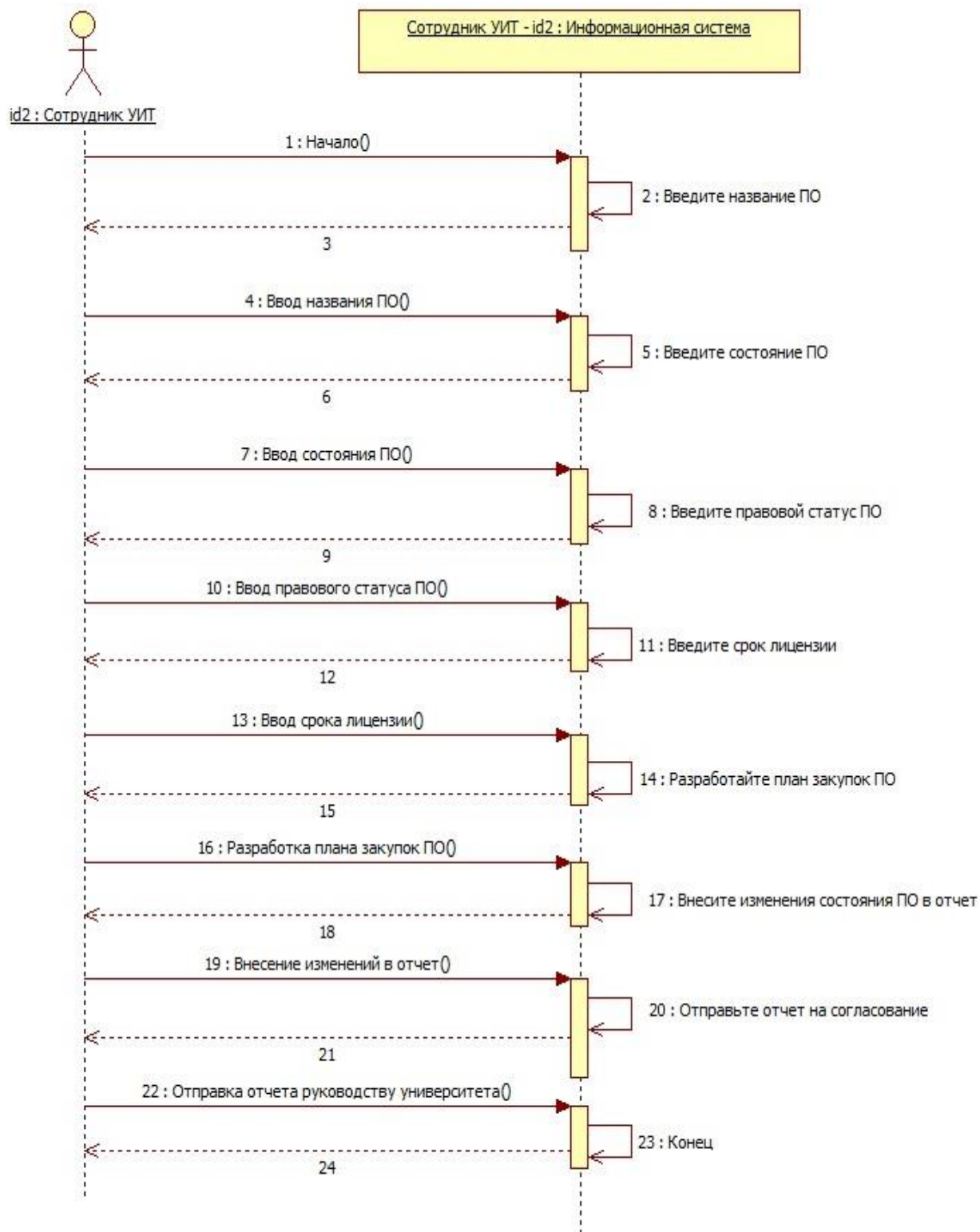


Рисунок 5 – Диаграмма последовательности, моделирующая отправку отчета на согласование закупки ПО сотрудником УИТ

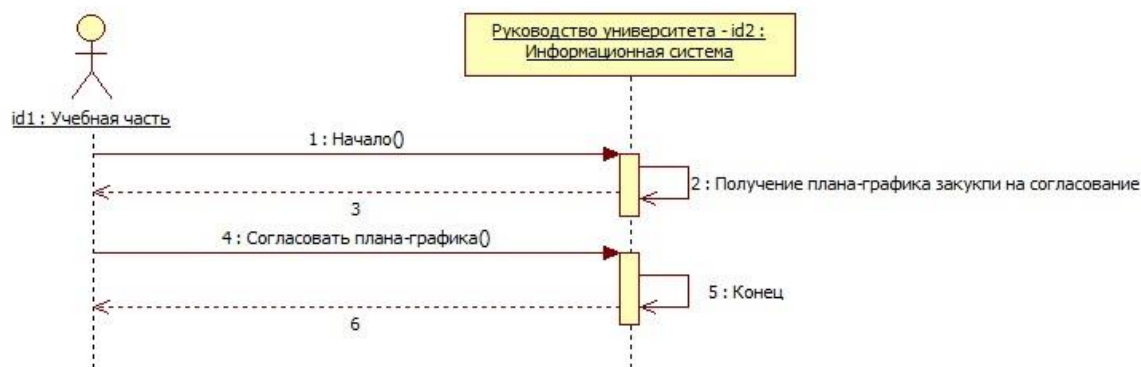


Рисунок 6 – Диаграмма последовательности, моделирующая согласование закупки ПО руководством университета

Для того чтобы грамотно построить ИС, функционирующую эффективно и надежно, необходимо выбрать правильную архитектуру.

Архитектура программного компонента ИС связана с архитектурой самой ИС, и определяется архитектурными шаблонами, например клиент-серверным шаблоном.

Архитектурный шаблон – это набор принципов, высокоуровневая схема, обеспечивающая абстрактную инфраструктуру для семейства систем [7].

При разработке ИС управления использованием ПО предполагается использование трехслойную клиент-серверной архитектуры, которая состоит из трех слоев: слоя представления, слоя предметной области (приложения) и слоя источника данных.

Слой представления содержит графический интерфейс пользователя, который взаимодействует с двумя другими слоями архитектуры. Слой источника данных обеспечивает хранение информации, а слой предметной области обрабатывает логику.

Преимущества трехслойной архитектуры достаточно велики. К примеру, использование такой архитектуры предполагает улучшенную масштабируемость, т.е. возможность расширить наше приложение на несколько серверов, возможность защитить сервер от атак, невысокие запросы к скоростям канала (сети) между терминалами и сервером приложений, а также возмож-

ность быстро переконфигурировать систему при появлении сбоев за счет изолирования уровней друг от друга [8].

Описание слоев трехслойной шаблонной архитектуры представлено в таблице 3.

Таблица 3 – Описание архитектуры ИС управления использованием ПО.

№	Наименование слоя	Описание слоев
1	Слой представления	Отвечает за обработку данных при «общении» пользователя с ИС. Под «общением» пользователя с ИС подразумевается пользовательский ввод данных и пользовательский интерфейс, а также обработка событий.
2	Слой предметной области	Выполняет вычисление на основе вводимых и хранимых данных, проверку всех элементов данных и обработку команд, поступающих от слоя представления, а также выполняет передачу информации слою источника данных.
3	Слой источника данных	Выполняет обмен информацией о состояниях ПО.

Руководствуясь высказываниями шаблона трехслойной архитектуры и принимая во внимание результаты анализа функциональных требований, разрабатывают диаграммы классов, моделирующую структуру управления использованием лицензионного ПО университета ИС концептуального уровня.

Описание назначения классов концептуальной модели по слоям представлена в таблице 4.

Таблица 4 – Назначение классов концептуальной модели

№	Наименование класса	Назначение класса
Слой представления		
1	UI-Admin	Граничный класс, отвечающий за отображение пользовательского интерфейса руководства университета.
2	UI-User	Граничный класс, отвечающий за отображение пользовательского интерфейса сотрудника УИТ.
3	UI-Employ	Граничный класс, отвечающий за отображение пользовательский интерфейс учебного отдела университета.
4	ControlDialog	Управляющий класс, методы которого отвечают за управление приложения в целом.
Слой предметной области		
5	CallModel	Управляющий класс, методы которого отвечают за управление приложения в целом на стороне сервера.
6	ControlConfig	Граничный класс, отвечающий за конфигурацию ИТ-актива.
7	ManagerType	Класс-сущность, содержащий информацию о типах ИТ-актива (например, рабочее место и т.д.).
8	Application	Класс-сущность, содержащий информацию о ПО университета.
9	ManagerConfigITA	Класс-сущность, содержащий информацию о конфигурации лицензионного ПО.
10	CurrentConfigSW	Класс-сущность, содержащий информацию о текущей конфигурации состояния лицензионного ПО.
11	ReportStateSW	Класс-сущность, содержащий информацию о отчет о состояниях лицензионного ПО.
12	FeatureType	Класс-сущность, содержащий информацию о характеристиках типа (например, рабочее место бухгалтера или секретаря).
13	Device	Класс-сущность, содержащий информацию о компьютерах университета.



Продолжение таблицы 4

14	SegmentNet	Класс-сущность, содержащий информацию о сегменте сети.
15	PlanConfigSW	Класс-сущность, содержащий информацию о плановой конфигурации лицензионного ПО.
16	Location	Класс-сущность, содержащий информацию о местоположении ИТ-актива.
17	AcquiSpecSw	Класс-сущность, содержащий актуальную информацию о состоянии лицензионного ПО.
18	RequestChanges	Класс-сущность, содержащий информацию о запросах на изменение состояний ПО.
19	IT-Company	Класс-сущность, содержащий информацию о поставщиках ПО.
20	SW-Price	Класс-сущность, содержащий информацию об ценах ПО.
21	ControlService	Управляющий класс, который отвечает за запросы на изменение состояния ПО и аутентификацию пользователей системы.
22	Person	Класс-сущность, содержащий информацию об авторизованных пользователях ИС.
23	Privilege	Класс-сущность, содержащий информацию о правах доступа для аутентификации пользователей.
Слой источника данных		
12	DataGateway	Граничный класс, отвечающий за поставку данных их БД для объектов предметной области, сохранение изменений.

Согласно сложившимся правилам разработки объектно-ориентированных моделей управления использованием лицензионного ПО, на диаграммах классов не отображают компоненты БД, а отображают только классы слоя источника данных, которые обеспечивают информационное взаимодействие с БД.

На рисунке 7 представлена диаграмма классов, моделирующая отношения классов.

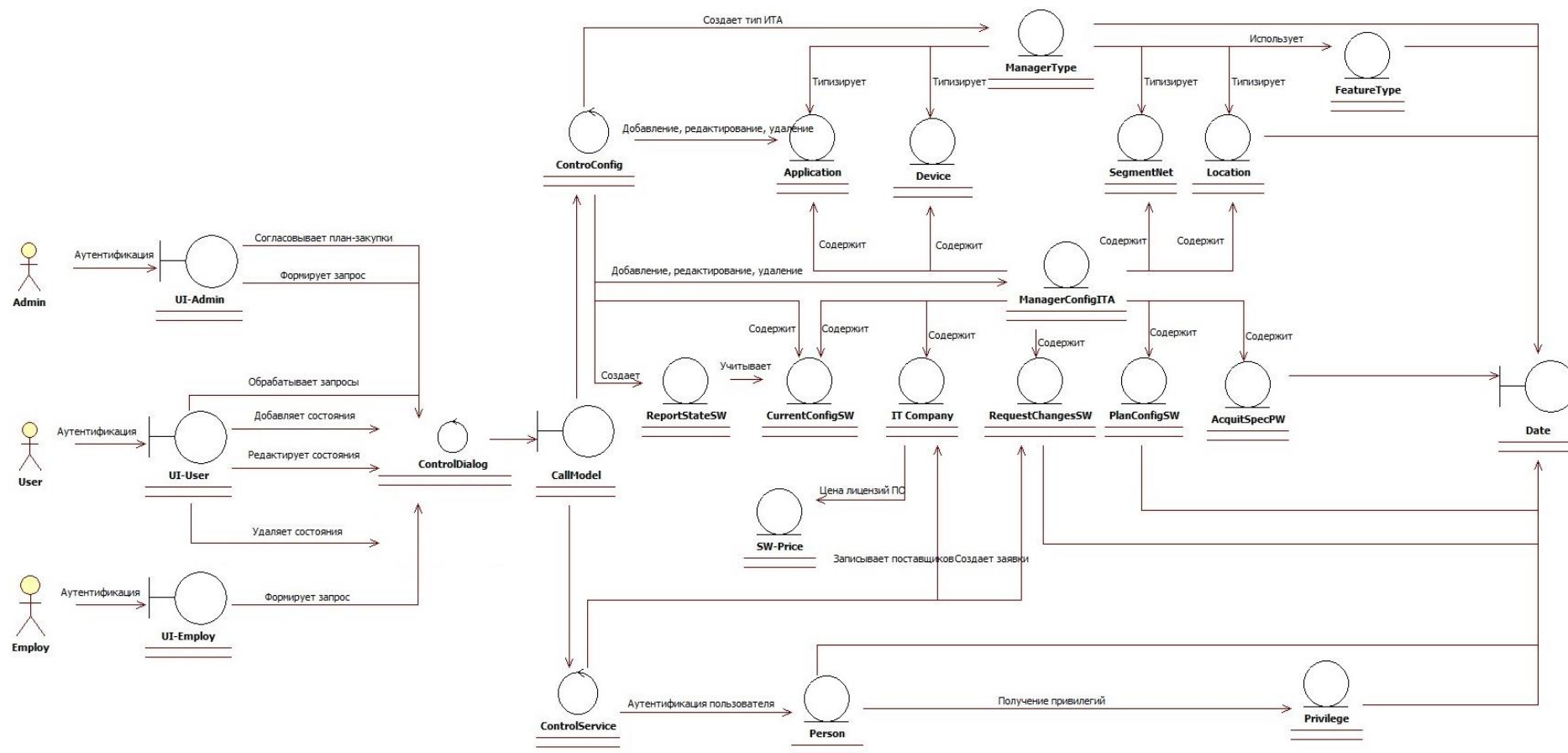


Рисунок 7 – Диаграмма классов, моделирующая отношения классов ИС на концептуальном уровне

## 1.5 Разработка логической модели макета приложения «Мониторинг использования ПО»

Логическая модель должна содержать решение по взаимодействию объектов в рамках выполнения основных функций системы. Решение по взаимодействию выделяются в отдельную модель, которая называется моделью поведения. Основным способом представления модели поведения является диаграмма последовательности.

