|  |  |
| --- | --- |
| Институт (факультет) | Институт информационных технологий |
| Направление подготовки (специальность) | 09.03.02 Информационные системы и технологии |
| Выпускающая кафедра | Математическое и программное обеспечения электронно-вычислительных машин |

**ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА**

|  |  |
| --- | --- |
| Название работы | Разработка подсистемы учета рабочего времени и |
| местоположения сотрудников АО «Сбербанк-Технологии» | |

|  |  |
| --- | --- |
| Студента | Белов Антон Сергеевич |
|  | Ф.И.О. |

ДОПУСТИТЬ К ЗАЩИТЕ

|  |  |
| --- | --- |
| Директор института | Е.В.Ершов |
| Заведующий выпускающей кафедрой | Е.В.Ершов |
| Руководитель выпускной квалификационной работы | А.С.Григоров |
| Нормоконтролер | Л.Н.Виноградова |
| Выпускник | А.С.Белов |

Череповец

2018 г.

Оглавление

[Введение 4](#_Toc512235571)

[1. Описание предметной области 6](#_Toc512235572)

[2. Сравнительный анализ известных технических решений 8](#_Toc512235573)

[3. Выбор технологии, среды и языка программирования 10](#_Toc512235574)

[3.1. Выбор жизненного цикла 11](#_Toc512235575)

[3.2. Выбор подхода к разработке 13](#_Toc512235576)

[3.3. Выбор инструментальных средств 14](#_Toc512235577)

[4. Выбор структур данных для решения поставленной задач 16](#_Toc512235578)

[5. Разработка спецификаций проектируемой системы 18](#_Toc512235579)

[5.1. Построение диаграмм вариантов использования 18](#_Toc512235580)

[5.1.1. Вариант использования «Просмотр данных сотрудника» 19](#_Toc512235581)

[5.1.2. Вариант использования «Создание временной пары» 20](#_Toc512235582)

[5.1.3. Вариант использования «Создание отсутствия» 21](#_Toc512235583)

[5.1.4. Вариант использования «Работа с данными сотрудников» 21](#_Toc512235584)

[5.1.5. Вариант использования «Управление временными данными» 22](#_Toc512235585)

[5.1.6. Вариант использования «Ведение рабочих графиков» 23](#_Toc512235586)

[5.1.7. Вариант использования «Ведение отсутствий» 24](#_Toc512235587)

[5.1.8. Вариант использования «Ведение временных пар» 25](#_Toc512235588)

[5.2. Построение контекстной диаграммы классов 26](#_Toc512235589)

[5.3. Построение диаграмм последовательностей системы 27](#_Toc512235590)

[5.3.1. Диаграмма последовательностей системы для варианта использования «Просмотр данных сотрудника» 28](#_Toc512235591)

[5.3.2. Диаграмма последовательностей системы для варианта использования «Создание временной пары» 30](#_Toc512235592)

[5.3.3. Диаграмма последовательностей системы для варианта использования «Создание отсутствия» 32](#_Toc512235593)

[5.3.4. Диаграмма последовательностей системы для варианта использования «Работа с данными сотрудников» 35](#_Toc512235594)

[5.3.5. Диаграмма последовательностей системы для варианта использования «Ведение временных пар» 40](#_Toc512235595)

[5.3.6. Диаграмма последовательностей системы для варианта использования «Ведение отсутствий» 44](#_Toc512235596)

[5.3.7. Диаграмма последовательностей системы для варианта использования «Ведение рабочих графиков» 47](#_Toc512235597)

[5.4. Построение диаграмм деятельностей сценариев вариантов использования 53](#_Toc512235605)

[5.4.1. Построение диаграммы деятельностей для сценария варианта использования «Управление временными данными» 53](#_Toc512235606)

[6. Проектирование системы 55](#_Toc512235607)

[6.1. Построение структуры системы и диаграмм пакетов 55](#_Toc512235608)

[6.2. Проектирование функционального пакета ПГРВ 56](#_Toc512235609)

[6.2.1. Построение исходной диаграммы классов 56](#_Toc512235610)

[6.2.2. Построение диаграмм последовательностей взаимодействия объектов классов пакета ПГРВ 56](#_Toc512235611)

[6.2.3. Построение диаграммы коопераций пакета ПГРВ 59](#_Toc512235612)

[6.2.4. Построение уточненной диаграммы классов пакета ПГРВ 59](#_Toc512235613)

[6.2.5. Построение детальной диаграммы классов 60](#_Toc512235614)

[6.3. Построение диаграмм компонентов 62](#_Toc512235615)

[6.4. Диаграмма размещения 63](#_Toc512235616)

[7. Кодирование 64](#_Toc512235617)

[8. Тестирование 65](#_Toc512235618)

[9. Администрирование 68](#_Toc512235619)

[Заключение 71](#_Toc512235620)

[ПРИЛОЖЕНИЕ 1. Техническое задание 73](#_Toc512235621)

[ПРИЛОЖЕНИЕ 2. Текст программы 90](#_Toc512235622)

[ПРИЛОЖЕНИЕ 3. Текст прошивки 94](#_Toc512235623)

[ПРИЛОЖЕНИЕ 4. Руководство пользователя 95](#_Toc512235624)

Введение

В 21 веке, в веке информационных технологий, стремительно растёт объём информации, которую с каждым годом обрабатывать становится все сложнее в силу своего объема. Для упрощения работы персонала разрабатываются специализированные информационные системы. Информационная система — это взаимосвязанная совокупность средств, методов и персонала, используемых для хранения, обработки и выдачи информации для достижения цели управления [4].

В Череповце находится один из крупнейших банков на сегодняшний день – Сбербанк, в группу которого входит акционерное общество «Сбербанк-Технологии» (АО «СберТех»), которое занимается разработкой, внедрением и поддержкой программного обеспечения (ПО) для сотрудников и клиентов банка [3].

Во главе отдела стоит ведущий программист, в обязанности которого в первую очередь входит ведение проектов, распределение проектных задач по отделам разработки, обучение младших программистов, встречи с представителями офисов «Сбербанк» по всей России. По мимо этого, ведущий программист выполняет задачи, схожие с функционалом специалиста по управлению сотрудниками:

* ведение присутствий;
* ведение отсутствий;
* ведение рабочих графиков;
* ведение больничных листов.

Существует проблема, связанная с большим объмом информации, которую необходимо вести вручную ведущему программисту.

Актуальность тему выпускной квалификационной работы обусловлена тем, что череповецкий отдел ОА «СберТех» оборудован системой контроля и управления доступом (СКУД), но она не обладает должным функционалом, чтобы полностью автоматизировать процесс контроля учёта рабочего времени сотрудников.

Целью выпускной квалификационно работы является обеспечить череповецкий отдел ОА «СберТех» СКУД с полуавтоматическим ведением временных данных.

Задачей выпускной квалификационно работы является разработка подсистемы, состоящая из клиентского и серверного приложения, а также системы считывания пропускных карт.

1. Описание предметной области

Акционерное общество АО «СберТех» – молодая российская ИТ-компания, входящая в Группу Сбербанк. Дата основания - 6 июля 2011 года. Центральный офис «СберТеха» находится в Москве. Региональные представительства компании открыты в 17 городах России, среди которых: Санкт-Петербург, Новосибирск, Екатеринбург, Ростов-на-Дону, Рязань, Самара, Иннополис, Череповец и другие [3].

Данная компания специализируется на создании программного обеспечения для крупнейшего банка в России и Восточной Европе – Сбербанка, решает сложные инфраструктурные и сервисные задачи, которые улучшают жизнь 70% россиян, а также обеспечивают удобство и доступность сервисов Сбербанка для 110 млн. человек в мире [3].

Весь Череповецкий отдел можно разделить на следующие подразделения:

* программисты, занимающиеся разработкой ПО для стационарных программно-технических комплексов, предназначенных для автоматизированной выдачи и/или приёма наличных денежных средств как с использованием платёжных карт, так и без, а также выполнения других операций;
* программисты, занимающиеся разработкой функций для официального сайта банка, а также для мобильного приложения «Сбербанк Онлайн»;
* тестировщики, занимающиеся тестированием программ;
* центр поддержки пользователей (Call-центр), занимающийся консультацией как работников банка, так и клиентов.

Каждое подразделение в офисе имеет свой кабинет. Помимо этого, офис насчитывает еще 4 дополнительных кабинета:

* переговорная;
* учебный класс;
* столовая;
* комната отдыха.

В Череповецком отделе АО «СберТех» работают 43 человека:

* 2 главных руководителя разработки;
* 4 руководителя разработки;
* 22 java-разработчика;
* 7 javascript-разработчиков;
* 6 инженеров по тестированию;
* 2 системных администратора.

Для допуска в офис сотрудники проходят через СКУД, которая расположена при входе. Данная система представляет из-себя установку, состоящая из турникета и устройства считывания пропускных карт. Турникет становится активным, как только сотрудник приложил к месту считывания свою карту, предварительно зарегистрированную в системе.

Как только сотрудник прикладывает свою карту, система формирует дату и время прихода сотрудника, а также ухода, если тот покидает офис. В любой момент времени ведущий программист (HR Администратор), используя клиентское приложение, установленное на его рабочем компьютере, способен сформировать отчет за период по определенным сотрудникам, в котором будет отражена информация о фактическом и нормативном времени работы, общем времени переработки или недоработки.

1. Сравнительный анализ известных технических решений

На современном рынке представлено не малое количество систем, позволяющих контролировать рабочее время сотрудников. Большинство сервисов достаточно дорогие, сложны в установке и последующем администрировании. Некоторые ИС работают в фоновом режиме на каждом устройстве, что нагружает их.

Сравнительная характеристика известных технических решений приведена в табл.1.

Таблица 1.

Сравнение известных технических решений с разрабатываемым ПО

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Kickidler | StaffCop | ManicTime | СКУД |
| Оформление заявок на отсутствия | - | - | + | - |
| Ведение графиков рабочего времени | - | - | - | - |
| Формирование отчётов | + | + | + | + |
| Взаимодействие с персональными пропускными картами | - | - | + | + |
| Работа с собственными серверами баз данных | - | + | + | + |
| Работа без доступа в интернет | - | - | + | + |
| Контроль рабочего времени | + | + | + | +/- |
| Контроль метаположения | - | - | - | - |

Из табл.1 можно выделить следующие общие недостатки готовых технических решений:

* отсутствие автоматического ведения такой информации, как переработки, задержки и ранние уходы сотрудников, которая влияет в дальнейшем на выплаты заработной платы сотрудникам;
* отсутствие возможности оформление заявок на отсутствие самими сотрудниками, из-за чего HR Администратору придётся вводить данную информацию вручную;
* зависимость от серверов компании, поставляющих ИС, а также отсутствие возможности работы без доступа в Интернет, что может повлечь за собой потерю персональных данных сотрудников и клиентов банка.

Поэтому было принято решение разработать собственную ИС, под названием HRSaveTime, состоящую из двух подпрограмм:

* серверное приложение, способное принимать информацию от считывающих устройств пропускных карт и передавать информацию в базу данных (БД);
* клиентское приложение, позволяющее HR Администратору в полуавтоматическом режиме вести временные данные по сотрудникам отдела, а работникам – оформлять временные отсутствия.

Считывающие устройства будут выполнены на базе микроконтроллера Arduino Nano v3.0 и RFID-считывателя пропускных карт RC522C.

1. Выбор технологии, среды и языка программирования

Согласно техническому заданию (ТЗ), приведенное в прил.1, информационная система разрабатывается в Visual Studio 17 на языке высоко уровня С# с использованием объектно-ориентированного подхода (ООП) под операционную систему Windows.

Программирование микроконтроллера Arduino осуществляется в среде разработки Arduino IDE на С-подобном языке.