Как было сказано ранее, диаграмма последовательности служит для представления в графическом виде логической структуры исследуемой предметной области. Логическая модель иллюстрирует сущности и их взаимоотношения между собой.

### 1.5.1 Модель поведения

На рисунках 8-9 представлены диаграммы, моделирующие работу руководства университета и сотрудника УИТ при реализации следующих функций:

1. Авторизация пользователя и формирование запроса на изменение состояния ПО представлены на рисунке 8.

2. Изменение состояния ПО представлено на рисунке 9.

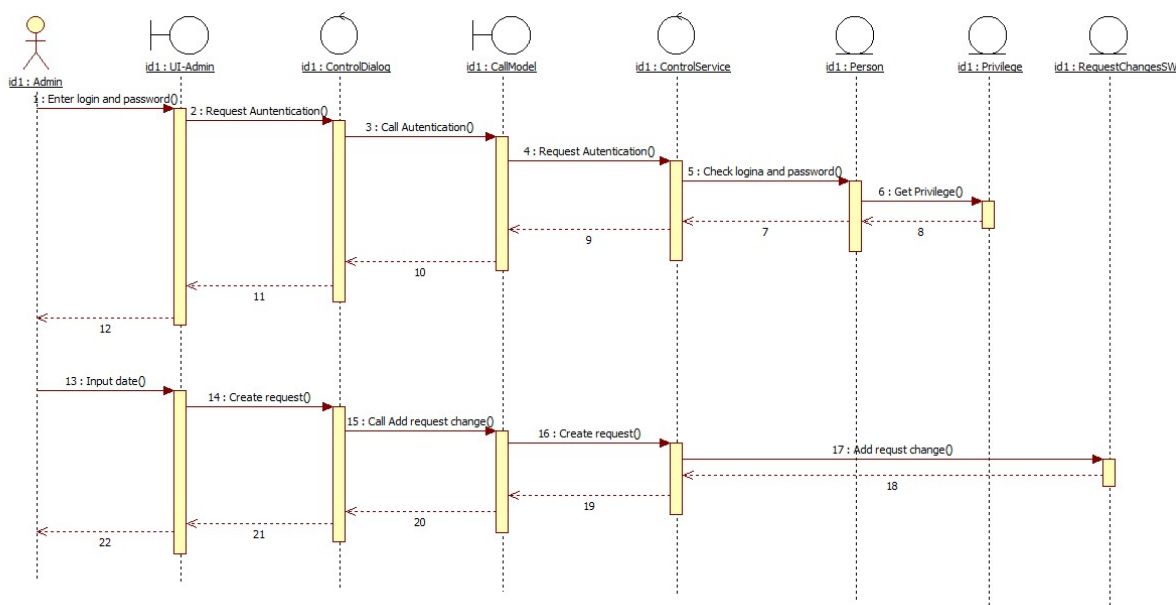


Рисунок 8 – Диаграмма последовательности, моделирующая аутентификацию пользователя и формирование запроса

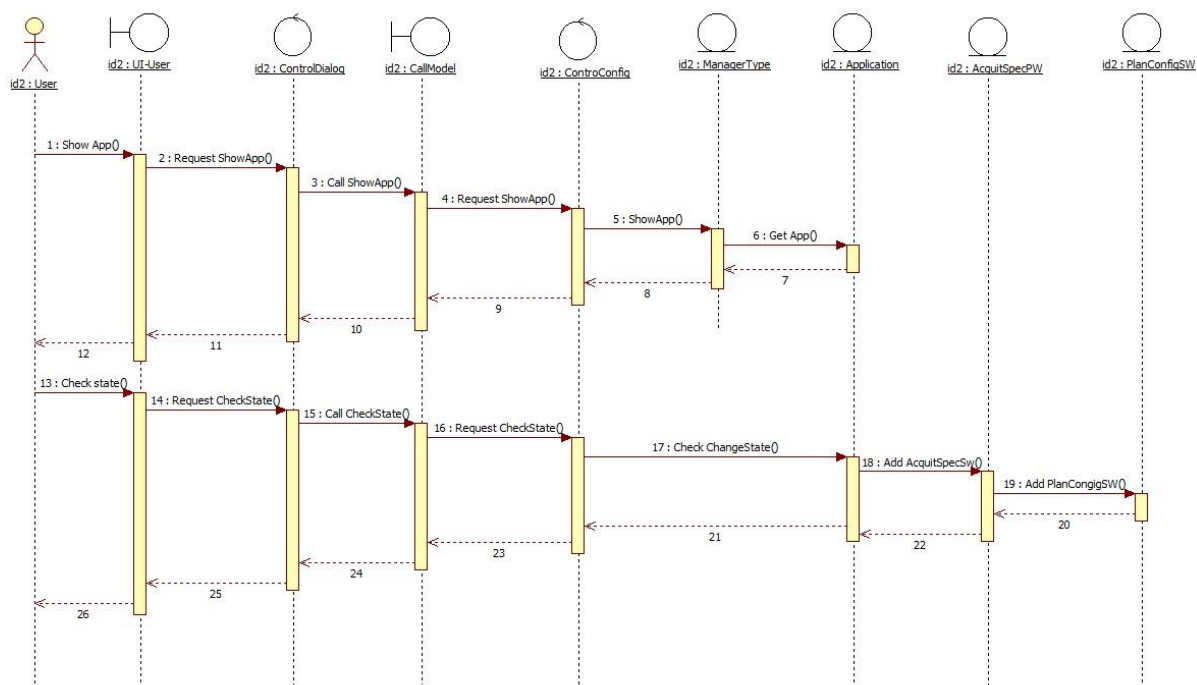


Рисунок 9 – Диаграмма последовательности, моделирующая проверку актуальности состояния лицензионного ПО

### 1.5.2 Модель структуры

Некоторые вопросы решить с помощью моделей «черного ящика» нельзя, так как, имея только отдельные составляющие объекта, невозможно спроектировать ИС. Необходимо установить между объектами определенные связи для достижения поставленной цели. Такая совокупность связей необходимых и достаточных для достижения цели называется структурой системы [9].

Модель структуры отображает связи между компонентами, которые существенны по отношению к созданию ИС. Модель структуры представляется по средствам диаграммы классов, а также содержит наименование свойств и операций, которые представлены на рисунке 10.

Определение операций и объектов выполним посредством шаблонов проектирования, в которых предложены решения по назначению операций. Задание сценариев взаимодействия опирается на шаблон проектирования и на практические объектно-ориентированные решения программистов, которые реализуют разработку ПО.

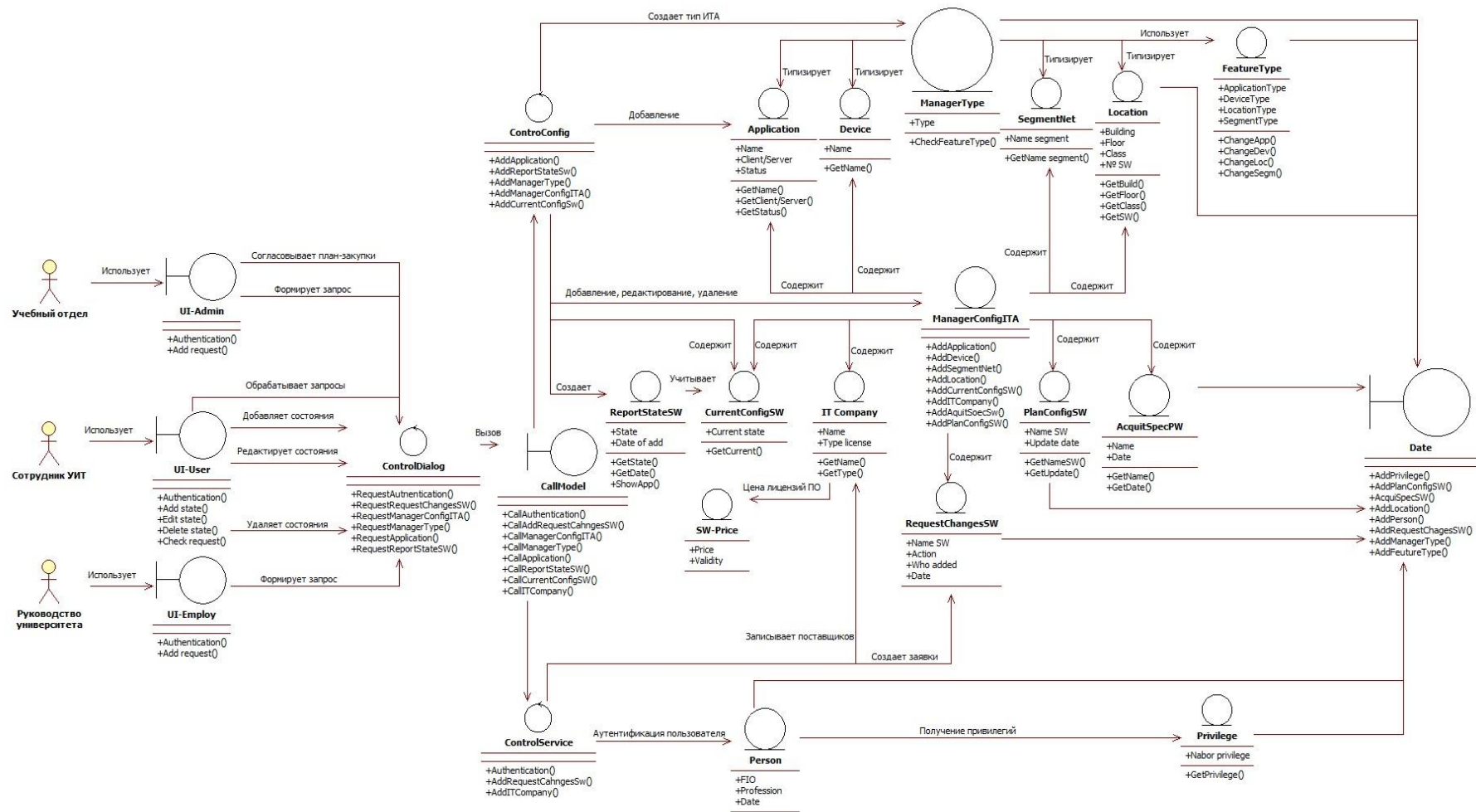


Рисунок 10 – Диаграмма классов, моделирующая отношения классов ПО на логическом уровне

## **2 Технологический раздел**

### **2.1 Применение ИТ-активов университета для целей разработки макета приложения и функционирования ИС**

В университете активно используется технология аппаратной виртуализации серверов. Технология заключается в разделении аппаратных ресурсов одного сервера между большим количеством виртуальных машин. При аппаратной виртуализации на сервере-хосте устанавливается операционная система, а также создаются полностью изолированные друг от друга виртуальные машины, каждая из которых имеет свою операционную систему.

Преимущество аппаратной виртуализации серверов заключается в полноценном разделении ресурсов серверов – ни одна виртуальная машина не может влиять на качество и скорость работы другой [10].