* 1. Выбор жизненного цикла

Программный продукт является результатом некоего производственного процесса. Сначала оценивается доступность ресурсов, которые потребуются для реализации ПО. Далее необходимо спроектировать продукт в виде системы, состоящей из многих компонентов, описать функции этих компонентов и их связи между собой, после чего компоненты нужно запрограммировать, автономно отладить, собрать вместе, провести комплексную отладку, подготовить документацию на систему, обучить пользователей, провести опытную эксплуатацию и организовать сопровождение системы на весь период ее эксплуатации. Весь данный процесс называется жизненный цикл программного обеспечения [4].

Жизненный цикл программного обеспечения — ряд событий, происходящих с системой в процессе ее создания и дальнейшего использования. Говоря другими словами, это время от начального момента создания какого либо программного продукта, до прекращения его поддержки после внедрения [2]. Жизненный цикл программного обеспечения можно представить в виде моделей.

Модель жизненного цикла программного обеспечения — структура, содержащая процессы действия и задачи, которые осуществляются в ходе разработки, использования и сопровождения программного продукта [6].

Для разработки описываемой информационной системы учета контроля рабочего времени, будет использоваться модель быстрой разработки Scrum.

Scrum — это набор принципов, на которых строится процесс разработки, позволяющий в жёстко фиксированные и небольшие по времени итерации (организация обработки данных, при которой действия повторяются многократно) называемые спринтами (sprints), предоставлять конечному пользователю работающее ПО с новыми возможностями, для которых определён наибольший приоритет [4].

Спринт — жёстко фиксирован по времени итерация,в ходе которой происходит функциональное развитиепрограммного обеспечения. Длительность одного спринта длится от 2 до 4 недель.

В методологии Scrum разработчики имеют свои роли. В классическом Scrum существует 3 базовых роли:

* ответственный за продукт (Product owner, PO) – член Scrum-команды, являющийся связующим звеном между командой разработки и заказчиком. Задача PO — максимальное увеличение ценности разрабатываемого продукта и работы команды;
* скрам мастер (Scrum master, SM) является «служащим лидером». Задача Scrum Master — помочь команде максимизировать ее эффективность посредством устранения препятствий, помощи, обучении и мотивации команде, помощи PO;
* команда разработки (Development team, DT) состоит из специалистов, производящих непосредственную работу над производимым продуктом.

Методология Scrum состоит из 5 этапов:

* создание списка требований к продукту;
* планирование спринта и создание списка требований к спринту;
* работа над спринтом;
* тестирование и демонстрация продукта;
* ретроспектива - планирование следующего спринта.
  1. Выбор подхода к разработке

Можно выделить два основных подхода к проектированию систем: структурный подход и объектно-ориентированный подход.

Для проектирования данной системы будет использован объектно-ориентированный подход. Объектно-ориентированный подход будет давать следующие преимущества:

1. уменьшение сложности программного обеспечения (ПО);
2. повышение надежности ПО;
3. обеспечение возможности модификации отдельных компонент программ без изменения остальных компонент;
4. обеспечение возможности повторного использования отдельных компонент ПО.

Для проектирования системы был выбран унифицированный язык моделирования UML – это унифицированный графический язык моделирования для описания, визуализации, проектирования и документирования объектно-ориентированных систем [4]. UML призван поддерживать процесс моделирования программ на основе объектно-ориентированного подхода, организовывать взаимосвязь концептуальных и программных понятий, отражать проблемы масштабирования сложных систем. Модели на UML используются на всех этапах жизненного цикла программ, начиная с бизнес-анализа и заканчивая сопровождением системы.

* 1. Выбор инструментальных средств

Для разработки клиентского и серверного программного обеспечения (ПО) выбрана среда разработки Visual Studio 17, т.к. она обладает следующими технологиями и функциями, упрощающим процесс разработки [1]:

* IntelliTrace, который автоматически ведет журналы выполнения кода, запоминает и отмечает события в таймлайнах, которые далее можно просматривать, перемещаться и проверять состояния. Технология [IntelliTest](https://msdn.microsoft.com/en-us/library/dn823749(v=vs.140).aspx) позволяет проанализировать код и автоматически сгенерировать соответствующие модульные тесты;
* CodeMap позволяет оптимизировать работу с большим числом строк кода;
* возможность установить дополнительное ПО – Xamarin, позволяющее создать мультиплатформенное приложение.

Язык программирования – C#, так как этот язык обладает следующими особенностями:

* полная поддержка классов и объектно-ориентированного программирования, включая наследование интерфейсов и реализаций, виртуальных функций и перегрузки операторов;
* встроенная поддержка автоматической генерации XML-документации;
* автоматическое освобождение динамически распределенной памяти;
* возможность отметки классов и методов атрибутами, определяемыми пользователем;
* полный доступ к библиотеке базовых классов .NET, а также легкий доступ к Windows API;

указатели и прямой доступ к памяти, если они необходимы.

Для программирования микроконтроллеров Arduino, используемых в считываемых устройствах пропускных карт, будет использоваться Arduino IDE по ряду причин:

* простота интерфейса;
* всплывающие подсказки во время написания кода;
* простота установки дополнительных библиотек;
* простая работа с портами компьютера, что позволяет быстро программировать сразу нескольких микроконтроллеров;
* встроенное тестирование микроконтроллеров.

1. Выбор структур данных для решения поставленной задач

Вся информация о сотрудниках будет храниться в базе данных, располагающейся на локальном сервере АО «СберТех», в соответствующих таблицах. При этом некоторые данные, например пароли, будут храниться в зашифрованном виде. Для реализации решаемой проблемы была выбрана реляционная модель данных.

Реляционная модель данных (РМД) — это совокупность данных, состоящая из набора двумерных таблиц. При табличной организации данных отсутствует иерархия элементов. Строки и столбцы могут быть просмотрены в любом порядке, поэтому высока гибкость выбора любого подмножества элементов в строках и столбцах [2].

Любая таблица в реляционной базе состоит из строк, которые называют записями, и столбцов, которые называют полями. На пересечении строк и столбцов находятся конкретные значения данных.

В реальном проектировании структуры базы данных применяется семантическое моделирование. Семантическое моделирование представляет собой моделирование структуры данных, опираясь на смысл этих данных. В качестве инструмента семантического моделирования используются различные варианты диаграмм сущность-связь (ER - Entity-Relationship) [4].

ER-диаграмма ИС HRSaveTime приведена на рис.1.



Рис. 1. ER-диаграмма ИС HRSaveTime

1. Разработка спецификаций проектируемой системы

Модель системы (или какого-либо другого объекта) – это формальное описание системы, в котором выделены основные объекты, составляющие систему, и отношения между этими объектами.

Спецификация требований программного обеспечения – это законченное описание поведения разрабатываемой программы. Включает ряд пользовательских сценариев (англ. use cases), которые описывают все варианты взаимодействия между пользователями и программным обеспечением.

Пользовательские сценарии являются средством представления функциональных требований. В дополнение к пользовательским сценариям, спецификация также содержит нефункциональные требования, которые налагают ограничения на дизайн или реализацию (такие как требования производительности, стандарты качества, или проектные ограничения).

* 1. Построение диаграмм вариантов использования

Диаграмма прецедентов (диаграмма вариантов использования) в UML – диаграмма, отражающая отношения между актерами и прецедентов и являющаяся составной частью модели прецедентов, позволяющей описать систему на концептуальном уровне [1].

Для отражения модели прецедентов на диаграмме используются:

* рамки системы (англ. systemboundary) — прямоугольник с названием в верхней части и эллипсами (прецедентами) внутри. Часто может быть опущен без потери полезной информации;
* актёр (англ. actor) — стилизованный человечек, обозначающий набор ролей пользователя (понимается в широком смысле: человек, внешняя сущность, класс, другая система), взаимодействующего с некоторой сущностью (системой, подсистемой, классом). Актёры не могут быть связаны друг с другом (за исключением отношений обобщения/наследования);
* прецедент — эллипс с надписью, обозначающий выполняемые системой действия (могут включать возможные варианты), приводящие к наблюдаемым актёрами результатам. Надпись может быть именем или описанием (с точки зрения актёров) того, «что» делает система (а не «как»). Имя прецедента связано с непрерываемым (атомарным) сценарием — конкретной последовательностью действий, иллюстрирующей поведение. В ходе сценария актёры обмениваются с системой сообщениями. С одним прецедентом может быть связано несколько различных сценариев.

Диаграмма вариантов использования подсистемы учета рабочего времени и местонахождения сотрудников АО Сбербанк Технологии представлен на рис. 2.

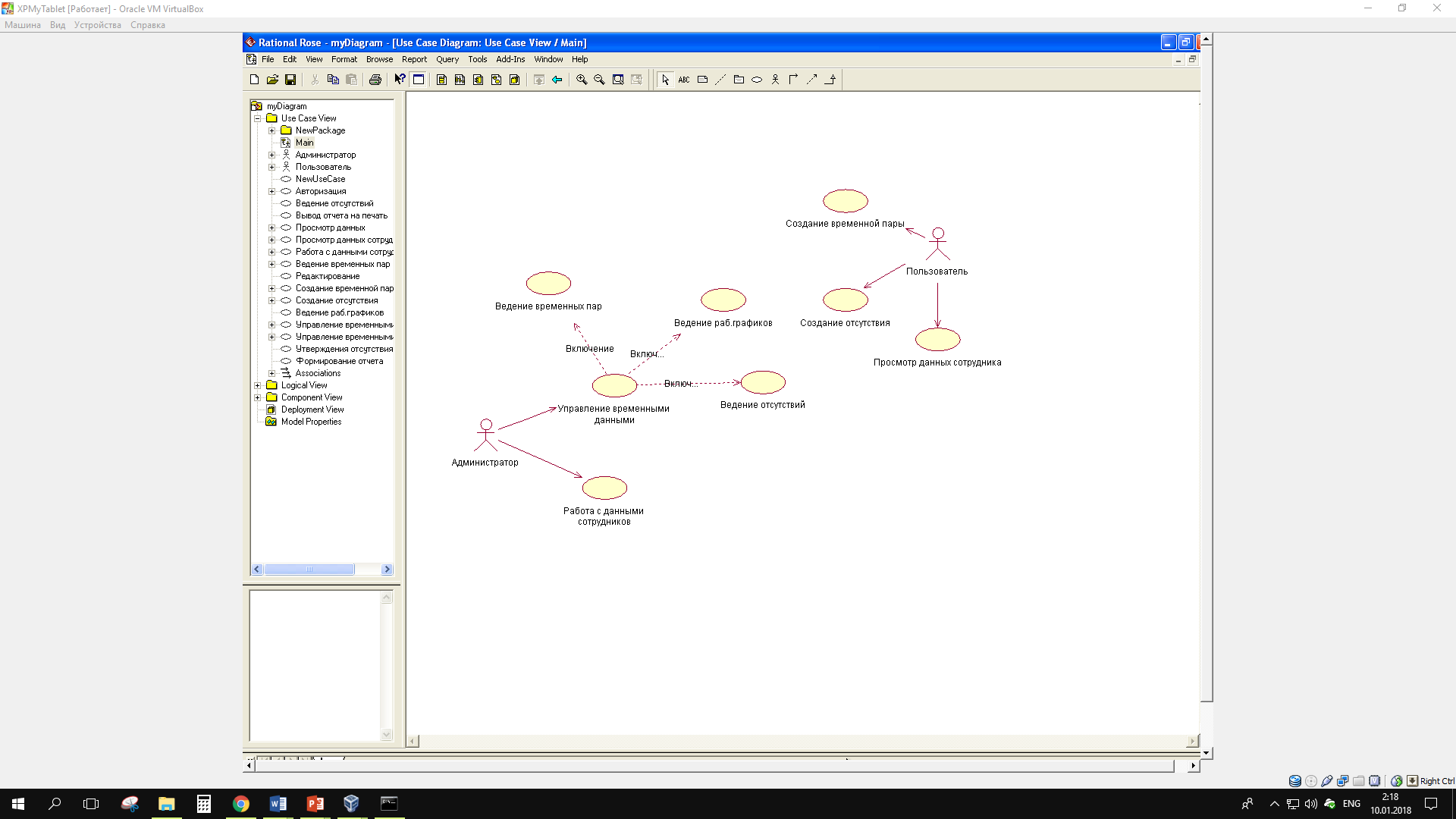


Рис. 2. Диаграмма вариантов использования подсистемы учета рабочего времени и местонахождения сотрудников АО СберТех

С подсистемой могут взаимодействовать два пользователя:

1. HR Администратор (специалист по управлению персоналом), котором доступен следующий функционал:
   1. работа с данными сотрудников (создание пользователя, просмотр, редактирование, удаление);
   2. управление временными данными (ведение отсутствий, создание графиков рабочего времени, ведение заявок на отсутствие, ведение временных пар);
2. Сотрудник АО «СберТех», которому доступен следующий функционал:
   1. просмотр данных сотрудника (ФИО, контактные данные, должность и т.п.);
   2. создание отсутствия;
   3. создание временной пары.
      1. Вариант использования «Просмотр данных сотрудника»

Данный вариант использования описывает взаимодействие сотрудника с профилем пользователя.