Сервер виртуализации имеет следующие характеристики:

- CPU: 2 x Intel(R) Xeon(R) Silver 4214;
- RAM: 512 GB;
- HDD: 5 x 10 TB SAS;
- NET: SFP+ Intel Original X710DA2BLK 2x10Gb.

Используемая система виртуализации – Xenserver 7.1, работающая на платформе Debian GNU Linux 10. Xenserver обеспечивает стабильное и гарантированное выделение ресурсов для виртуальных машин.

Итоговая система виртуализации состоит из характеристик:

- 24 ядра и 48 потоков;
- 512 Гб оперативной памяти;
- дисковую подсистему объемом 27 Тб из 4 дисков в режиме RAID 5 +1 на горячей замене;
- сетевое подключение по оптоволокну на скорости 10 Гбит.

Сеть вуза построена на основе коммутаторов 3 уровня (маршрутизаторов), с резервированием сетевых подключений, т.е. каждый коммутатор имеет более чем одно подключение к магистральной сети. В сети вуза на данный момент выделено 80 сетей, для каждого подразделения своя сеть.

Рассмотрим виртуальную машину, необходимую для нашей автоматизированной ИС. Одним из главных этапов в разработке автоматизированной ИС является выбор правильного оборудования для размещения ПО ИС.

В рамках нашей работы разрабатываемая система не будет являться высоконагруженной, а, следовательно, не будет требовать серьезных характеристик. Примерные характеристики, которым должна соответствовать виртуальная машина, зададим следующим перечнем:

- оперативная память 8 Гб;
- 4 ядра процессора;
- сетевое подключение 100 Мбит.

В современном мире непросто представить жизнь без компьютерных технологий. Вычислительная техника используется практически во всех сферах человеческой деятельности. Почти у каждого человека имеется персональный компьютер. С его помощью люди работают, получают знания, играют в видеоигры или просто смотрят видеоролики в интернете. Однако для работы компьютера одного системного блока и монитора будет недостаточно, так как нужна операционная система (ОС), которая позволяет взаимодействовать деталям между собой, так включается компьютер, и пользователь видит свой интерфейс в виде рабочего стола [11].

Наиболее популярные типы операционных систем – Linux и Windows. Это два основных типа ОС, между которыми ведется некая условная «битва» за предпочтения пользователей.

Приведем сравнение двух ОС в виде таблицы. Сравнительные характеристики представлены в таблице 5 [12].

Таблица 5 – Сравнительные характеристики ОС

Характеристики	Windows	Linux
Ядро	Микроядро (занимает меньше места, но снижает эффективность работы)	Монолитное ядро (потребляет больше ресурсов)

Продолжение таблицы 5

Надежность	ОС Windows значительно менее надежна, чем ОС Linux.	Высокий уровень надежности и безопасности.
Совместимость	Высокая совместимость.	Малосовместима с другими платформами.
Простота использования	ОС Windows может использоваться как новичок, так и продвинутый пользователь, так как интерфейс очень прост и удобен.	Используя Linux, потребуется гораздо больше времени, чтобы изучить интерфейс данной ОС.
Процесс установки	Проста в процессе установке, требует знаний малого количества команд, но установка ОС Windows займет чуть больше времени, чем ОС Linux.	Сложно настроить, требуется большое количество знаний для освоения различных команд.

Наиболее оптимальным выбором для университета является операционная система Windows, так как эта система легко настраивается, имеет большую совместимость с различными приложениями, а также не требует сложностей в освоении для пользователей.

Для устранения проблем с совместимостью на клиенте и на сервере виртуальной машины операционные системы, желательно, должны быть фирмы. Поэтому на виртуальной машине будет установлена операционная система Microsoft Server 2012, а СУБД – MS SQL Server 2012, на клиенте операционной системой будет Windows 10.

Рабочая станция – это профессиональный компьютер, который используется каких-либо задач (произведения расчетов или научно-исследовательских исследований) [13].

Станции сотрудника УИТ, руководства университета и учебного отдела нужны для создания заявок на изменение состояния ПО, формирование



отчета о состоянии ПО, а также для добавления, редактирования, удаления или просмотра конфигурации ПО и их лицензий.

Минимальные требования к рабочим станциям:

- операционная система Windows 10;
- оперативная память 8 Гб;
- 4 ядра процессора;
- сетевое подключение 100 Мбит.

## **2.2 Разработка макета приложения «Мониторинг использования ПО» (функции пользовательского интерфейса)**

Для разработки макета приложения «Мониторинг использования ПО» необходимо:

1. Выбрать среду для разработки БД, а также разработать саму БД.
2. Выбрать язык программирования.
3. Выбрать среду программирования.

### **2.2.1 База данных**

Для разработки макета необходимо разработать БД, используя СУБД Microsoft SQL Server 2012 Express. Это сильная и надежная система управления данными, которая является бесплатной, обеспечивающая функциональное и надежное хранилище данных для веб-серверов и настольных приложений [14].

Для того чтобы наглядно отобразить логические связи между элементами данных необходимо построить даталогическую модель, исходными данными которой являются знания о ПрО и возможности выбранной СУБД.

Результатом даталогического проектирования является описание логической структуры БД на языке определения данных MS SQL Server Express 2012. Созданная даталогическая модель БД включает все информационные единицы и связи между ними, определение имен информационных единиц, их тип и количественные характеристики (например, длина поля) [15].

Перечень и описание сущностей, используемые в даталогической модели представлены в таблице 6.

Таблица 6 – Описание сущностей

№	Наименование и описание сущности	Наименование и описание атрибута
1	List_of_employees – содержит информацию об авторизованных пользователях.	ID_employee – содержит ID пользователя.
		FIO – содержит ФИО пользователя.
		Profession – содержит должность пользователя.
		Experience – содержит стаж пользователя.
2	List_of_changes – содержит информацию об изменениях.	ID_changes – содержит ID изменения.
		Event – содержит название изменения.
		Date – содержит дату изменения.
		ID_employee – содержит ID пользователя, который внес изменение.
3	ITA-PS – связывает две сущности между собой.	ID_ITA – содержит ID ИТ-актива.
		ID_PS – содержит ID ПС.
4	Computer – содержит информацию о компьютерах, на которые установлены ПС.	ID_computer – содержит ID компьютера.
		Name – содержит название компьютера.
		Serial_number – содержит серийного номера компьютера.
		ID_manufacturer – содержит ID производителя
5	Class_PS – содержит название класса ПС.	ID_class_PS – содержит ID класса ПС.
		Name_class_PS – содержит название ID класса ПС.
6	Use – содержит информацию об области применения.	ID_use – содержит ID области применения.
		Nave_use – содержит название области применения.
7	Computer-PS – связывает две сущности между собой.	ID_computer – содержит ID компьютера.
		ID_PS – содержит ID ПС.

Продолжение таблицы 6

8	PS – содержит информацию о программных средствах (ПС).	ID_PS – содержит ID ПС.
		Name_PS – содержит название ПС.
		Description_function – содержит описание функций ПС.
		Version – содержит версию ПС.
		ID_class_PS – содержит ID класса ПС.
		ID_use – содержит ID области применения ПС.
		ID_status – содержит ID статуса.
		ID_manufacturer – содержит ID производителя.
		ID_state_PS – содержит ID состояния ПС.
9	Location – содержит информацию о местоположении ПС и компьютеров.	ID_location – содержит ID местоположения.
		Building – содержит номер корпуса
		Floor – содержит номер этажа.
		Cabinet – содержит номер аудитории.
10	Status – содержит информацию о правовом статусе ПС.	ID_status – содержит ID правового статуса.
		Status – содержит название правового статуса.
		Validity – содержит срок окончания лицензии.
11	Manufacturer – содержит информацию о производителе.	ID_manufacturer – содержит ID производителя.
		Manufacturer – содержит название производителя.
		Country – содержит страну производителя.
		Year – содержит год производителя.
		Provider – содержит поставщика производителя.
12	State_PP – содержит информацию о состоянии программного продукта (ПП)	ID_state_PP – содержит ID состояния ПП.
		State_PP – содержит состояние ПП.

Продолжение таблицы 6

13	SegmentNet – содержит информацию о сегменте сети.	Id_SegmentNet – содержит ID сегмента сети.
		Name – содержит название сегмента.
14	ИТА – связывает сущность с сущностью Class_ИТА	ID_ИТА – содержит ID ИТ-актива.
		ID_class_ИТА – содержит ID класса ИТ-актива.
15	Class_ИТА – содержит информацию о классе ИТ-актива	ID_class_ИТА – содержит ID класса ИТ-актива.
		Type – содержит тип класса.
16	ИТА-Computer – связывает две сущности между собой.	ID_ИТА – содержит ID ИТ-актива.
		ID_computer – содержит ID компьютера.
17	ИТА-Location – связывает две сущности между собой.	ID_ИТА – содержит ID ИТ-актива.
		ID_location – содержит ID местоположения.
18	PS-Use – связывает две сущности между собой.	ID_PS – содержит ID ПС.
		ID_use – содержит ID области применения.
19	List_of_changes-PS – связывает две сущности между собой	ID_changes содержит ID изменения.
		ID_PS содержит ID ПС.
20	PS-Status – связывает две сущности между собой.	ID_PS – содержит ID ПС.
		ID_status – содержит ID правового статуса.
21	PS-State_PP – связывает две сущности между собой.	ID_PS – содержит ID ПС.
		ID_State_PP – содержит ID статуса ПП.

Схема даталогической модели представлена на рисунке 11.

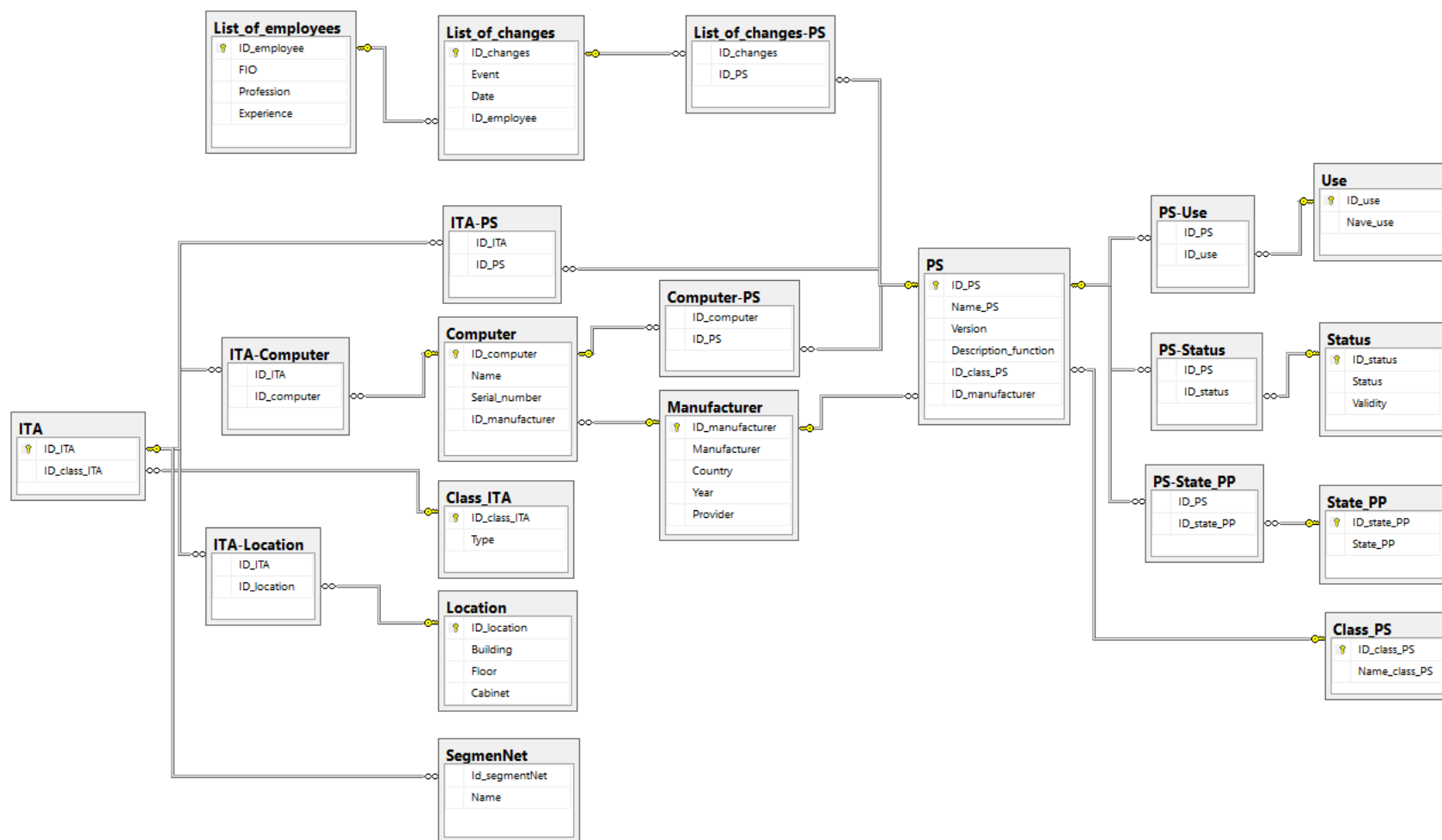


Рисунок 11 – Схема даталогической модели

### **2.2.2 Язык программирования**

Разработка макета приложения «Мониторинг использования ПО» будет выполнена при помощи языка программирования C++. Этот язык является объектно-ориентированным, высокоуровневым и компилируемым, который подходит для разработки различных приложений.

Положительными сторонами при использовании данного языка можно считать поддержку объектно-ориентированного программирования (ООП), что позволяет писать код проще и быстрее; схожий синтаксис с другими популярными языками программирования; огромное количество библиотек и компиляторов, а также возможность работать с данными на уровне, близком к аппаратному [16].

### **2.2.3 Среда разработки**

Для разработки макета приложения был выбран фреймворк для разработки кроссплатформенного ПО на языке C++ для Windows, Linux и Mac OS – Qt Creator 4.10.0.

Выбор именно этого инструмента обусловлен многими факторами. Qt работает более чем на одной аппаратной платформе или операционной системе. Для правильности и надежности кода в Qt используется разные компиляторы C++. Qt обладает встроенной программой Qt Designer, которая позволяет в считанные минуты получить любой наглядный интерфейс. Наличие двух- и трехмерной графики, возможность использовать json и xml файлы, поддержка стандартных протоколов ввода и вывода – это далеко не все отличительные черты Qt.

В настоящее время Qt используется большим количеством разработчиков всего мира, так как это полностью объектно-ориентированная библиотека. Этот инструмент обладает тремя главными факторами – простота, быстрота и мощность [17].

### **2.2.4 Разработка экранных форм пользовательского интерфейса**

Для разработки экранных форм пользовательского интерфейса открываем Qt и заходим в Qt Design, с помощью которого будем создавать формы.

Главное меню приложения представлено на рисунке 12.

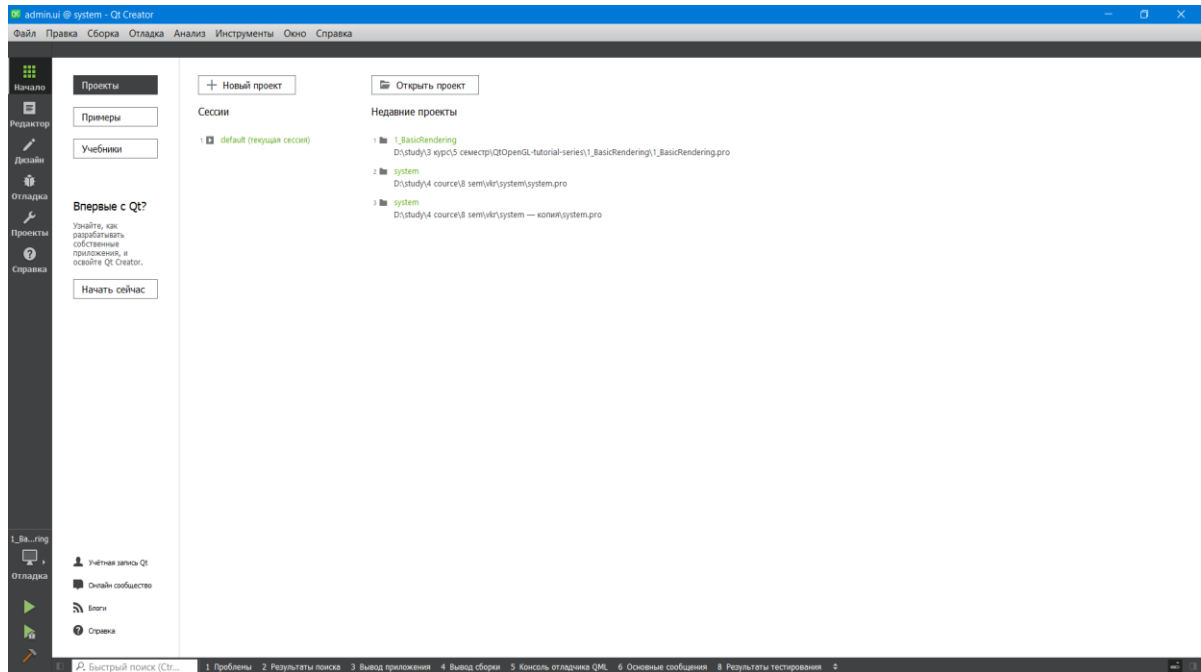


Рисунок 12 – Главный экран

Далее создаем новый проект и заполняем информацию о классе. Результат представлен на рисунке 13.

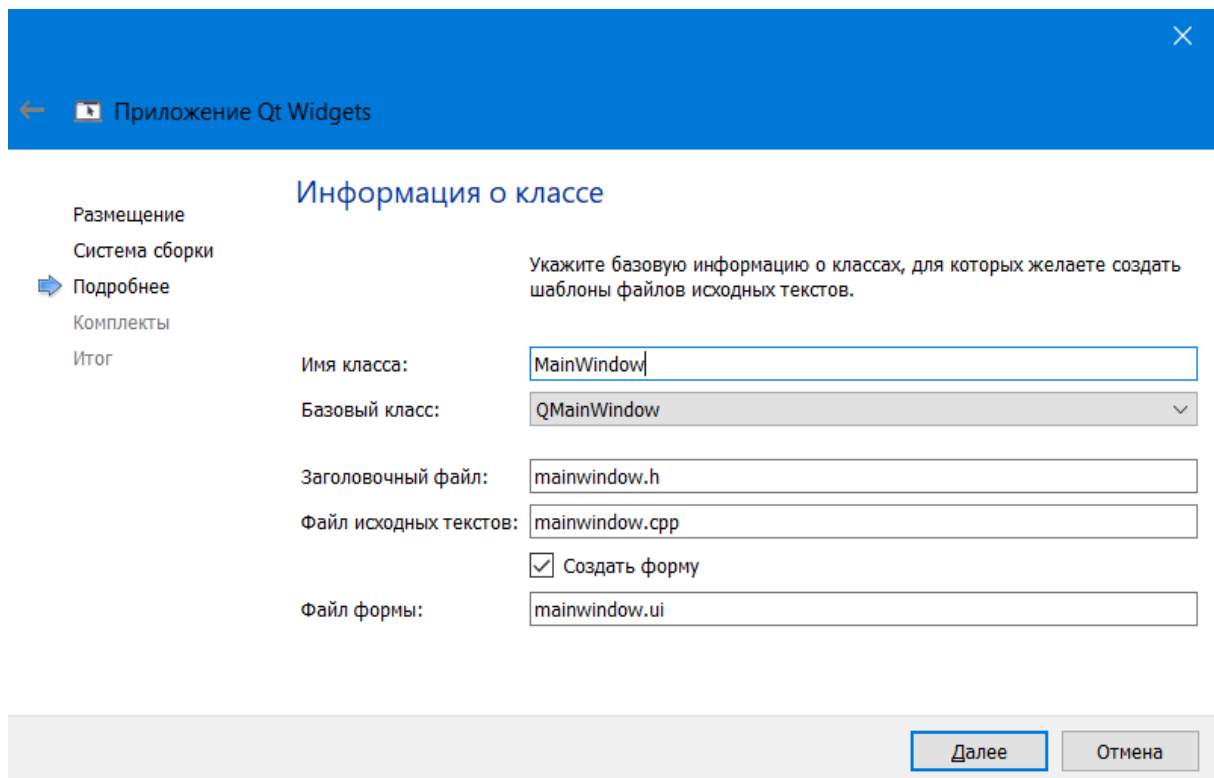


Рисунок 13 – Создание проекта

С помощью ui-файла создадим формы для нашего приложения «Мониторинг состояния ПО». Разберем на примере создание формы для учебного отдела.

Рабочий интерфейс программы представлен на рисунке 14.

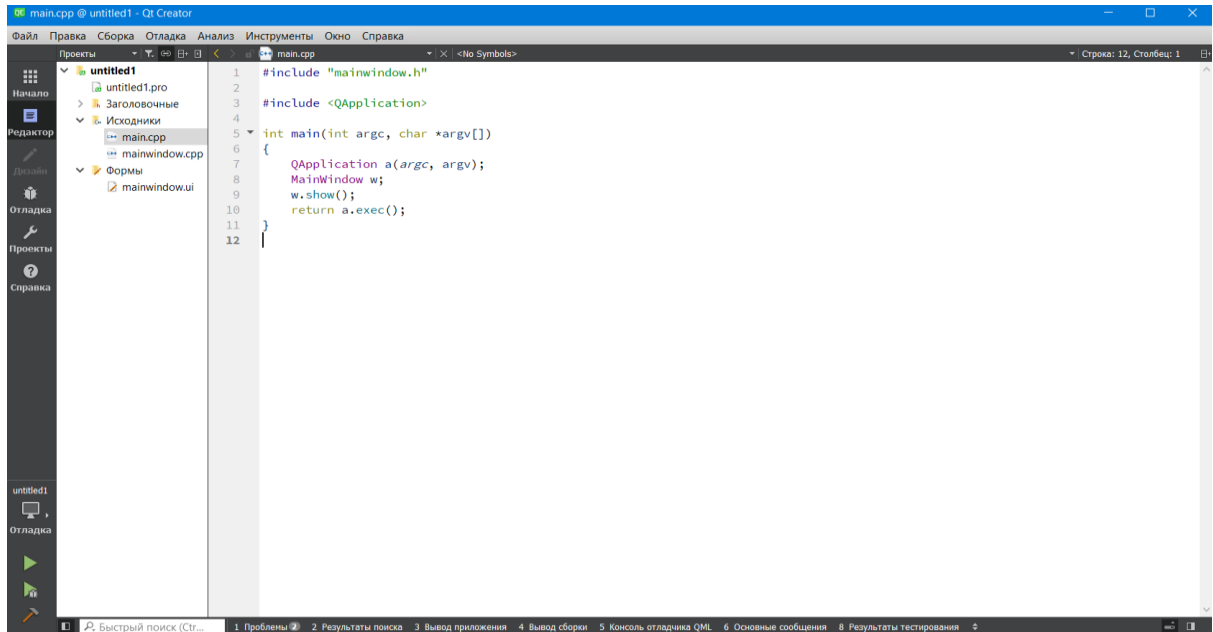


Рисунок 14 – Интерфейс программы

Открываем вкладку «Формы» из дерева виджетов, выбираем нужный ui-файл. В данном примере – это mainwindow.ui. Окно ui-файла представлено на рисунке 15.

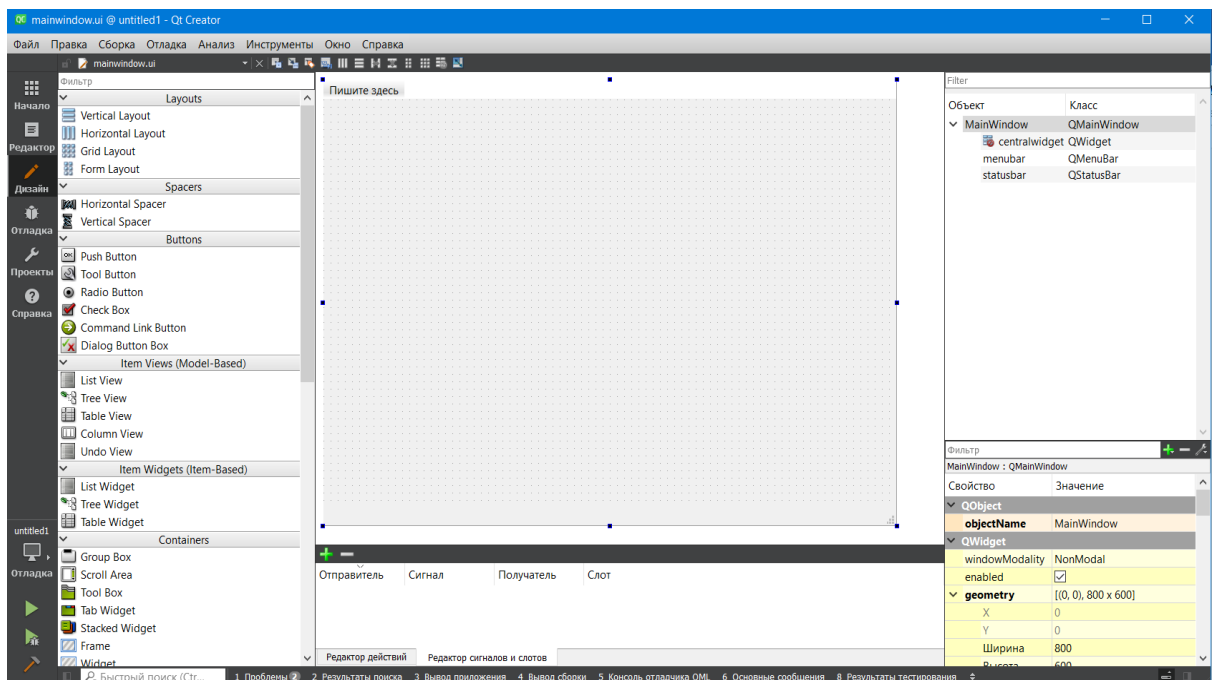


Рисунок 15 – Окно ui-файла



Для создания основной формы учебного отдела потребуются следующие виджеты:

- TableView для создания таблицы, в которую будет выводиться список всех активных запросов пользователей.
- Push Botton для создания кнопки, по которой будет осуществляться открытие формы для заполнения заявки пользователя на изменение состояния ПО.
- Group Vox для визуального выделения области, в которую будут помещены таблица и кнопки формы.