Данный вариант использования начинает выполняться, когда пользователь открывает свой или чужой профиль в системе (табл.2).

Типичный ход событий

Таблица 2

Типичный ход событий

|  |  |
| --- | --- |
| Действие исполнителя | Отклик системы |
| 1. Пользователь выбирает свой или чужой профиль в системе | 2. Открывается профиль пользователя, на котором отражены фото, ФИО, должность, стаж, направление, контактные данные. |

* + 1. Вариант использования «Создание временной пары»

Данный вариант использования описывает взаимодействие с пропускными пунктами в офисе АО «СберТех».

Типичный ход событий

Данный вариант использования начинает выполняться, когда пользователь прикладывает пропуск к датчику на пропускном пункте (табл.3).

Таблица 3

Типичный ход событий

|  |  |
| --- | --- |
| Действие исполнителя | Отклик системы |
| 1. Пользователь подносит пропуск к датчику на пропускном пункте | 2. В Базу данных (БД) системы записывается фактическое время прихода на рабочее место, если пользователь подносит пропуск к датчику, который расположен на входе в кабинет, либо ухода, если датчик расположен на выходе из кабинета, а так же наименование кабинета |

Альтернативные потоки

Пропуск не зарегистрирован. В этом случае на пропускном пункте загорится красная лампочка, символизируя ошибку. В следствии чего в БД вместо временной пары запишется ошибка.

* + 1. Вариант использования «Создание отсутствия»

Данный вариант использования описывает взаимодействие с подпрограммой создания отсутствия.

Типичный ход событий

Данный вариант использования начинает выполняться, когда пользователь выберет пункт меню «Создание отсутствия» (табл.4).

Таблица 4

Типичный ход событий

|  |  |
| --- | --- |
| Действие исполнителя | Отклик системы |
| 1. Пользователь запускает «Создание отсутствия» | 2. Открывается окно для заполнения заявки на отсутствие |
| 3. Пользователь заполняет поля ввода | 4. Система автоматически сохраняет внесенные изменения в локальный файл, хранящийся на персональном ПК. |
| 5. Пользователь отправляет заявку на согласование | 6. В БД создалась новая запись |
|  | 7. Система оповещает HR Администратора о новой заявке |

* + 1. Вариант использования «Работа с данными сотрудников»

Данный вариант использования описывает взаимодействие HR Администратора с данными сотрудников АО «СберТех».

Типичный ход событий

Данный вариант использования начинает выполняться, когда HR Администратор переходит в профиль сотрудника и нажимает кнопку редактировать (табл.5).

Таблица 5

Типичный ход событий

|  |  |
| --- | --- |
| Действие исполнителя | Отклик системы |
| 1 | 2 |
| 1. HR Администратор, находясь в профиле сотрудника, запускает процесс редактирования | 2. Открывается окно для редактирования профиля |

Таблица 5. Продолжение

|  |  |
| --- | --- |
| 1 | 2 |
| 3. HR Администратор заполняет поля ввода | 4. Система автоматически сохраняет внесенные изменения в локальный файл, хранящийся на персональном ПК. |
| 5. HR Администратор сохраняет изменения | 6. В БД обновляются данные по сотруднику |

* + 1. Вариант использования «Управление временными данными»

Данный вариант использования описывает взаимодействие HR Администратора с подпрограммой управления временными данными.

Типичный ход событий

Данный вариант использования начинает выполняться, когда HR Администратор выберет пункт меню «Управление временными данными» (табл. 6).

Таблица 6

Типичный ход событий

|  |  |
| --- | --- |
| Действие исполнителя | Отклик системы |
| 1. HR Администратор запускает «Управление временными данными» | 2. Открывается окно со списком возможных действий |

* + 1. Вариант использования «Ведение рабочих графиков»

Данный вариант использования описывает взаимодействие HR Администратора с подпрограммой управления временными данными – Создание рабочих графиков.

Типичный ход событий

Данный вариант использования начинает выполняться, когда HR Администратор в «Управление временными данными» выбирает «Создание рабочих графиков» (табл. 7).

Таблица 7

Типичный ход событий

|  |  |
| --- | --- |
| Действие исполнителя | Отклик системы |
| 1. HR Администратор запускает «Создание рабочих графиков» | 2. Система открывает список всех персональных графиков рабочего времени (ПГРВ) |
| 3. HR Администратор запускает процесс создания нового графика | 4. Система открывает окно для создания перерыва |
| 5. HR Администратор заполняет поля ввода |  |
| 6. HR Администратор сохраняет изменения | 7. Система записывает в БД данные о новом перерыве |
|  | 8. Система открывает окно для создания однодневного графика рабочего времени (ОГРВ) |
| 9. HR Администратор заполняет поля ввода |  |
| 10. HR Администратор сохраняет изменения | 11. Система записывает в БД данные о новом ОГРВ |
|  | 12. Система открывает окно для создания персонального графика рабочего времени (ПГРВ) |
| 13. HR Администратор заполняет поля ввода |  |
| 14. HR Администратор сохраняет изменения | 15. Система записывает в БД данные о новом ПГРВ |

* + 1. Вариант использования «Ведение отсутствий»

Данный вариант использования описывает взаимодействие HR Администратора с подпрограммой управления временными данными – Ведение отсутствий.

Данный вариант использования начинает выполняться, когда HR Администратор в «Управление временными данными» выбирает «Ведение отсутствий», либо, перейдя в профиль любого сотрудника, нажать Ведение отсутствий (табл.8).

Типичный ход событий

Таблица 8

Типичный ход событий

|  |  |
| --- | --- |
| Действие исполнителя | Отклик системы |
| 1. HR Администратор запускает «Ведение отсутствий» | 2. Система открывает список всех зарегистрированных отсутствий по всему персоналу. Если вход в Ведение отсутствий осуществлялся через профиль конкретного сотрудника, то система выдаст список отсутствий только данного человека |

* + 1. Вариант использования «Ведение временных пар»

Данный вариант использования описывает взаимодействие HR Администратора с подпрограммой управления временными данными – Ведение отсутствий.

Типичный ход событий

Данный вариант использования начинает выполняться, когда HR Администратор в «Управление временными данными» выбирает «Ведение отсутствий», либо, перейдя в профиль любого сотрудника, нажать Ведение отсутствий (табл.9).

Таблица 9

Типичный ход событий

|  |  |
| --- | --- |
| Действие исполнителя | Отклик системы |
| 1. HR Администратор запустил «Ведение временных пар» | 2. Система открывает список персонала |
| 3. HR Администратор выбирает нужного сотрудника | 4. Система выводит все временные пары по данному сотруднику |
| 5. HR Администратор запускает процесс редактирования нужной временной пары | 6. Система выводит форму для заполнения данных |
| 7. HR Администратор вносит изменения |  |
| 8. HR Администратор сохраняет данные | 9. В БД обновляются данные по сотруднику |

Альтернативные потоки

Если в системе по конкретному сотруднику нет данных об отсутствиях, то система выдаст соответствующее сообщение.

* 1. Построение контекстной диаграммы классов

Класс - это группа сущностей (объектов), обладающих сходными свойствами, а именно, данными и поведением. Отдельный представитель некоторого класса называется объектом класса или просто объектом [4].

Диаграмма классов служит для представления статической структуры модели системы в терминологии классов объектно-ориентированного программирования [1]. Диаграмма классов может отражать, в частности, различные взаимосвязи между отдельными сущностями предметной области, такими как объекты и подсистемы, а также описывает их внутреннюю структуру и типы отношений. На данной диаграмме не указывается информация о временных аспектах функционирования системы.

Цель основного варианта использования системы –ведение временных данных сотрудников.

Определим возможные обобщения: HR Администратор и сотрудники.

Исходя из описания предметной области и диаграммы вариантов использования, основными классами являются «Данные пользователей», которые связаны с классами «Графики рабочего времени» и «Временные пары».

В разрабатываемой системе необходимо реализовать возможность создания сотрудникам отсутствия:

* отгул;
* прогул;
* больничный;
* командировка.

При этом, чтобы сами работники могли подавать заявку на отсутствие HR Администратору, а тот в свою очередь мог принимать решения: согласовывать её или нет.

Помимо этого, необходимо реализовать взаимодействие с аппаратной частью: пропускными системами, которые будут передавать в систему данные о сотруднике: кто пришел/ушел, во сколько и откуда.

На основе графиков, система должна понимать, что если человек не пришел в рабочее время, то ему необходимо зафиксировать прогул.

Полученная концептуальная модель предметной области представлена на рис.3.