После расположения следующих виджетов, форма со списком заявок представлена на рисунке 16

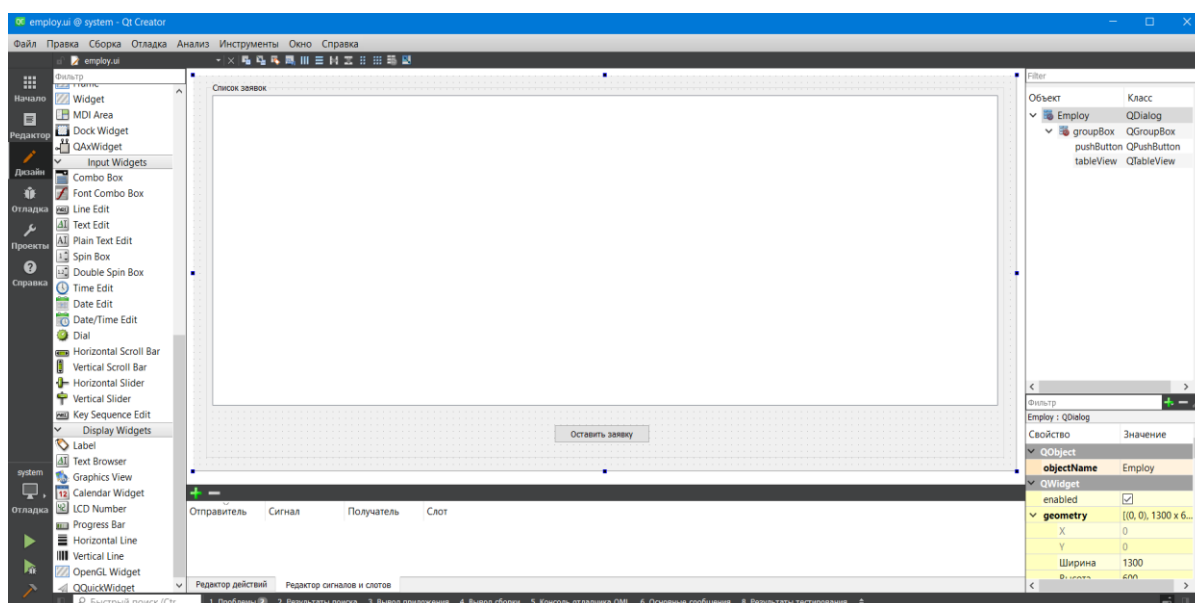


Рисунок 16 – Форма с таблицей активных заявок

Также необходимы виджеты, которые потребуются для создания второй по формы, которая будет открываться при нажатии на кнопку «Создать заявку» в форме с таблицей активных запросов:

- Label для создания надписей на форме (ФИО, должность, название ПО, его назначение и местоположение ИТ-актива).
- Line Edit для графы, в которую будут вводиться с клавиатуры данные пользователя (ФИО, должность, название ПО, его назначение и местоположение ИТ-актива).

- Spin Box для изменения диапазона значений при выборе количества необходимых ПО.
- Group Box для визуального выделения области, в которую будут помещена форма создания заявки.
- Push Botton для создания кнопок «Оставить заявку» и «Отмена», по которым заявка будет добавляться в список заявок или происходить отмена добавления.

После расположения следующих виджетов, форма со списком заявок представлена на рисунке 17.

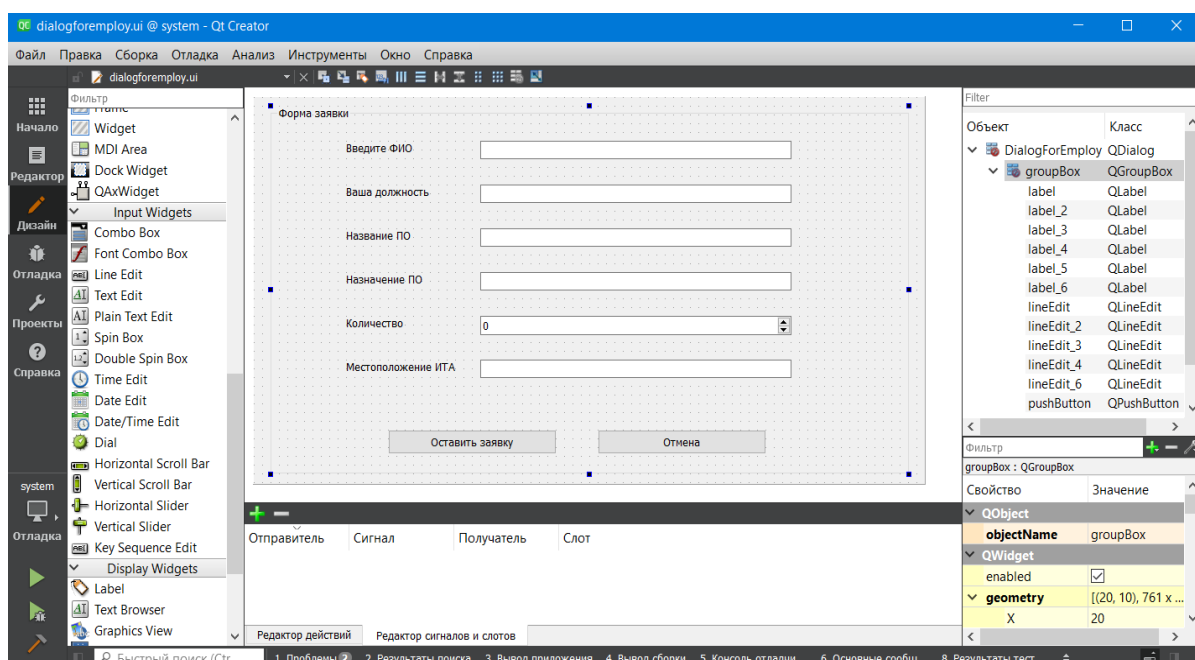


Рисунок 17 – Форма создания заявки

Внешний вид формы при запуске программы представлен на рисунке 18 и 19.

Представитель может найти свой запрос, который он быть может добавил ранее, посмотреть статус рассмотрение запроса сотрудников УИТ, а также добавить новый, тем самым нажать кнопку «Оставить заявку».

При нажатии на кнопку «Создать запрос» появляется окно с формой заполнения запроса, где потребуется ввести ФИО представителя учебного отдела, название необходимого ПО, его назначение, а также количество и место, куда необходимо внедрить или обновить ПО. После заполнения формы необходимо нажать на кнопку «Оставить заявку», если представитель уверен,

что хочет оставить запрос, в противном случае необходимо нажать кнопку «Отмена».

	ФИО	Должность	Название ПО	Назначение ПО	Количество	Местоположение	Состояние заявки
1	Подлесная Валерия Николаевна	лаборант	matcad	учебное	10	Старый корпус, 2 этаж, ауд. 350	одобрена
2	Чухрова Анна Вадимовна	преподаватель	LibreOffice	учебное	1	Старый корпус, 3 этаж, ауд. 310	одобрена

Оставить заявку

Рисунок 18 – Активные заявки

Список заявок

	ФИО	Должность	Название ПО	Назначение ПО	Количество	Местоположение	Состояние заявки
1	Подлесная ...	лаборант					
2	Чухрова Анна ...	преподав					

Добавить устройство

Форма заявки

Введите ФИО: Иванов Иван Иванович

Ваша должность: инженер

Название ПО: simulink

Назначение ПО: учебное

Количество: 20

Местоположение ИТА: Старый корпус, 2 этаж, ауд. 274

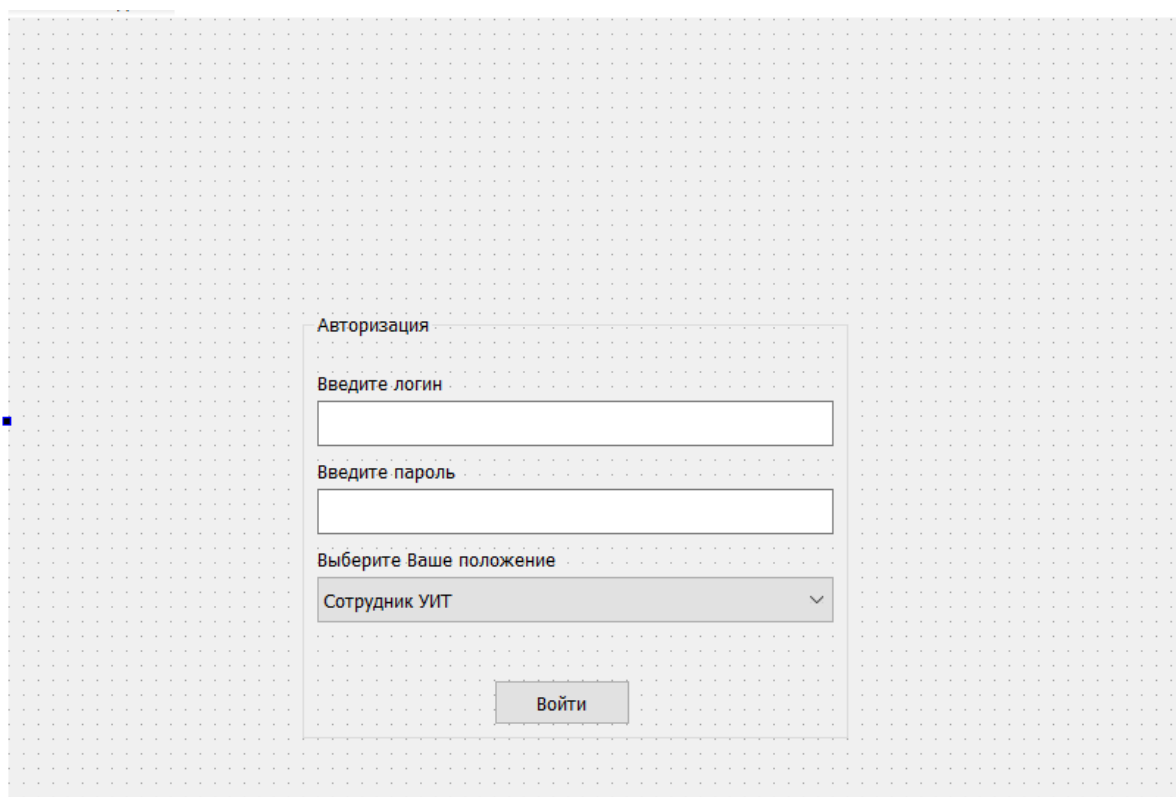
Оставить заявку Отмена

Оставить заявку

Рисунок 19 – Заполнение формы заявки

По аналогии создадим оставшиеся формы для авторизации, для сотрудника УИТ и для руководства университета.

Форма авторизации пользователя представлена на рисунке 18. Внешний вид формы при запуске программы представлен на рисунке 19.



Авторизация

Введите логин

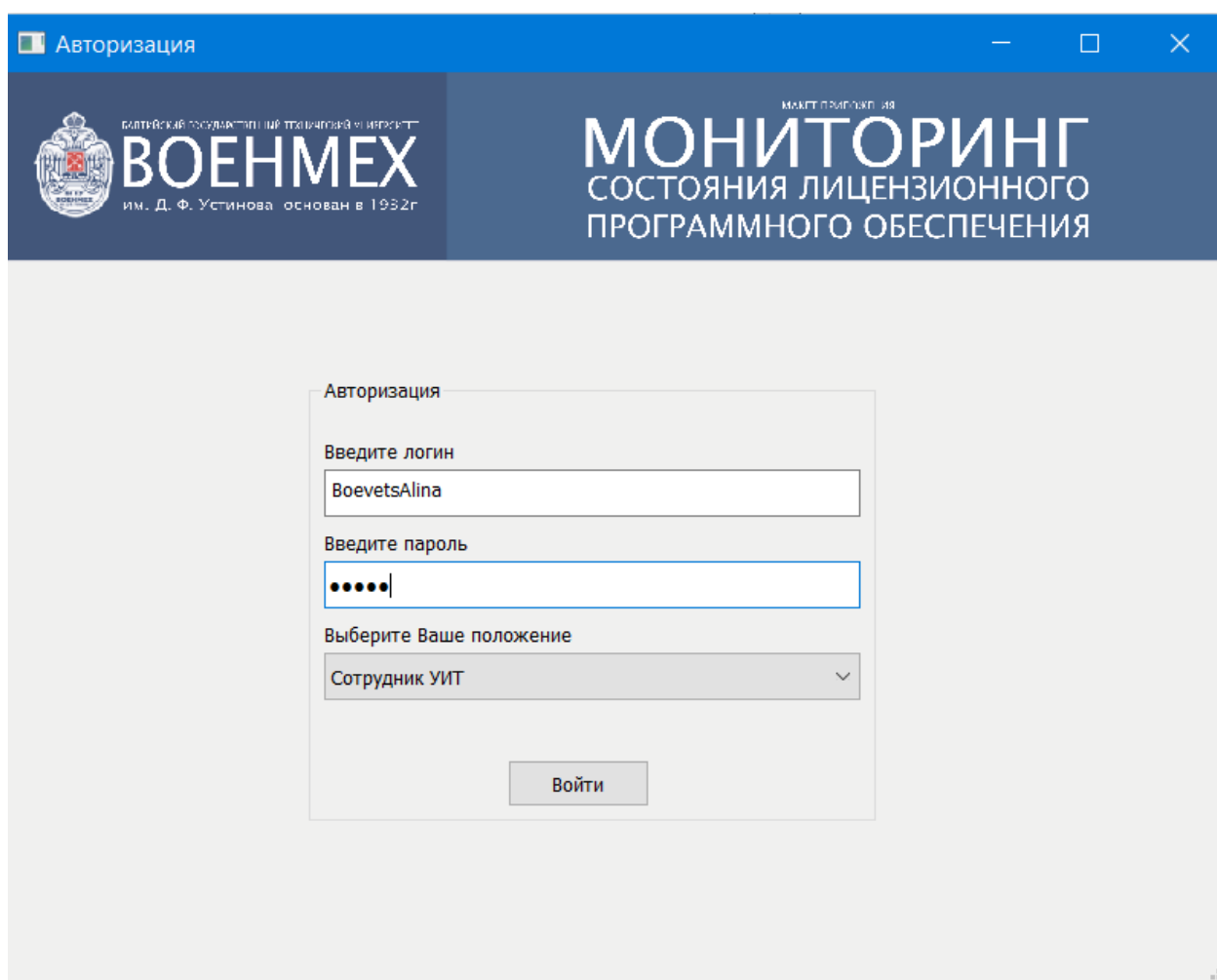
Введите пароль

Выберите Ваше положение

Сотрудник УИТ

Войти

Рисунок 18 – Форма авторизации



Авторизация

Балтийский государственный технический университет  
**ВОЕНМЕХ**  
им. Д. Ф. Устинова основан в 1932г

МАКЕТ ПРИЛОЖЕНИЯ  
**МОНИТОРИНГ**  
СОСТОЯНИЯ ЛИЦЕНЗИОННОГО  
ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ

Авторизация

Введите логин

Введите пароль

Выберите Ваше положение

Сотрудник УИТ

Войти

Рисунок 19 – Авторизация пользователя

Форма сотрудника УИТ представлена на рисунке 20, а внешний вид при запуске программы представлен на рисунке 21.

В форме будет отображаться информация о местоположении ИТ-актива, список самих ИТ-актива, которые расположены в определенных аудиториях, а также подробная информация об ИТ-активе при нажатии в строку таблицы списка.

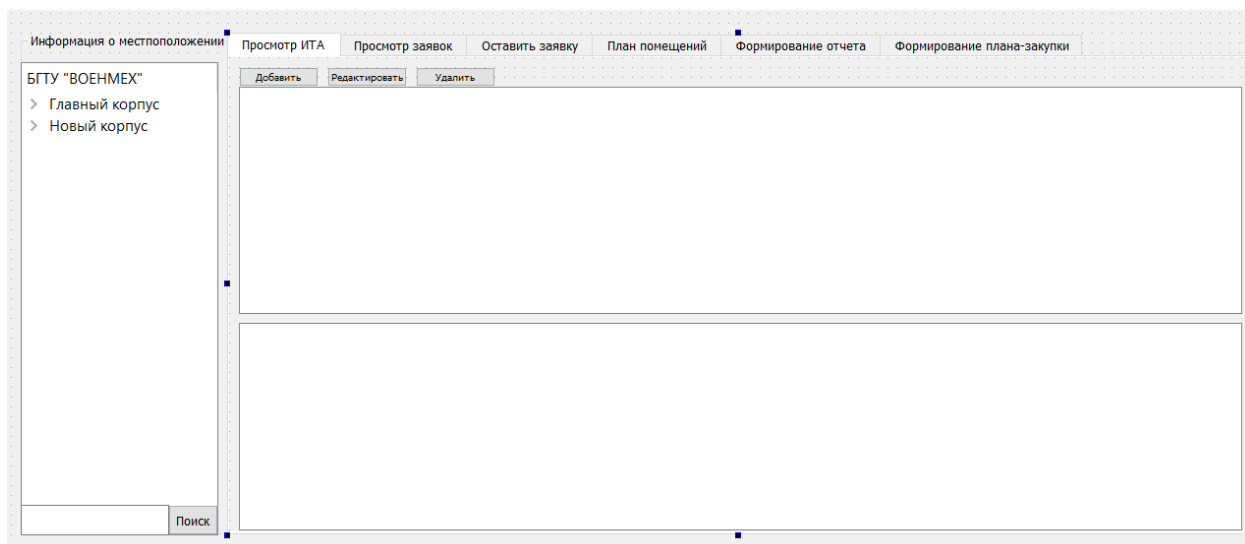


Рисунок 20 – Форма конфигурации ПО

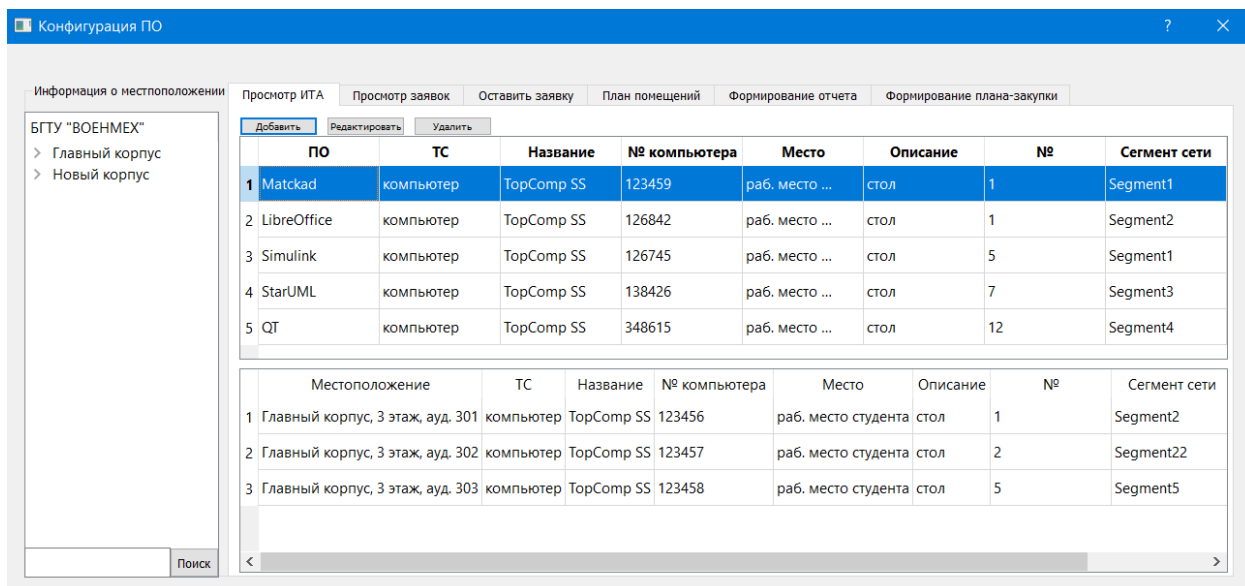


Рисунок 21 – Конфигурация ПО

Система предусматривает редактирование, удаление и добавление ИТ-актива уже в ходе работы при нажатии правой кнопкой мыши на строку с

ПО, а также составление отчетов и планов-графиков, просмотр и создание заявок.

Внешний вид редактирования и удаления ПО при запуске программы представлена на рисунках 23 и 24, а форма удаления соответствует форме добавления ПО.

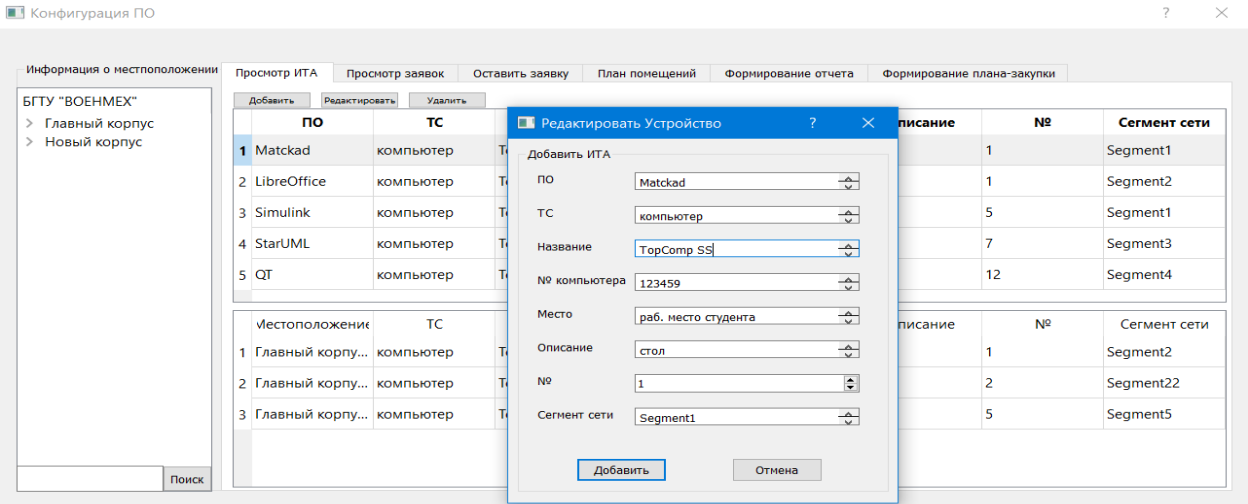


Рисунок 23 – Редактирование записи в конфигурации ПО

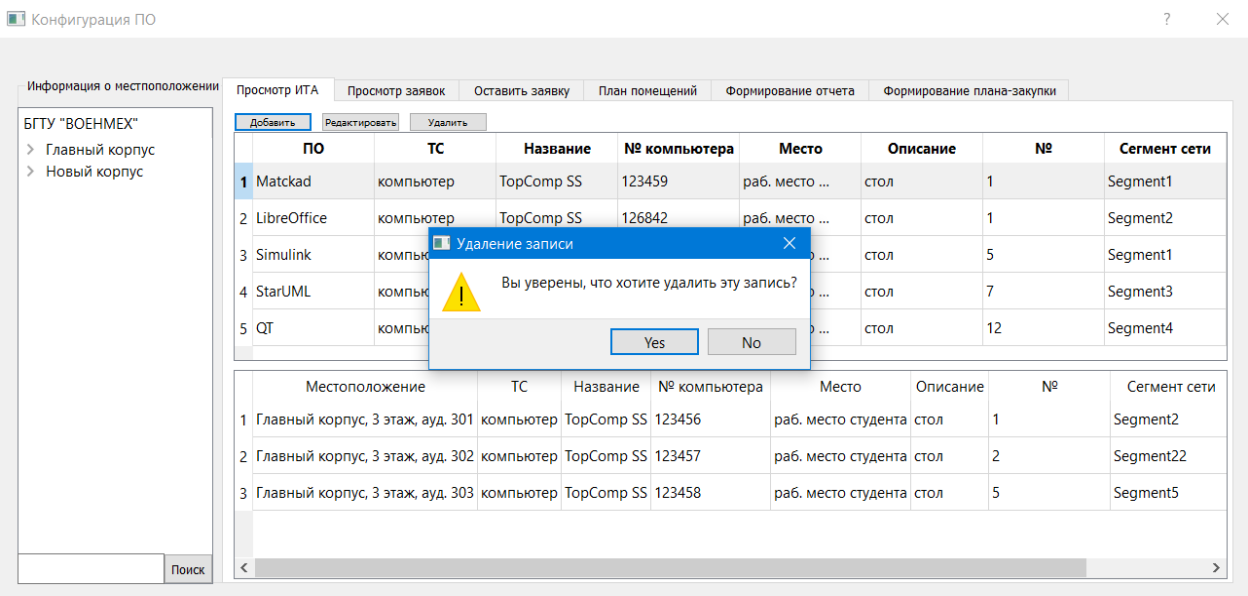


Рисунок 24 – Удаление записи в конфигурации ПО

Форма руководства университета представлена на рисунке 25, а внешний вид при запуске программы представлен на рисунке 26.