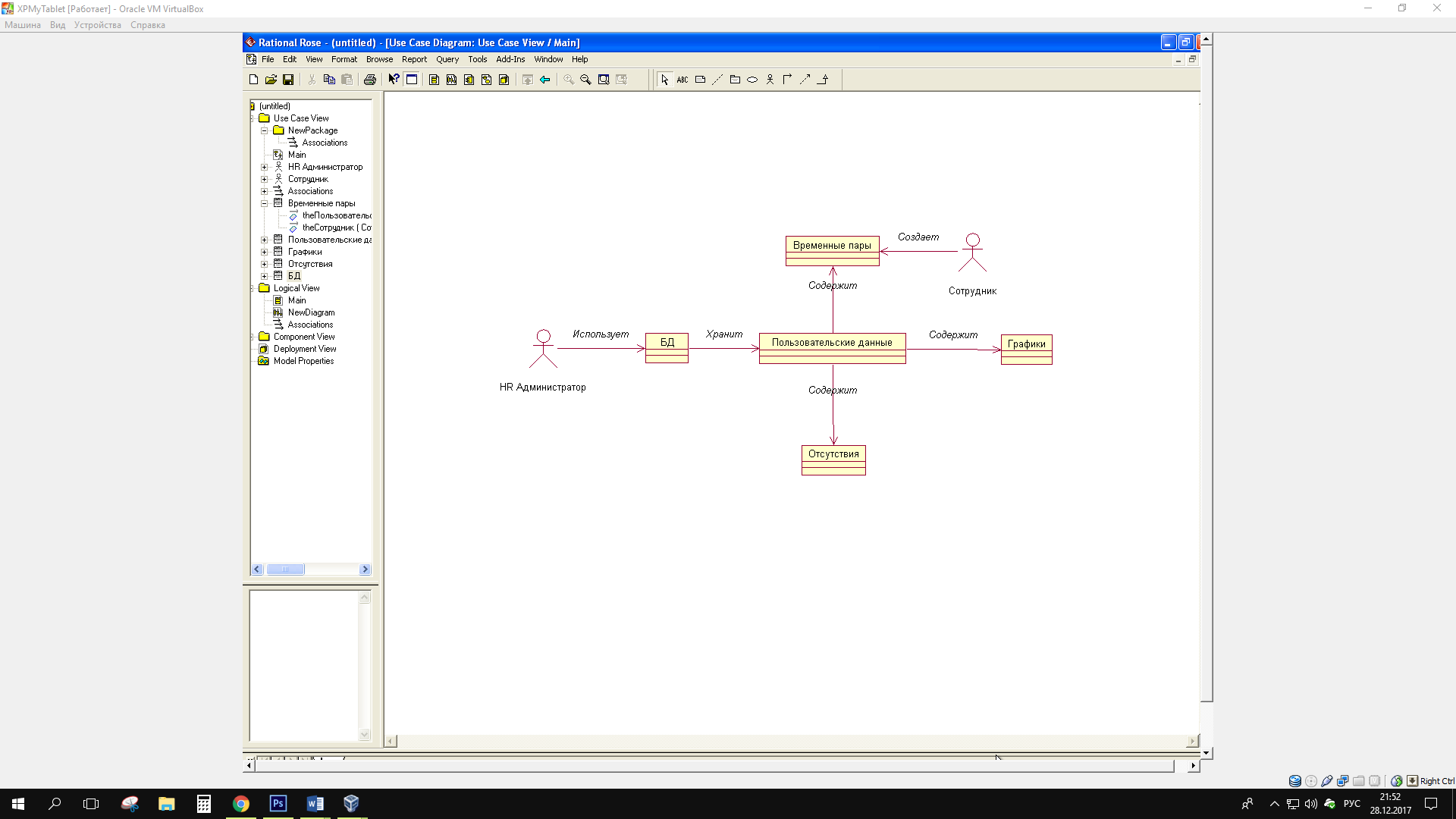


Рис. 3. Концептуальная модель предметной области

* 1. Построение диаграмм последовательностей системы

Диаграмма последовательности (англ. sequencediagram) — диаграмма, на которой для некоторого набора объектов на единой временной оси показан жизненный цикл какого-либо определённого объекта (создание-деятельность-уничтожение некой сущности) и взаимодействие актёров (действующих лиц) ИС в рамках какого-либо определённого прецедента (отправка запросов и получение ответов). Используется в языке UML [1].

Основными элементами диаграммы последовательности являются обозначения объектов (прямоугольники с названиями объектов), вертикальные «линии жизни» (англ. lifeline), отображающие течение времени, прямоугольники, отражающие деятельность объекта или исполнение им определенной функции (прямоугольники на пунктирной «линии жизни»), и стрелки, показывающие обмен сигналами или сообщениями между объектами.

* + 1. Диаграмма последовательностей системы для варианта использования «Просмотр данных сотрудника»

Диаграмма последовательностей представлена на рис. 4.

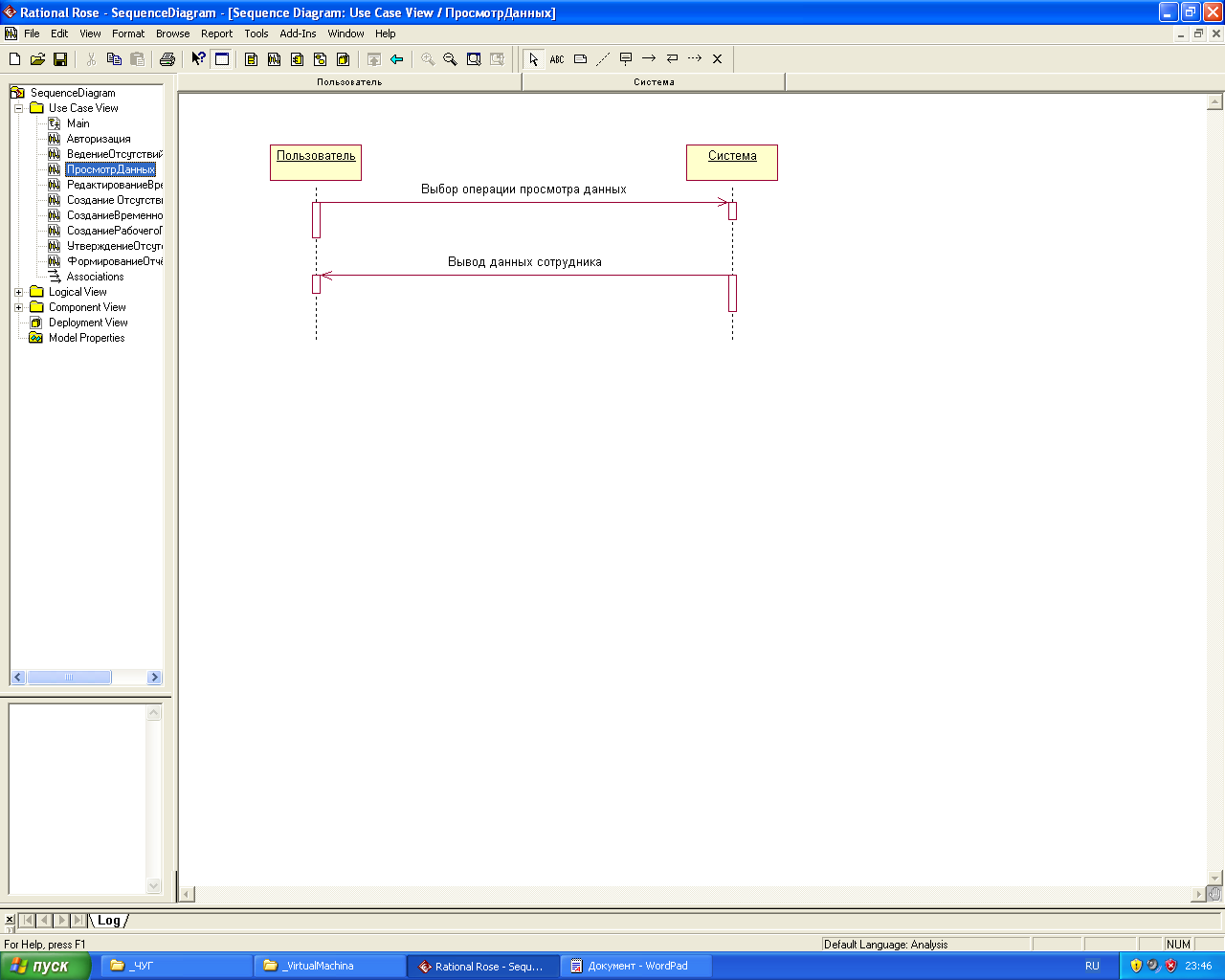


Рис. 4. Диаграмма последовательностей системы для варианта использования Просмотр данных сотрудника

Описание операции «Выбор операции просмотра данных» представлено в табл.10.

Таблица 10

Описание операции «Выбор операции просмотра данных»

|  |  |
| --- | --- |
| Раздел | Описание |
| Имя | Выбор операции просмотра данных |
| Обязанности | Выбор операции просмотра данных пользователя |
| Тип | Системная |
| Ссылки | Вариант использования Просмотр данных сотрудника |
| Примечания | Добавить возможность отменить действие |
| Исключения | - |
| Вывод | - |
| Предусловия | Выбрана операция просмотра данных |
| Постусловие | Окно с информацией о пользователе |

Описание операции «Вывод данных сотрудника» представлено в табл.11.

Таблица 11

Описание операции «Вывод данных сотрудника»

|  |  |
| --- | --- |
| Раздел | Описание |
| Имя | Вывод данных сотрудника |
| Обязанности | Вывод данных сотрудника |
| Тип | Системная |
| Ссылки | Вариант использования Просмотр данных сотрудника |
| Примечания | - |
| Исключения | - |
| Вывод | Окно с информацией о выбранном сотруднике |
| Предусловия | Выбрана операция просмотра данных |
| Постусловие | Окно с информацией о пользователе |

* + 1. Диаграмма последовательностей системы для варианта использования «Создание временной пары»

Диаграмма последовательностей представлена на рис. 5.

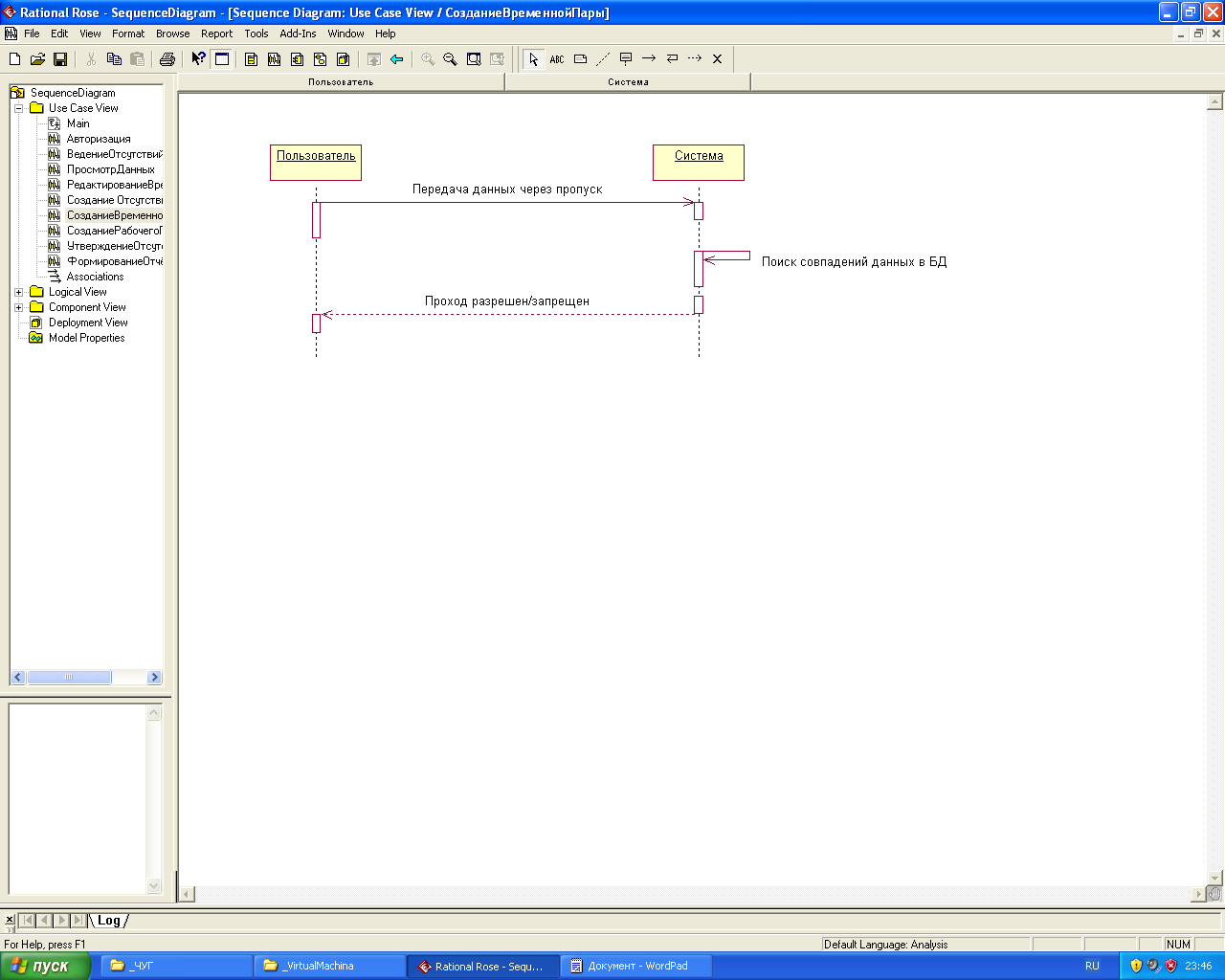


Рис. 5. Диаграмма последовательностей системы для варианта использования Создание временной пары

Описание операции «Передача данных через пропуск» представлено в табл.12.