В форме будет отображаться информация о поступивших заявках, отчетах и план-графиках. Также представитель руководства университета может оставить заявку на изменение состояния ПО.

Рисунок 25 – Форма руководства университета

Название	Пользователь	Дата
1 План-график ...	Петров П. П.	31.01.2021
2 План-график ...	Петров П. П.	28.02.2021
3 План-график ...	Сидоров А. С.	31.03.2021
4 План-график ...	Петров П. П.	30.04.2021
5 План-график ...	Петров П. П.	31.05.2021

Название ПО	Количество	Статус
1 Matchad	10	продление лицензии
2 Simulink	15	продление лицензии
3 Qt	20	установка ПО
4 C++ Builder	20	установка ПО
5 Matlab	15	продление лицензии
6 Multisim	15	продление лицензии
7 Compas 3D	13	продление лицензии

Рисунок 26 – Экранная форма руководства университета

Исходные тексты, разработанных экранных форм, представлены в приложении А.

### 2.2.5 Обобщенная схема информационной системы

Обобщенная схема ИС представлена в виде неориентированного графа, вершинами которого являются ИТ-актива с установленным лицензионным ПО в различных помещениях, а ребрами являются подсети сервера.

В конечном итоге ИС можно представить в обобщенном виде. Результат представлен на рисунке 27.

Для клиента:  
 ОС Windows 10  
 4 ядра процессора  
 8 Гб оперативной памяти  
 сетевое подключение 100 Мбит  
 Прикладное программное обеспечение  
 (клиентская часть) приложения ИС

— сеть 1  
 — сеть 2

Для сервера:  
 ОС Microsoft Server 2012  
 СУБД MS SQL Server 2012  
 Прикладное программное обеспечение  
 (серверная часть) приложения ИС

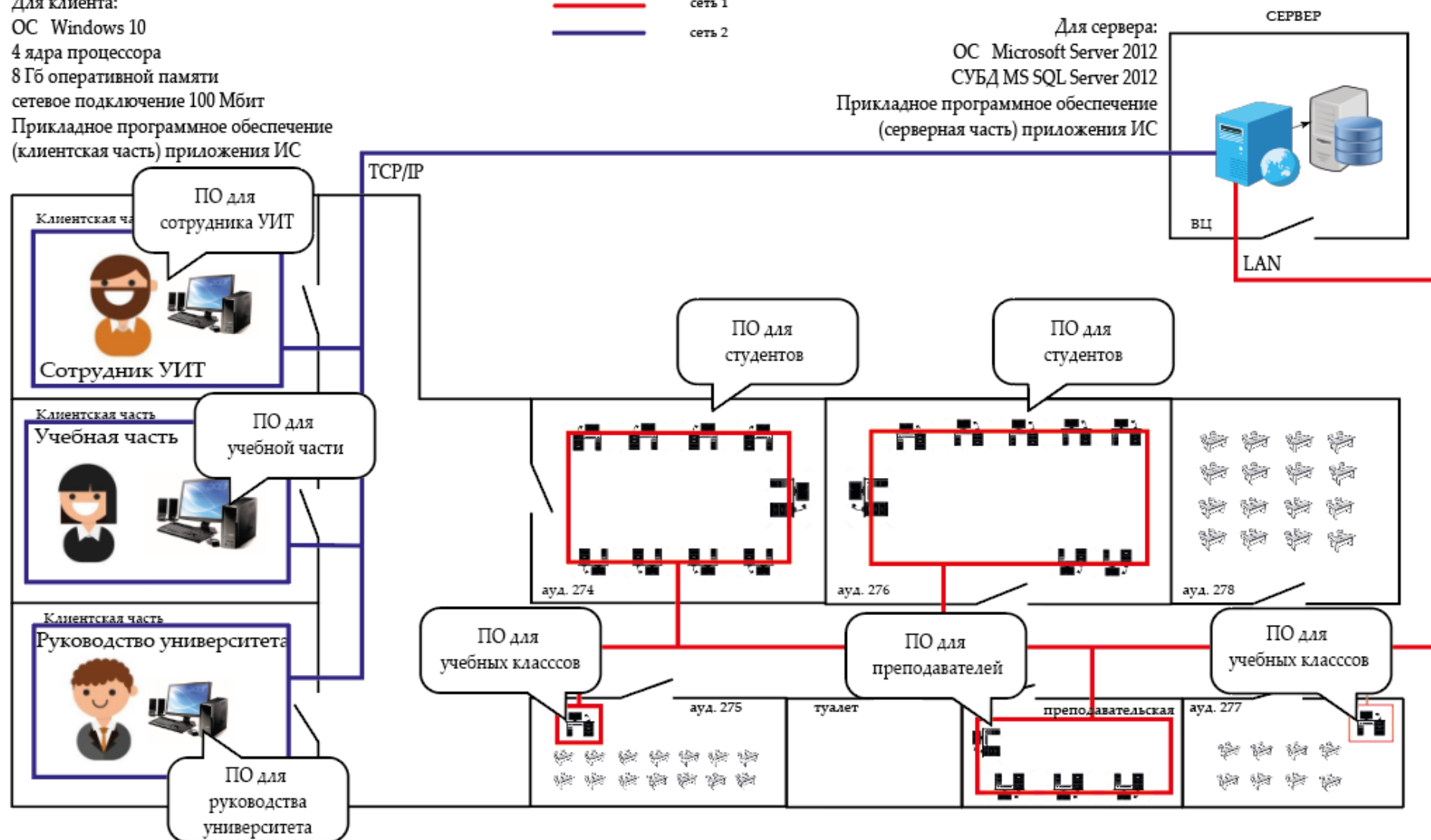


Рисунок 27 – Обобщенная схема ИС



## **ЗАКЛЮЧЕНИЕ**

В процессе выполнения выпускной квалификационной работы обследована и описана предметная область управления использованием лицензионного ПО университета. Разработана концептуальная модель предметной области и по результатам проблемного анализа выявлены проблемы, сдерживающие эффективное управление использованием ПО. В конечном итоге разработаны проектные решения и макет приложения, необходимые для последующей разработки ИС.

Разработка БД для ИС выполнена в СУБД Microsoft SQL Server 2012 Express. Разработка макета приложения «Мониторинг использования ПО» выполнена в среде разработки Qt 4.10.0 на языке C++.

Все требования технического задания выполнены, задачи решены.

Разработанная в дальнейшем на основе проектных решений и макета ИС позволит управлять конфигурацией и использованием лицензионного ПО университета, обеспечит централизованное хранение сведений об изменениях состояний ПО, а также сократит время подготовки заявок на пролонгацию лицензий и закупку дополнительного лицензионного ПО.

## СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. ГОСТ Р 59193-2020 Управление конфигурацией. Основные положения, М.: Стандартинформ, 2020
2. Балтийский государственный университет «ВОЕНМЕХ» [Интернет-ресурс] URL: <https://www.voenmeh.ru/university/sveden/common> (дата обращения 01.04.2021г)
3. Содержание и система управленческой деятельности в вузе [Интернет-ресурс] URL: [https://studref.com/387339/ekonomika/soderzhanie\\_sistema\\_upravlencheskoy\\_deyatelnosti\\_vuze](https://studref.com/387339/ekonomika/soderzhanie_sistema_upravlencheskoy_deyatelnosti_vuze) (дата обращения 01.04.2021г).
4. Диаграмма прецедентов (вариантов использования) UML [Интернет-ресурс] URL: <https://planerka.info/item/diagramma-precedentov-variantov-ispolzovaniya-uml/> (дата обращения 04.04.2021г).
5. Теория и практика UML. Диаграмма деятельности [Интернет-ресурс] URL: [http://www.it-gost.ru/articles/view\\_articles/96](http://www.it-gost.ru/articles/view_articles/96) (дата обращения 04.04.2021г).
6. Основы UML. Диаграммы последовательности [Интернет-ресурс] URL: <https://pro-prof.com/archives/2769> (дата обращения 04.04.2021г).
7. Проектирование информационных систем: конспект лекций [Электронный ресурс] / Н.В. Смирнов; Балт. гос. ун-т. – СПб., 2021.-230с.
8. Трехуровневая архитектура. Достоинства и недостатки [Интернет-ресурс] URL: [https://studopedia.ru/8\\_43424\\_trehurovnevaya-arhitektura-dostoinstva-i-nedostatki.html](https://studopedia.ru/8_43424_trehurovnevaya-arhitektura-dostoinstva-i-nedostatki.html) (дата обращения 24.04.2021г).
9. Что такое виртуализация серверов [Интернет-ресурс] URL: <https://ru.hostings.info/schools/virtualizatsiya-serverov.html> (дата обращения 10.05.2021г).
10. Модель структуры системы [Интернет-ресурс] URL: <https://poisk.ru/s53499t9.html> (дата обращения 10.05.2021г).

11. Компьютерные информационные технологии в современном обществе [Интернет-ресурс] URL: <https://studwood.ru/1699572/informatika/vvedenie> (дата обращения 24.05.2021г).
12. Сравнение Linux и Windows [Интернет-ресурс] URL: [https://skillbox.ru/media/code/vybiraem\\_yazyk\\_programmirovaniya\\_chno\\_nuzhno\\_znat\\_os/](https://skillbox.ru/media/code/vybiraem_yazyk_programmirovaniya_chno_nuzhno_znat_os/) (дата обращения 24.05.2021г).
13. Что такое рабочая станция и как её выбрать [Интернет-ресурс] URL: <https://zen.yandex.ru/media/andpro/chno-takoe-rabochaia-stanciia-i-kak-ee-vybrat-5d67941132335400acb7929d> (дата обращения 24.05.2021г).
14. Microsoft SQL Server 2012 Express с пакетом обновления 1 (SP1) [Интернет-ресурс] URL: <https://www.microsoft.com/ru-ru/download/details.aspx?id=35579>. (дата обращения 28.05.2021г).
15. Нильсен П. Microsoft SQL Server 2005. Библия пользователя [Электронный-ресурс]: Пер. с англ. – М.: ООО «И.Д. Вильямс», 2008. –624с.
16. Выбираем язык программирования: что нужно знать о C++ [Интернет-ресурс] URL: <https://ravesli.com/sravnenie-linux-i-windows-v-chem-raznitsa-i-chno-luchshe/> (дата обращения 28.05.2021г).
17. Шлее М. Qt 5.10. Профессиональное программирование на C++. – СПб.: БХВ-Петербург, 2018.–1072 с.: ил. – (В подлиннике)

## ПРИЛОЖЕНИЕ А

В приложении представлен исходный код экранных форм приложения «Мониторинг состояния лицензионного ПО».

Исходный код представлен в следующих файлах:

– **Mainwindow.cpp**

```
MainWindow::MainWindow(QWidget *parent): QMainWindow(parent)    ,
ui(new Ui::MainWindow)
{
    ui->setupUi(this);
    QPixmap bkgnd("D:/study/4 course/8 sem/vkr/voenmeh.bmp");
    QSize bkgn(800,600);
    bkgnd = bkgnd.scaled(bkgn, Qt::KeepAspectRatio);
    QPalette palette;
    palette.setBrush(QPalette::Background, bkgnd);
    this->setPalette(palette);}

 QVector<QString> MainWindow::input()
{
    QVector<QString> vec;
    vec.push_back(ui -> textEdit -> toPlainText());
    vec.push_back(ui -> lineEdit -> text());
    vec.push_back(ui -> comboBox -> currentText());
    return vec;}

void MainWindow::on_pushButton_clicked()
{
    QVector<QString> pr = input();
    if (pr[2] == "Сотрудник УИТ" && pr[1] == "users" && pr[0] != "")
    {
        hide();
        QMessageBox::information(this, "Help!", "Вы успешно авторизова-
лись!");
        users = new Users(this);
        users->show();}
    else if (pr[2] == "Руководство университета" && pr[1] == "admin" &&
pr[0] != "")
```

```

    {hide();
        QMessageBox::information(this, "Help!", "Вы успешно авторизова-
лись!");
        admin = new Admin(this);
        admin->show();}
else if (pr[2] == "Учебная часть" && pr[1] == "employ" && pr[0] != "")
    {hide();
        QMessageBox::information(this, "Help!", "Вы успешно авторизова-
лись!");
        employ = new Employ(this);
        employ->show();}
else
    QMessageBox::information(this, "Help!", "Авторизация не выполне-
на! Проверьте введенные данные!");}

```

– **Users.cpp**

```

Users::Users(QWidget *parent) : QDialog(parent), ui(new Ui::Users)
{
    ui->setupUi(this);
    deviceDialog = std::unique_ptr<DialogAddDevice>(new DialogAddDe-
vice(this));
    deviceDialog -> setWindowTitle("Добавить Устройство");
    connect(deviceDialog.get(), &DialogAddDevice::accepted, this,
    &Users::setTableDat);
    ui->tableView -> setContextMenuPolicy(Qt::CustomContextMenu);
    connect(ui->tableView, SIGNAL(doubleClicked(QModelIndex)), this,
    SLOT(slotEditRecord()));
    connect(ui->tableView,
    SIGNAL(customContextMenuRequested(QPoint)), this,
    SLOT(slotCustomMenuRequested(QPoint)));
    csvModel = new QStandardItemModel(this);
}

```

```

csvModel->setHorizontalHeaderLabels(QStringList() << "Программного
обеспечение" << "№ компьютера" << "Место" << "№ места" <<"Сегмент
сети");

ui->tableView->setModel(csvModel);
ui->tableView->setSelectionMode(QAbstractItemView::SingleSelection);
ui->tableView->setSelectionBehavior(QAbstractItemView::SelectRows);
ui->tableView->horizontalHeader()->setStretchLastSection(true);
QFile file("D:/study/4 course/8 sem/vkr/system/basa.csv");
if (!file.open(QFile::ReadOnly | QFile::Text) )
{qDebug() << "File not exists";}
else
{QTextStream in(&file);
while (!in.atEnd())
{QString line = in.readLine();
QList <QStandardItem *> standardItemsList;
for (QString item : line.split(";"))
{standardItemsList.append(new QStandardItem(item));}
csvModel->insertRow(csvModel->rowCount(), standardItemsList); }
file.close(); }

csvModel = new QStandardItemModel(this);
csvModel->setHorizontalHeaderLabels(QStringList() << "Местополо-
жение" << "№ компьютера");

ui->tableView_2->setModel(csvModel);
ui->tableView_2-
>setSelectionMode(QAbstractItemView::SingleSelection);
ui->tableView_2-
>setSelectionBehavior(QAbstractItemView::SelectRows);
ui->tableView_2->horizontalHeader()->setStretchLastSection(true);
QFile file2("D:/study/4 course/8 sem/vkr/system/info.csv");
if (!file2.open(QFile::ReadOnly | QFile::Text) )

```

```

    { qDebug() << "File not exists"; }
    else
    { QTextStream in(&file2);
      while (!in.atEnd())
      { QString line = in.readLine();
        QList <QStandardItem *> standardItemsList;
        for (QString item : line.split(";"))
        { standardItemsList.append(new QStandardItem(item)); }
        csvModel->insertRow(csvModel->rowCount(), standardItemsList); }
      file2.close(); } }

void Users::on_pushButton_2_clicked()
{ deviceDialog -> exec(); }

void Users::slotCustomMenuRequested(QPoint pos)
{ QMenu * menu = new QMenu(this);
  QAction * editDevice = new QAction(("Редактировать"), this);
  QAction * deleteDevice = new QAction(("Удалить"), this);
  connect(editDevice, SIGNAL(triggered()), this, SLOT(slotEditRecord()));
  connect(deleteDevice,          SIGNAL(triggered()),          this,
SLOT(slotRemoveRecord()));
  menu->addAction(editDevice);
  menu->addAction(deleteDevice);
  menu->popup(ui->tableView->viewport()->mapToGlobal(pos)); }

void Users::slotEditRecord()
{ std::unique_ptr<DialogAddDevice> addDeviceDialog(new DialogAddDe-
vice(this));
  connect(addDeviceDialog.get(), &DialogAddDevice::inputChanged, this,
&Users::slotUpdateModel);
  addDeviceDialog->setWindowTitle("Редактировать Устройство");
  addDeviceDialog->exec(); }

void Users::slotUpdateModel(QVector<QString> inputData)

```

```

        { setTableData(ui->tableView->selectionModel()->currentIndex().row(), inputData); }

void Users::slotRemoveRecord()
{
    int row = ui->tableView->selectionModel()->currentIndex().row();
    if(row >= 0)
    {
        if (QMessageBox::warning(this, "Удаление записи", "Вы уверены, что хотите удалить эту запись?",
        QMessageBox::Yes | QMessageBox::No) == QMessageBox::No)
        {
            return;
        }
        else
        {
            if(!csvModel->removeRow(row)) {
                QMessageBox::warning(this, "Уведомление", "Не удалось удалить запись\n");
            }
            ui->tableView->setCurrentIndex(csvModel->index(-1, -1));
        }
    }
}

void Users::setTableDat()
{
    setTableData(-1, deviceDialog->GetInputData());
}

void Users::setTableData(int rowIndex, QVector<QString> inputData)
{
    QAbstractItemModel* tableViewModel = ui -> tableView -> model();
    size_t curRowIndex;
    if(rowIndex == -1)
    {
        curRowIndex = tableViewModel->rowCount();
        tableViewModel -> insertRow(curRowIndex);
    }
    else
    {
        curRowIndex = rowIndex;
        for (size_t i = 0; i < inputData.size(); ++i)
        {
            tableViewModel -> setData(tableViewModel -> index(curRowIndex, i), inputData[i]);
        }
    }
}

```

#### – **Employ.cpp**

```

Employ::Employ(QWidget *parent) : QDialog(parent), ui(new Ui::Employ)

```



```

{ ui -> setupUi(this);

    deviceEmploy = std::unique_ptr<DialogForEmploy>(new Dialog-
ForEmploy(this));

    deviceEmploy -> setWindowTitle("Добавить устройство");
    connect(deviceEmploy.get(), &DialogAddDevice::accepted, this,
&Employ::setTableDat);

    csvModel = new QStandardItemModel(this);

    csvModel->setHorizontalHeaderLabels(QStringList() << "ФИО" <<
"Должность" << "Название По" << "Назначение ПО" << "Количество" <<
"Местоположение" << "Состояние заявки");

    ui -> tableView -> setModel(csvModel);
    ui->tableView->setSelectionMode(QAbstractItemView::SingleSelection);
    ui->tableView->setSelectionBehavior(QAbstractItemView::SelectRows);
    ui->tableView->horizontalHeader()->setStretchLastSection(true);
    QFile file("D:/study/4 course/8 sem/vkr/system/request.csv");
    if ( !file.open(QFile::ReadOnly | QFile::Text) )
    { qDebug() << "File not exists"; }
    else
    { QTextStream in(&file);
        while (!in.atEnd())
        { QString line = in.readLine();
            QList <QStandardItem *> standardItemsList;
            for (QString item : line.split(";"))
            { standardItemsList.append(new QStandardItem(item)); }
            csvModel->insertRow(csvModel->rowCount(), standardItemsList); }
        file.close();} }

void Employ::setTableDat()
{ setTableData(-1, deviceEmploy->GetInputData()); }

void Employ::on_pushButton_clicked()
{ deviceEmploy -> exec(); }

```

```

void Employ::setTableData(int rowIndex, QVector<QString> inputData)
{ QAbstractItemModel* tableViewModel = ui -> tableView -> model();
  size_t curRowIndex;
  if(rowIndex == -1)
  { curRowIndex = tableViewModel->rowCount();
    tableViewModel -> insertRow(curRowIndex); }
  else
    curRowIndex = rowIndex;
  for (size_t i = 0; i < inputData.size(); ++i)
    tableViewModel -> setData(tableViewModel -> index(curRowIndex,
i), inputData[i]); }

```

– **DialogForEmploy.cpp**

```

QVector<QString> DialogForEmploy::input()

```

```

{ QVector<QString> vec;
  vec.push_back(ui -> lineEdit -> text());
  vec.push_back(ui -> lineEdit_2 -> text());
  vec.push_back(ui -> lineEdit_3 -> text());
  vec.push_back(ui -> lineEdit_4 -> text());
  vec.push_back(ui -> spinBox -> text());
  vec.push_back(ui -> lineEdit_6 -> text());
  return vec; }

```

```

void DialogForEmploy::on_pushButton_clicked()

```

```

{ emit inputChanged(input());
  this->accept(); }

```

```

void DialogForEmploy::on_pushButton_2_clicked()

```

```

{ this->reject(); }

```

– **DialogAddDevice.cpp**

```

DialogAddDevice::DialogAddDevice(int row, QWidget *parent) :   QDia-
log(parent),   ui(new Ui::DialogAddDevice)
{ if(row == -1)

```

```

{model->insertRow(model->rowCount(QModelIndex()));
  mapper->toLast();}
else
{ mapper->setCurrentModelIndex(model->index(row,0)); } }

```

```

QVector<QString> DialogAddDevice::input()
{ QVector<QString> vec;
  vec.push_back(ui -> textEdit -> toPlainText());
  vec.push_back(ui -> textEdit_2 -> toPlainText());
  vec.push_back(ui -> textEdit_3 -> toPlainText());
  return vec; }

```

```

void DialogAddDevice::on_pushButton_clicked()
{ emit inputChanged(input());
  this->accept();}

```

```

void DialogAddDevice::on_pushButton_2_clicked()
{ this->reject();}

```

#### – **Admin.cpp**

```

Admin::Admin(QWidget *parent) : QDialog(parent), ui(new Ui::Admin)
{ ui->setupUi(this);
  csvModel = new QStandardItemModel(this);
  csvModel->setColumnCount(3);
  csvModel->setHorizontalHeaderLabels(QStringList() << "ФИО" <<
"Должность" << "Название По" << "Назначение ПО" << "Количество" <<
"Местоположение" << "Состояние заявки");
  ui -> tableView_1 -> setModel(csvModel);
  ui -> tableView_1 -
>setSelectionMode(QAbstractItemView::SingleSelection);
  ui -> tableView_1 -
>setSelectionBehavior(QAbstractItemView::SelectRows);
  ui -> tableView_1 ->horizontalHeader()->setStretchLastSection(true);
  QFile file("D:/study/4 cource/8 sem/vkr/system/request.csv");

```

```

if ( !file.open(QFile::ReadOnly | QFile::Text) )
{qDebug() << "File not exists";}
else
{QTextStream in(&file);
    while (!in.atEnd())
    {QString line = in.readLine();
        QList <QStandardItem *> standardItemsList;
        for (QString item : line.split(";"))
        {standardItemsList.append(new QStandardItem(item)); }
        csvModel->insertRow(csvModel->rowCount(), standardItemsList); }
    file.close();}

csvModel = new QStandardItemModel(this);
csvModel->setColumnCount(3);
csvModel->setHorizontalHeaderLabels(QStringList() << "Название" <<
"Пользователь" << "Дата");
    ui -> tableView_2 -> setModel(csvModel);
ui->tableView_2->setSelectionMode(QAbstractItemView::SingleSelection);
ui->tableView_2->setSelectionBehavior(QAbstractItemView::SelectRows);
ui->tableView_2->horizontalHeader()->setStretchLastSection(true);
QFile file2("D:/study/4 course/8 sem/vkr/system/otchet.csv");
if ( !file2.open(QFile::ReadOnly | QFile::Text) )
{qDebug() << "File not exists";}
else
{QTextStream in(&file2);
    while (!in.atEnd())
    {QString line = in.readLine();
        QList <QStandardItem *> standardItemsList;
        for (QString item : line.split(";"))
        {standardItemsList.append(new QStandardItem(item)); }
        csvModel->insertRow(csvModel->rowCount(), standardItemsList); }

```

```

        file2.close();}

csvModel = new QStandardItemModel(this);

csvModel->setHorizontalHeaderLabels(QStringList() << "Location" <<
"№ device" << "PO" << "Status");

ui->tableView_3->setModel(csvModel);
ui->tableView_3->setSelectionMode(QAbstractItemView::SingleSelection);
ui->tableView_3->setSelectionBehavior(QAbstractItemView::SelectRows);
ui->tableView_3->horizontalHeader()->setStretchLastSection(true);
QFile file3("D:/study/4 course/8 sem/vkr/system/inform.csv");
if (!file3.open(QFile::ReadOnly | QFile::Text) )
{qDebug() << "File not exists";}
else
{QTextStream in(&file3);
while (!in.atEnd())
{QString line = in.readLine();
QList <QStandardItem *> standardItemsList;
for (QString item : line.split(";"))
{standardItemsList.append(new QStandardItem(item));}
csvModel->insertRow(csvModel->rowCount(), standardItemsList);}
file2.close();}}

```