Таблица 12

Описание операции «Передача данных через пропуск»

|  |  |
| --- | --- |
| Раздел | Описание |
| Имя | Передача данных через пропуск |
| Обязанности | Получить данные, хранящиеся в пропуске |
| Тип | Системная |
| Ссылки | Вариант использования Создание временной пары |
| Примечания | - |
| Исключения | - |
| Вывод | - |
| Предусловия | Сотрудник приложил пропуск |
| Постусловие | Поиск совпадений данных в БД |

Описание операции «Поиск совпадений данных в БД» представлено в табл.13.

Таблица 13

Описание операции «Поиск совпадений данных в БД»

|  |  |
| --- | --- |
| Раздел | Описание |
| 1 | 2 |
| Имя | Поиск совпадений данных в БД |
| Обязанности | Найти совпадение считанных данных с данными в БД |
| Тип | Системная |
| Ссылки | Вариант использования Создание временной пары |
| Примечания | - |
| Исключения | - |

Таблица 13. Продолжение

|  |  |
| --- | --- |
| 1 | 2 |
| Вывод | - |
| Предусловия | Сотрудник приложил пропуск |
| Постусловие | Проход разрешен/запрещен |

Описание операции «Проход разрешен/запрещен» представлено в табл.14.

Таблица 14

Описание операции «Проход разрешен/запрещен»

|  |  |
| --- | --- |
| Раздел | Описание |
| Имя | Проход разрешен/запрещен |
| Обязанности | Разрешить пропуск в кабинет или нет |
| Тип | Системная |
| Ссылки | Вариант использования Создание временной пары |
| Примечания | - |
| Исключения | - |
| Вывод | Загорается лампочка (зеленая – проход разрешен, красная – запрещен). Данные о приходе/уходе записываются в БД |
| Предусловия | Сотрудник приложил пропуск |
| Постусловие | Проход разрешен/запрещен |

* + 1. Диаграмма последовательностей системы для варианта использования «Создание отсутствия»

Диаграмма последовательностей представлена на рис. 5.

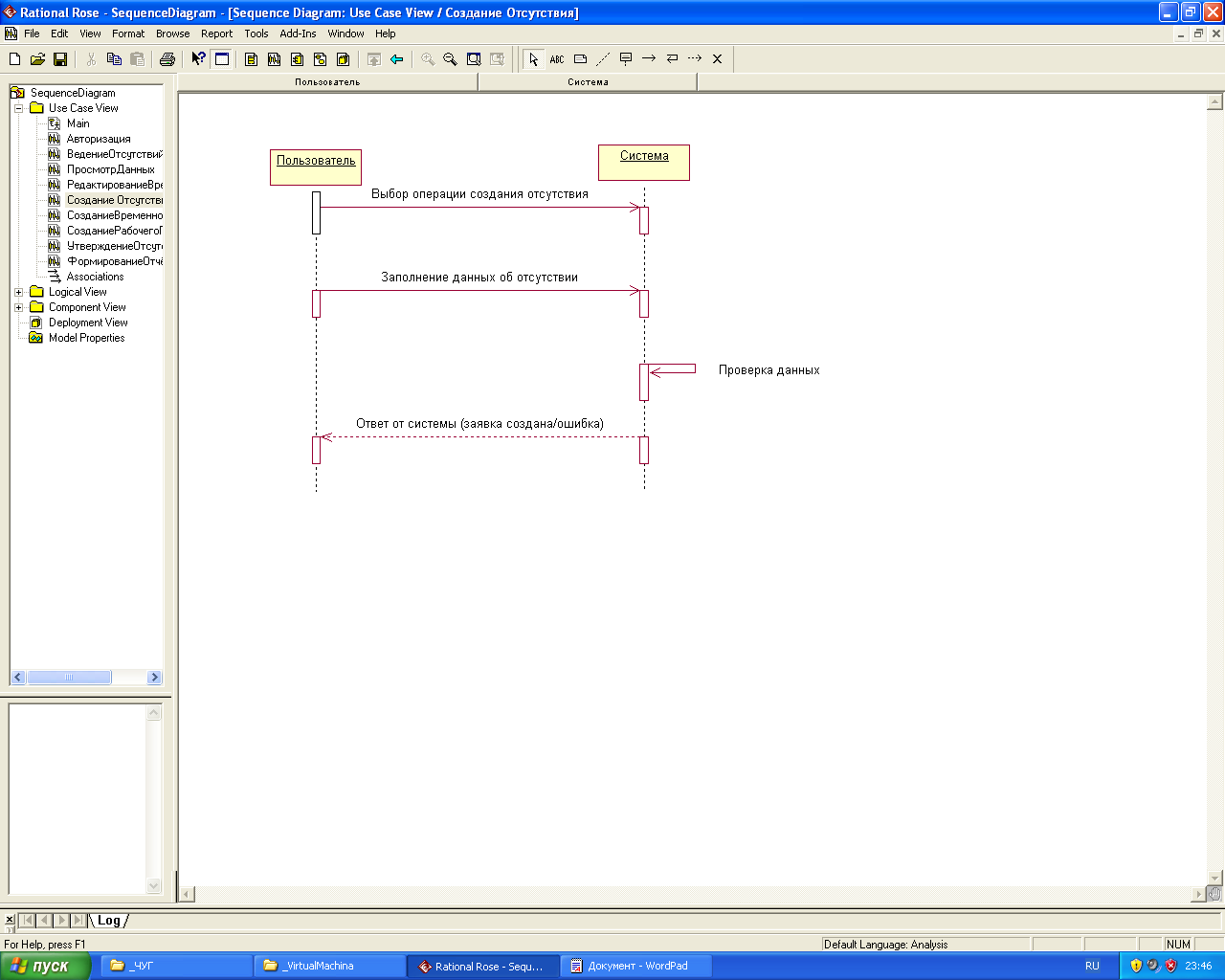


Рис. 6. Диаграмма последовательностей системы для варианта использования Создание отсутствия

Описание операции «Выбор операции создания отсутствия» представлено в табл.15.

Таблица 15

Описание операции «Выбор операции создания отсутствия»

|  |  |
| --- | --- |
| Раздел | Описание |
| 1 | 2 |
| Имя | Выбор операции создания отсутствия |
| Обязанности | Открыть окно для заполнения данных |
| Тип | Системная |
| Ссылки | Вариант использования Создание отсутствия |
| Примечания | - |
| Исключения | - |

Таблица 15. Продолжение

|  |  |
| --- | --- |
| 1 | 2 |
| Вывод | Окно для заполнения данных |
| Предусловия | - |
| Постусловие | Заполнение данных об отсутствии |

Описание операции «Заполнение данных об отсутствии» представлено в табл.16.

Таблица 16

Описание операции «Заполнение данных об отсутствии»

|  |  |
| --- | --- |
| Раздел | Описание |
| Имя | Заполнение данных об отсутствии |
| Обязанности | Дать возможность ввести данные пользователю.  Периодически сохранять введенные данные в локальный файл для возможности восстановить данные |
| Тип | Системная |
| Ссылки | Вариант использования Создание отсутствия |
| Примечания | - |
| Исключения | - |
| Вывод | - |
| Предусловия | Выбор операции создания отсутствия |
| Постусловие | Проверка данных |

Описание операции «Проверка данных» представлено в табл.17.

Таблица 17

Описание операции «Проверка данных»

|  |  |
| --- | --- |
| Раздел | Описание |
| Имя | Проверка данных |
| Обязанности | Проверить введенные данные |
| Тип | Системная |
| Ссылки | Вариант использования Создание отсутствия |
| Примечания | - |
| Исключения | Выделять поля, в которых была допущена ошибка |
| Вывод | Окно с надписью «заявка создана/ошибка» |
| Предусловия | Введены данные |
| Постусловие | Заявка создана/ошибка |

Описание операции «Ответ от системы» представлено в табл.18.

Таблица 18

Описание операции «Ответ от системы (заявка создана/ошибка)»

|  |  |  |
| --- | --- | --- |
| Раздел | Описание | |
| 1 | 2 | |
| Имя | Ответ от системы | |
| Обязанности | | Сообщить сотруднику о том, что заявка создана, либо – ошибка, с указанием места возникновения исключительной ситуации |
| Тип | | Системная |
| Ссылки | | Вариант использования Создание отсутствия |
| Примечания | | - |
| Исключения | | Сообщить сотруднику, если система не может подключиться к БД |

Таблица 18. Продолжение

|  |  |
| --- | --- |
| 1 | 2 |
| Вывод | Окно с надписью «заявка создана/ошибка». Данные записываются в БД |
| Предусловия | - |
| Постусловие | Заявка создана/ошибка |

* + 1. Диаграмма последовательностей системы для варианта использования «Работа с данными сотрудников»

Диаграмма последовательностей представлена на рис. 7.

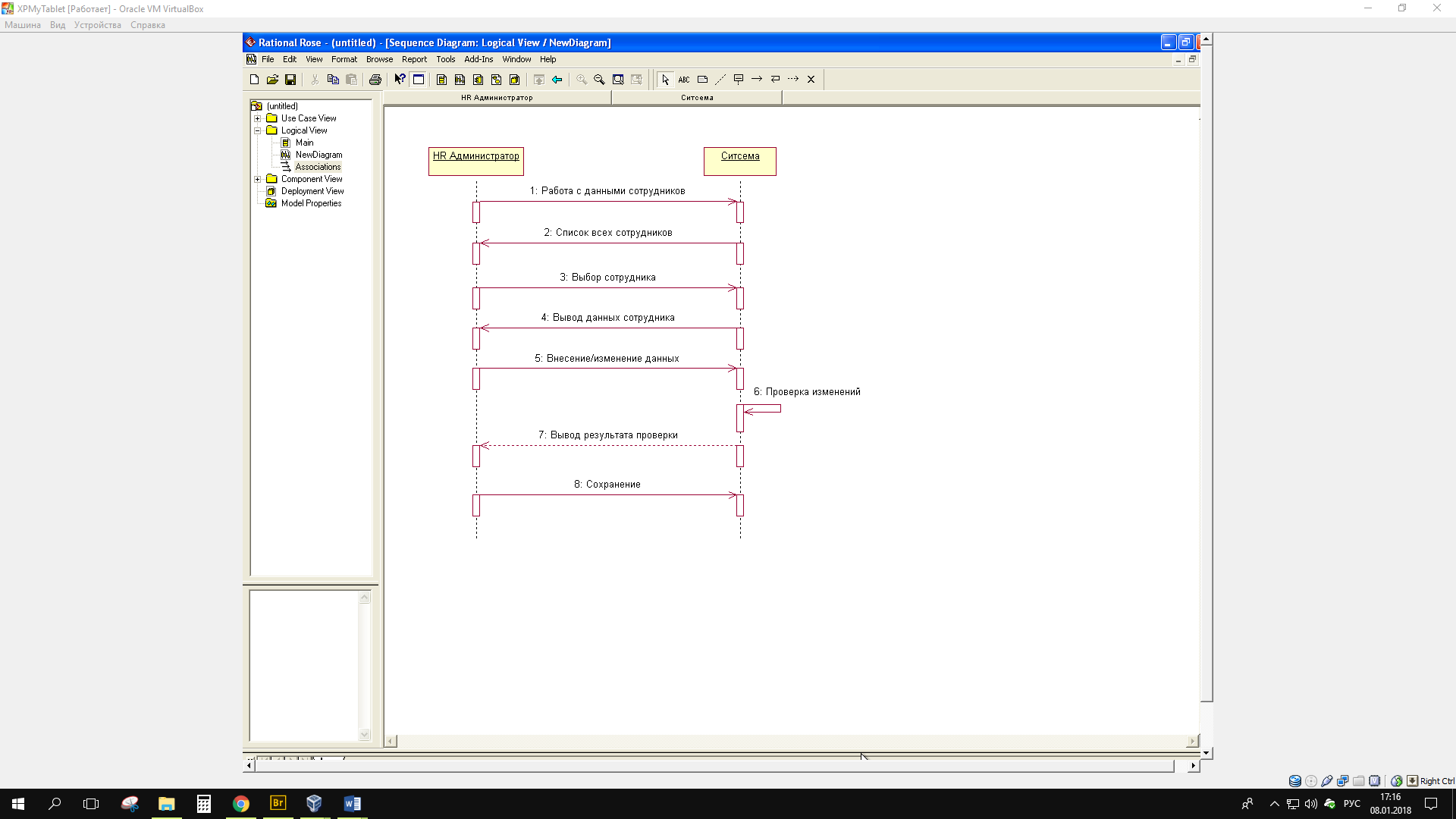


Рис. 7. Диаграмма последовательностей системы для варианта использования Работа с данными сотрудников

Описание операции «Работа с данными сотрудников» представлено в табл.19.

Таблица 19

Описание операции «Работа с данными сотрудников»

|  |  |
| --- | --- |
| Раздел | Описание |
| Имя | Работа с данными сотрудников |
| Обязанности | Предоставить возможность просматривать, корректировать, удалять данные сотрудников в системе |
| Тип | Системная |
| Ссылки | Работа с данными сотрудников |
| Примечания | - |
| Исключения | - |
| Вывод | - |
| Предусловия | - |
| Постусловие | Список всех сотрудников |

Описание операции «Список всех сотрудников» представлено в табл.20.

Таблица 20

Описание операции «Список всех сотрудников»

|  |  |
| --- | --- |
| Раздел | Описание |
| Имя | Список всех сотрудников |
| Обязанности | Предоставить список сотрудников |
| Тип | Системная |
| Ссылки | Работа с данными сотрудников |
| Примечания | - |
| Исключения | - |

Таблица 20. Продолжение

|  |  |
| --- | --- |
| Раздел | Описание |
| Вывод | Список сотрудников |
| Предусловия | Выбран процесс работы с данными сотрудников |
| Постусловие | Выбор сотрудников |

Описание операции «Выбор сотрудников» представлено в табл.21.

Таблица 21

Описание операции «Выбор сотрудников»

|  |  |
| --- | --- |
| Раздел | Описание |
| Имя | Выбор сотрудников |
| Обязанности | Выбрать нужного сотрудника |
| Тип | Системная |
| Ссылки | Работа с данными сотрудников |
| Примечания | - |
| Исключения | - |
| Вывод | Вывод информации о сотруднике |
| Предусловия | Список всех сотрудников |
| Постусловие | Вывод данных сотрудника |

Описание операции «Вывод данных сотрудника» представлено в табл.22.

Таблица 22

Описание операции «Вывод данных сотрудника»

|  |  |
| --- | --- |
| Раздел | Описание |
| Имя | Вывод данных сотрудника |
| Обязанности | Вывести данные сотрудников |
| Тип | Системная |
| Ссылки | Работа с данными сотрудников |
| Примечания | - |
| Исключения | - |
| Вывод | Вывод информации о сотруднике |
| Предусловия | Выбор сотрудников |
| Постусловие | Внесение/изменение данных |

Описание операции «Внесение/изменение данных» представлено в табл.23.

Таблица 23

Описание операции «Внесение/изменение данных»

|  |  |
| --- | --- |
| Раздел | Описание |
| 1 | 2 |
| Имя | Внесение/изменение данных |
| Обязанности | Предоставить возможность внести/изменить данные |
| Тип | Системная |
| Ссылки | Работа с данными сотрудников |
| Примечания | - |
| Исключения | - |
| Вывод | - |

Таблица 23. Продолжение

|  |  |
| --- | --- |
| 1 | 2 |
| Предусловия | Вывод данных сотрудника |
| Постусловие | Проверка изменения |

Описание операции «Проверка изменения» представлено в табл.24.

Таблица 24

Описание операции «Проверка изменения»

|  |  |
| --- | --- |
| Раздел | Описание |
| Имя | Проверка изменения |
| Обязанности | Проверить внесенные данные |
| Тип | Системная |
| Ссылки | Работа с данными сотрудников |
| Примечания | - |
| Исключения | - |
| Вывод | - |
| Предусловия | Внесение/изменение данных |
| Постусловие | Вывод результата проверки |

Описание операции «Вывод результата проверки» представлено в табл.25.

Таблица 25

Описание операции «Вывод результата проверки»

|  |  |
| --- | --- |
| Раздел | Описание |
| 1 | 2 |
| Имя | Ответ от системы |
| Обязанности | Вывести сообщение о результате проверки введенных данных |

Таблица 25. Продолжение

|  |  |
| --- | --- |
| 1 | 2 |
| Тип | Системная |
| Ссылки | Работа с данными сотрудников |
| Примечания | - |
| Исключения | - |
| Вывод | Вывод информации о сотруднике |
| Предусловия | Проверка изменения |
| Постусловие | Сохранение |

Описание операции «Сохранение» представлено в табл.26.

Таблица 26

Описание операции «Сохранение»

|  |  |
| --- | --- |
| Раздел | Описание |
| Имя | Сохранение |
| Обязанности | Сохранить внесенные данные в БД |
| Тип | Системная |
| Ссылки | Вариант использования Работа с данными сотрудников |
| Примечания | - |
| Исключения | - |
| Вывод | - |
| Предусловия | Ответ от системы |
| Постусловие | Сохранение данных |

* + 1. Диаграмма последовательностей системы для варианта использования «Ведение временных пар»

Диаграмма последовательностей представлена на рис. 8.

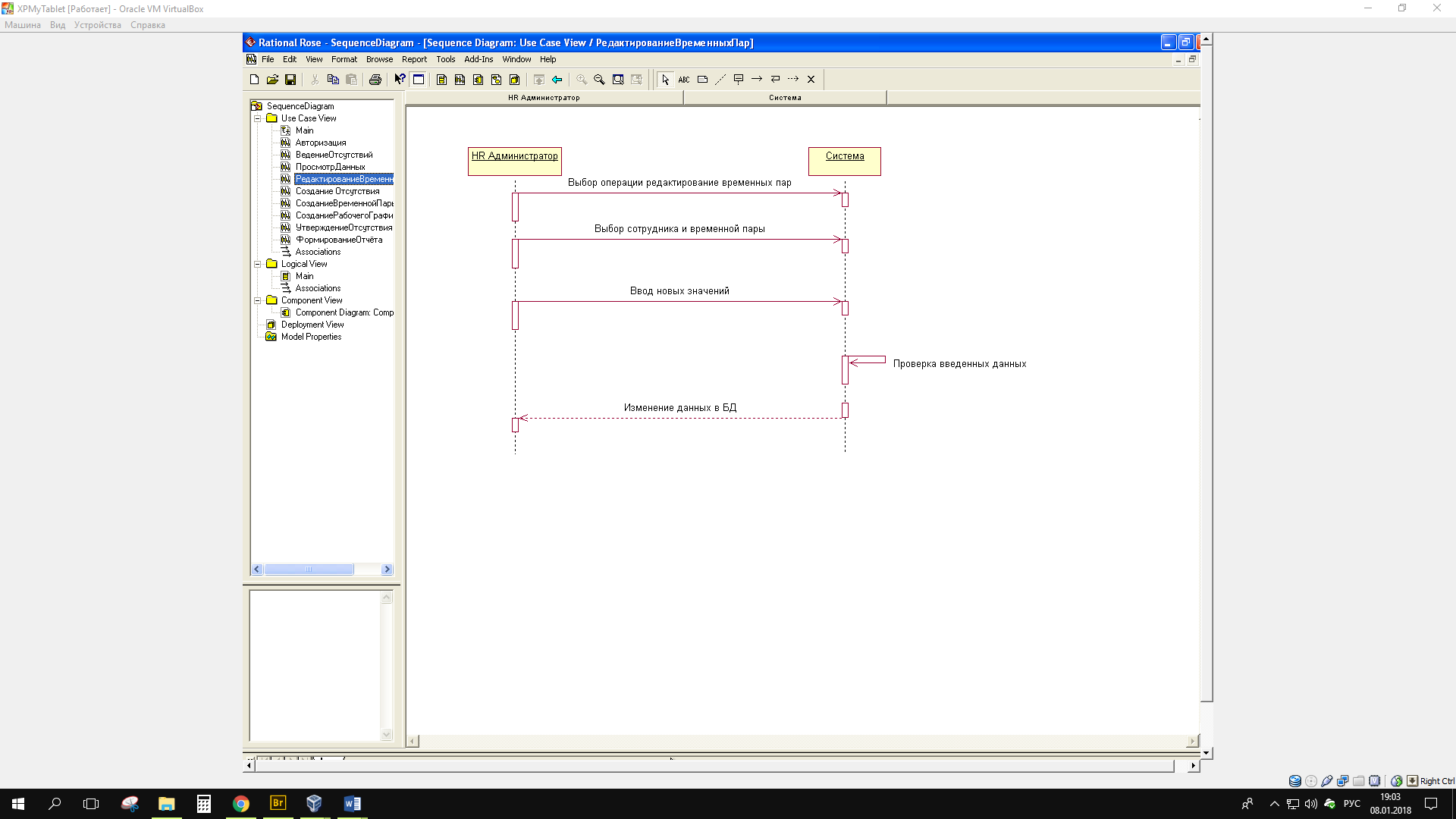


Рис. 8. Диаграмма последовательностей системы для варианта использования Ведение временных пар

Описание операции «Выбор операции редактирования временных пар» представлено в табл.27.

Таблица 27

Описание операции «Выбор операции редактирования временных пар»

|  |  |
| --- | --- |
| Раздел | Описание |
| Имя | Выбор операции редактирования временных пар |
| Обязанности | Открыть окно со списком сотрудников |
| Тип | Системная |
| Ссылки | Вариант использования Ведение временных пар |
| Примечания | - |
| Исключения | - |
| Вывод | Окно со списком сотрудников |
| Предусловия | - |
| Постусловие | Выбор сотрудника и временной пары |

Описание операции «Выбор сотрудника и временной пары» представлено в табл.28.

Таблица 28

Описание операции «Выбор сотрудника и временной пары»

|  |  |
| --- | --- |
| Раздел | Описание |
| Имя | Выбор сотрудника и временной пары |
| Обязанности | Открыть окно для редактирования временной пары |
| Тип | Системная |
| Ссылки | Вариант использования Ведение временных пар |
| Примечания | Реализовать возможность выбора конкретного периода |
| Исключения | - |
| Вывод | Окно для редактирования временной пары |
| Предусловия | Выбор операции редактирования временных пар |
| Постусловие | Ввод новых значений |

Описание операции «Ввод новых значений» представлено в табл.29.

Таблица 29

Описание операции «Ввод новых значений»

|  |  |
| --- | --- |
| Раздел | Описание |
| 1 | 2 |
| Имя | Ввод новых значений |
| Обязанности | Дать возможность внести новые данные |
| Тип | Системная |
| Ссылки | Вариант использования Ведение временных пар |
| Примечания | - |
| Исключения | - |
| Вывод | - |

Таблица 29. Продолжение

|  |  |
| --- | --- |
| 1 | 2 |
| Предусловия | Выбор сотрудника и временной пары |
| Постусловие | Проверка введенных данных |

Описание операции «Проверка введенных данных» представлено в табл.30.

Таблица 30

Описание операции «Проверка введенных данных»

|  |  |
| --- | --- |
| Раздел | Описание |
| Имя | Проверка введенных данных |
| Обязанности | Проверить введенные данные |
| Тип | Системная |
| Ссылки | Вариант использования Ведение временных пар |
| Примечания | - |
| Исключения | Выделять поля, в которых была допущена ошибка |
| Вывод | Сообщение, что данные внесены, либо ошибка |
| Предусловия | Ввод новых значений |
| Постусловие | Изменение данных в БД |

Описание операции «Изменение данных в БД» представлено в табл.31.

Таблица 31

Описание операции «Изменение данных в БД»

|  |  |
| --- | --- |
| Раздел | Описание |
| Имя | Изменение данных в БД |
| Обязанности | Записать новые данные в БД |
| Тип | Системная |
| Ссылки | Вариант использования Ведение временных пар |
| Примечания | - |
| Исключения | Сообщить сотруднику, если система не может подключиться к БД |
| Вывод | Вывод информации о сотруднике |
| Предусловия | - |
| Постусловие | Окно с информацией и пользователе |

* + 1. Диаграмма последовательностей системы для варианта использования «Ведение отсутствий»

Диаграмма последовательностей представлена на рис. 9.

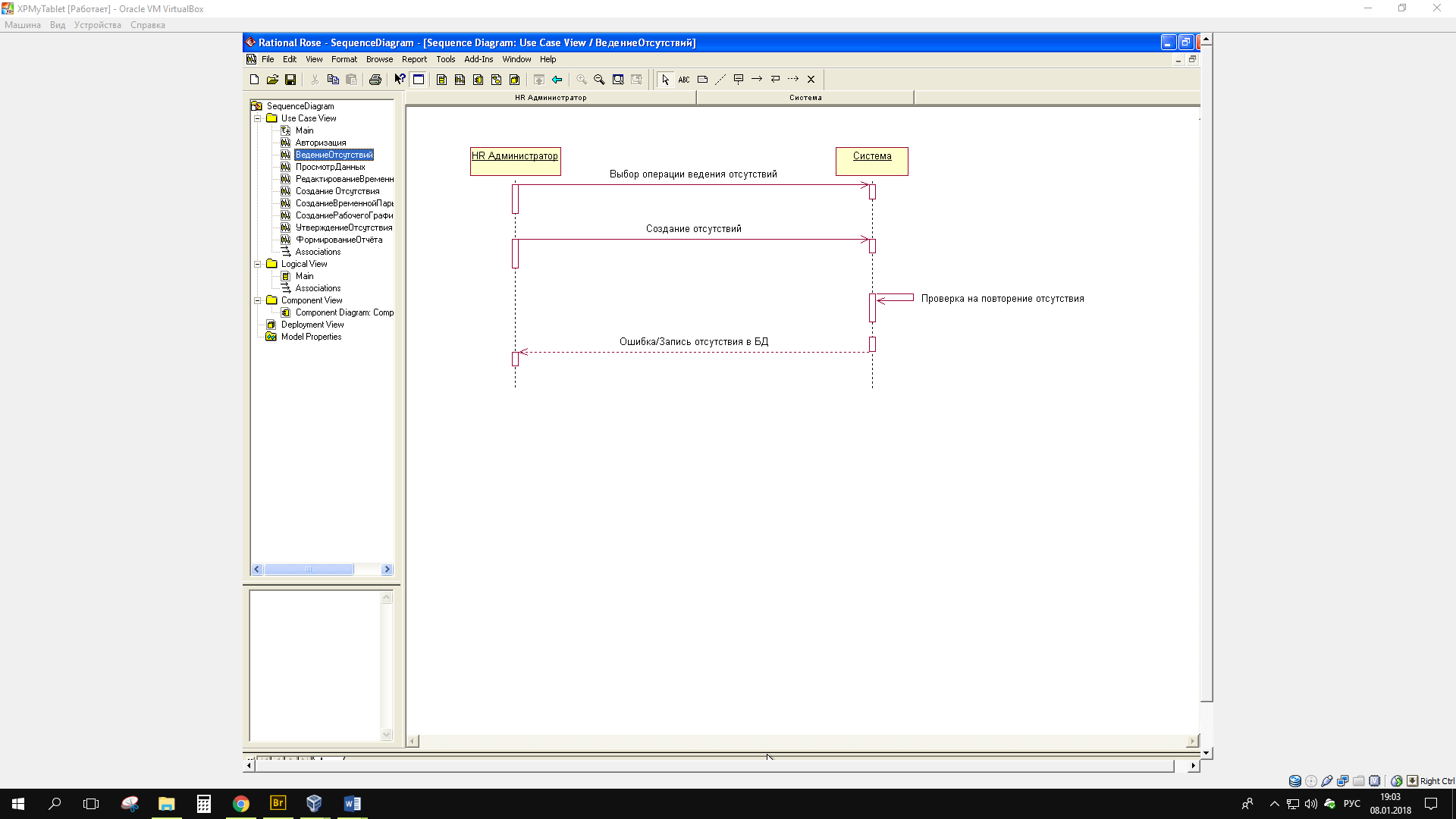


Рис. 9. Диаграмма последовательностей системы для варианта

использования «Ведение отсутствий»

Описание операции «Выбор операции ведения отсутствий» представлено в табл.31.

Таблица 32

Описание операции «Выбор операции ведения отсутствий»

|  |  |
| --- | --- |
| Раздел | Описание |
| Имя | Выбор операции ведения отсутствий |
| Обязанности | Открыть окно для ведения отсутствий |
| Тип | Системная |
| Ссылки | Вариант использования Ведение отсутствий |
| Примечания | - |
| Исключения | - |
| Вывод | Вывод информации о сотруднике |
| Предусловия | - |
| Постусловие | Создание отсутствий |

Описание операции «Создание отсутствий» представлено в табл. 33.

Таблица 33

Описание операции «Создание отсутствий»

|  |  |
| --- | --- |
| Раздел | Описание |
| 1 | 2 |
| Имя | Создание отсутствий |
| Обязанности | Дать возможность внести данные |
| Тип | Системная |
| Ссылки | Вариант использования Ведение отсутствий |
| Примечания | - |

Таблица 33. Продолжение

|  |  |
| --- | --- |
| Исключения | - |
| Вывод | - |
| Предусловия | Нажата кнопка Выбор операции ведения отсутствий |
| Постусловие | Проверка на повторение отсутствия |

Описание операции «Проверка на повторение отсутствия» представлено в табл.34.

Таблица 34

Описание операции «Проверка на повторение отсутствия»

|  |  |
| --- | --- |
| Раздел | Описание |
| Имя | Проверка на повторение отсутствия |
| Обязанности | Проверить БД на повторение отсутствия |
| Тип | Системная |
| Ссылки | Вариант использования Ведение отсутствий |
| Примечания | - |
| Исключения | 1. Выделять поля, в которых была допущена ошибка 2. Сообщить, что такое отсутствие уже внесено в систему |
| Вывод | Сообщение о том, что заявка создана, либо ошибка |
| Предусловия | Создание отсутствий |
| Постусловие | Ошибка/Запись в БД |

Описание операции «Ошибка/Запись в БД» представлено в табл.35.

Таблица 35

Описание операции «Ошибка/Запись в БД»

|  |  |
| --- | --- |
| Раздел | Описание |
| Имя | Ошибка/Запись в БД |
| Обязанности | Сообщить сотруднику об создании заявки, либо об ошибке |
| Тип | Системная |
| Ссылки | Вариант использования Ведение отсутствий |
| Примечания | - |
| Исключения | - |
| Вывод | Вывод информации о сотруднике |
| Предусловия | Проверка на повторение отсутствия |
| Постусловие | Окно с сообщением |

* + 1. Диаграмма последовательностей системы для варианта использования «Ведение рабочих графиков»

Диаграмма последовательностей представлена на рис. 10.

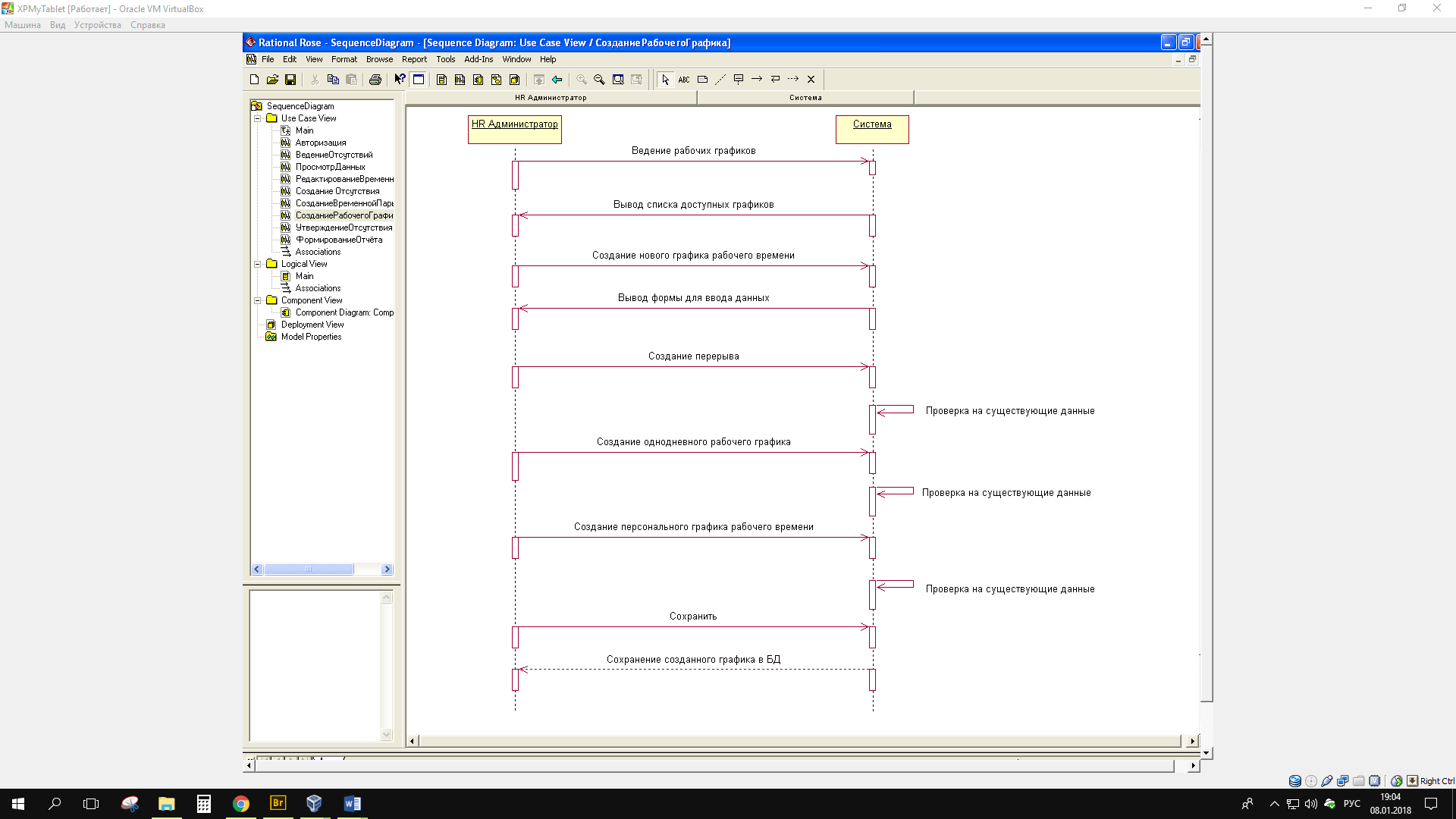


Рис. 10. Диаграмма последовательностей системы для варианта использования «Ведение рабочих графиков»

Описание операции «Выбор операции создания рабочего графика» представлено в табл.36.

Таблица 36

Описание операции «Выбор операции создания рабочего графика»

|  |  |
| --- | --- |
| Раздел | Описание |
| 1 | 2 |
| Имя | Выбор операции создания рабочего графика |
| Обязанности | Открыть окно со списком всех графиков |
| Тип | Системная |
| Ссылки | Вариант использования Создание рабочих графиков |

Таблица 36. Продолжение

|  |  |
| --- | --- |
| 1 | 2 |
| Примечания | - |
| Исключения | - |
| Вывод | Окна для ввода нормативного рабочего дня |
| Предусловия | - |
| Постусловие | Создание нормативного рабочего дня |

Описание операции «Создание нормативного рабочего дня» представлено в табл.37.

Таблица 37

Описание операции «Создание нормативного рабочего дня»

|  |  |
| --- | --- |
| Раздел | Описание |
| Имя | Создание нормативного рабочего дня |
| Обязанности | Дать возможность внести данные о новом графике |
| Тип | Системная |
| Ссылки | Вариант использования Создание рабочих графиков |
| Примечания | Реализовать возможность создания на основании копированного графика |
| Исключения | - |
| Вывод | - |
| Предусловия | Выбор операции создания рабочего графика |
| Постусловие | Проверка на существующие данные |

Описание операции «Проверка на существующие данные» представлено в табл.38.

Таблица 38

Описание операции «Проверка на существующие данные»

|  |  |
| --- | --- |
| Раздел | Описание |
| Имя | Проверка на существующие данные |
| Обязанности | Проверить данные о новом графике с данными в БД |
| Тип | Системная |
| Ссылки | Вариант использования Создание рабочих графиков |
| Примечания | - |
| Исключения | Выдать окно, если есть совпадение |
| Вывод | Вывод ошибки, если такой график уже существует |
| Предусловия | Создание нормативного рабочего дня |
| Постусловие | Создание сокращенного рабочего дня |

Описание операции «Создание сокращенного рабочего дня» представлено в табл.39.

Таблица 39

Описание операции «Создание сокращенного рабочего дня»

|  |  |
| --- | --- |
| Раздел | Описание |
| 1 | 2 |
| Имя | Создание сокращенного рабочего дня |
| Обязанности | Дать возможность ввести данные о новом сокращенном дне в рабочем графике |
| Тип | Системная |
| Ссылки | Вариант использования Создание рабочих графиков |
| Примечания | Реализовать возможность создания на основании копирования |
| Исключения | - |

Таблица 39. Продолжение

|  |  |
| --- | --- |
| 1 | 2 |
| Вывод | - |
| Предусловия | Проверка на существующие данные |
| Постусловие | Проверка на существующие данные |

Описание операции «Проверка на существующие данные» представлено в табл. 40.

Таблица 40

Описание операции «Проверка на существующие данные»

|  |  |
| --- | --- |
| Раздел | Описание |
| Имя | Проверка на существующие данные |
| Обязанности | Проверить данные о новом сокращенном дне с данными в БД |
| Тип | Системная |
| Ссылки | Вариант использования Создание рабочих графиков |
| Примечания | - |
| Исключения | Выдать окно, если есть совпадение |
| Вывод | Вывод ошибки, если такой день уже существует |
| Предусловия | Создание сокращенного рабочего дня |
| Постусловие | Создание перерыва |

Описание операции «Создание перерыва» представлено в табл.41.

Таблица 41

Описание операции «Создание перерыва»

|  |  |
| --- | --- |
| Раздел | Описание |
| Имя | Создание перерыва |
| Обязанности | Передать данных, хранящиеся в пропуске |
| Тип | Системная |
| Ссылки | Вариант использования Создание рабочих графиков |
| Примечания | - |
| Исключения | - |
| Вывод | Вывод информации о сотруднике |
| Предусловия | Проверка на существующие данные |
| Постусловие | Проверка на существующие данные |

Описание операции «Проверка на существующие данные» представлено в табл.42.

Таблица 42

Описание операции «Проверка на существующие данные»

|  |  |
| --- | --- |
| Раздел | Описание |
| 1 | 2 |
| Имя | Проверка на существующие данные |
| Обязанности | Проверить данные о новом перерыве с данными в БД |
| Тип | Системная |
| Ссылки | Вариант использования Создание рабочих графиков |
| Примечания | - |
| Исключения | Выдать окно, если есть совпадение |
| Вывод | Вывод ошибки, если такой перерыв уже существует |

Таблица 42. Продолжение

|  |  |
| --- | --- |
| 1 | 2 |
| Предусловия | Создание перерыва |
| Постусловие | Сохранение созданного графика в БД |

Описание операции «Сохранение созданного графика в БД» представлено в табл.43.

Таблица 43

Описание операции «Сохранение созданного графика в БД»

|  |  |
| --- | --- |
| Раздел | Описание |
| Имя | Сохранение созданного графика в БД |
| Обязанности | Сохранить данные о новом графике |
| Тип | Системная |
| Ссылки | Вариант использования Создание рабочих графиков |
| Примечания | - |
| Исключения | Сообщить сотруднику, если нет связи с ЮД |
| Вывод | Вывод сообщения об изменении статуса заявки или об ошибке |
| Предусловия | Сохранение созданного графика в БД |
| Постусловие | Сообщение о том, что график создан, либо об ошибке |



## Построение диаграмм деятельностей сценариев вариантов использования

На этапе анализа требований и уточнения спецификаций диаграммы деятельностей позволяют конкретизировать основные функции разрабатываемого программного обеспечения. Под деятельностью в данном случае понимают задачу (операцию), которую необходимо выполнить вручную или с помощью средств автоматизации [1]. Каждому варианту использования соответствует своя последовательность задач. В теоретическом плане диаграммы деятельности являются обобщенным представлением алгоритма, реализующего анализируемый вариант использования.

### Построение диаграммы деятельностей для сценария варианта использования «Управление временными данными»

Учитывая описание предметной области в виде контекстной диаграммы классов, анализируем описание сценария варианта использования Управление временными данными. Производим разбиение процесса на отдельные операции. Полученные операции показаны на диаграмме деятельностей (рис.11). Описание деятельностей представлено в табл. 44.

Таблица 44

Описание деятельностей для сценария варианта использования Управление временными данными

|  |  |  |  |
| --- | --- | --- | --- |
| № | Деятельность | Описание | |
| 1 | 2 | 3 | |
| 1 | Утверждение отсутствий | Подпрограмма с возможностью просмотра всех отсутствий, созданных сотрудниками | |
| 2 | Просмотр заявок | Вывод всех заявок на отсутствие | |
| 3 | Утверждение/отклонение заявок на отсутствие | | При просмотре конкретной заявки можно либо утвердить её, либо отклонить |
| 4 | Ведение рабочих графиков | | Ведение графиков рабочего времени |
| 5 | Просмотр графиков | | Просмотр общего списка графиков |
| 6 | Создание перерыва | | Создание перерыва |

Таблица 44. Продолжение

|  |  |  |
| --- | --- | --- |
| 1 | 2 | 3 |
| 7 | Создание ОГРВ | Создание однодневного графика рабочего времени (в нем создаем выходные и праздники) |
| 8 | Создание ПГРВ | Создание персонального графика рабочего времени (он включает в себя перерыв и ОГРВ) |

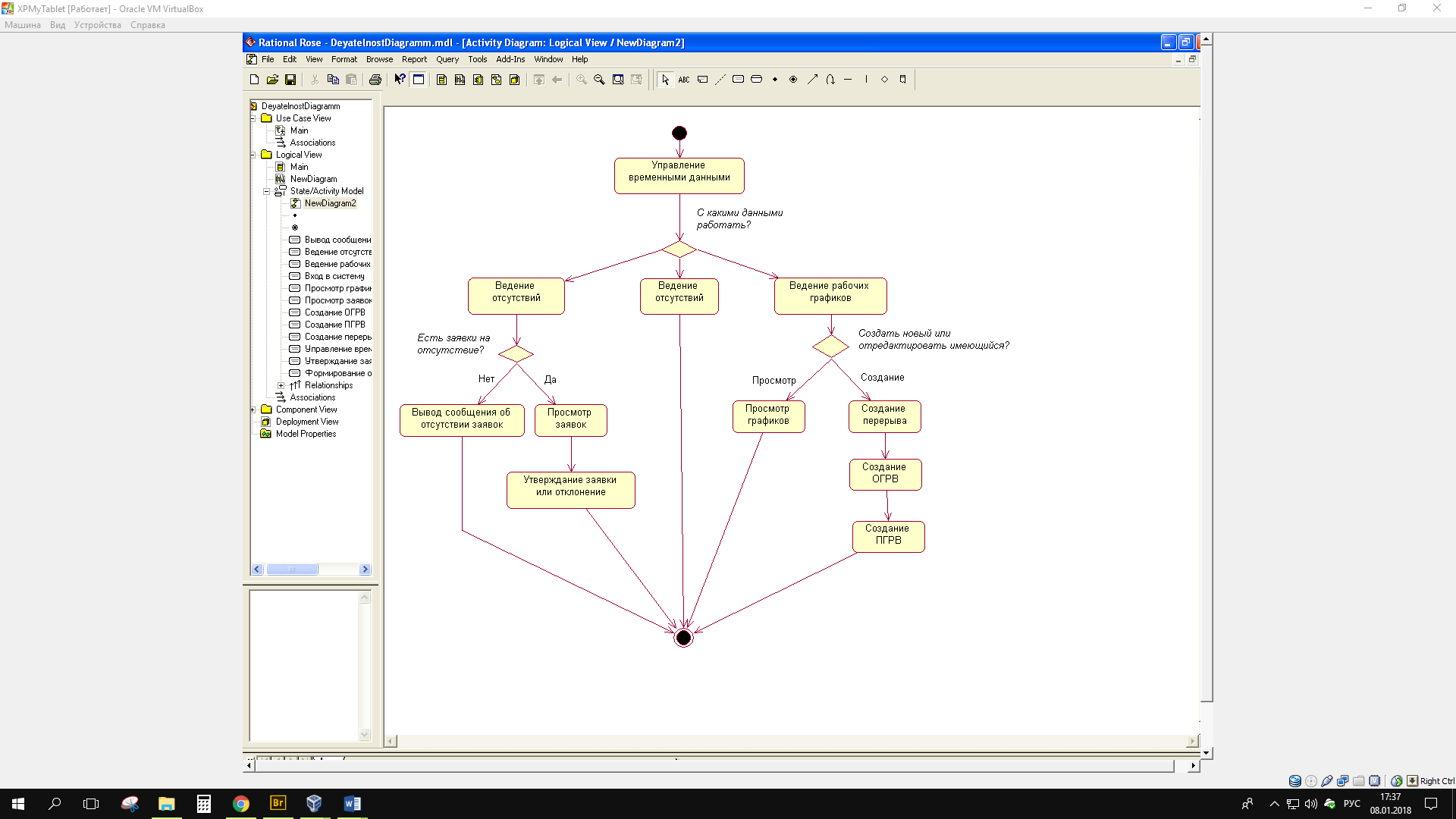


Рис. 11. Диаграмма деятельностей для сценария варианта использования «Управление временными данными»

* 1. Проектирование структур данных и построение диаграмм отношений компонентов данных

Основой информационной системы является база данных (БД). Первым шагом при проектировании БД является анализ предметной области. На основе полученных результатов формируется первый вариант таблицы, к которому затем необходимо применить процесс нормализации. При создании и разработке информационных систем управления предприятиями используются CASE-средства.

CASE-средства - это инструмент, который позволяет автоматизировать процесс разработки информационной системы и программного обеспечения. Для проектирования информационной системы электронного архива документов Череповецкого муниципального района было выбрано CASE-средство **ERwin**.

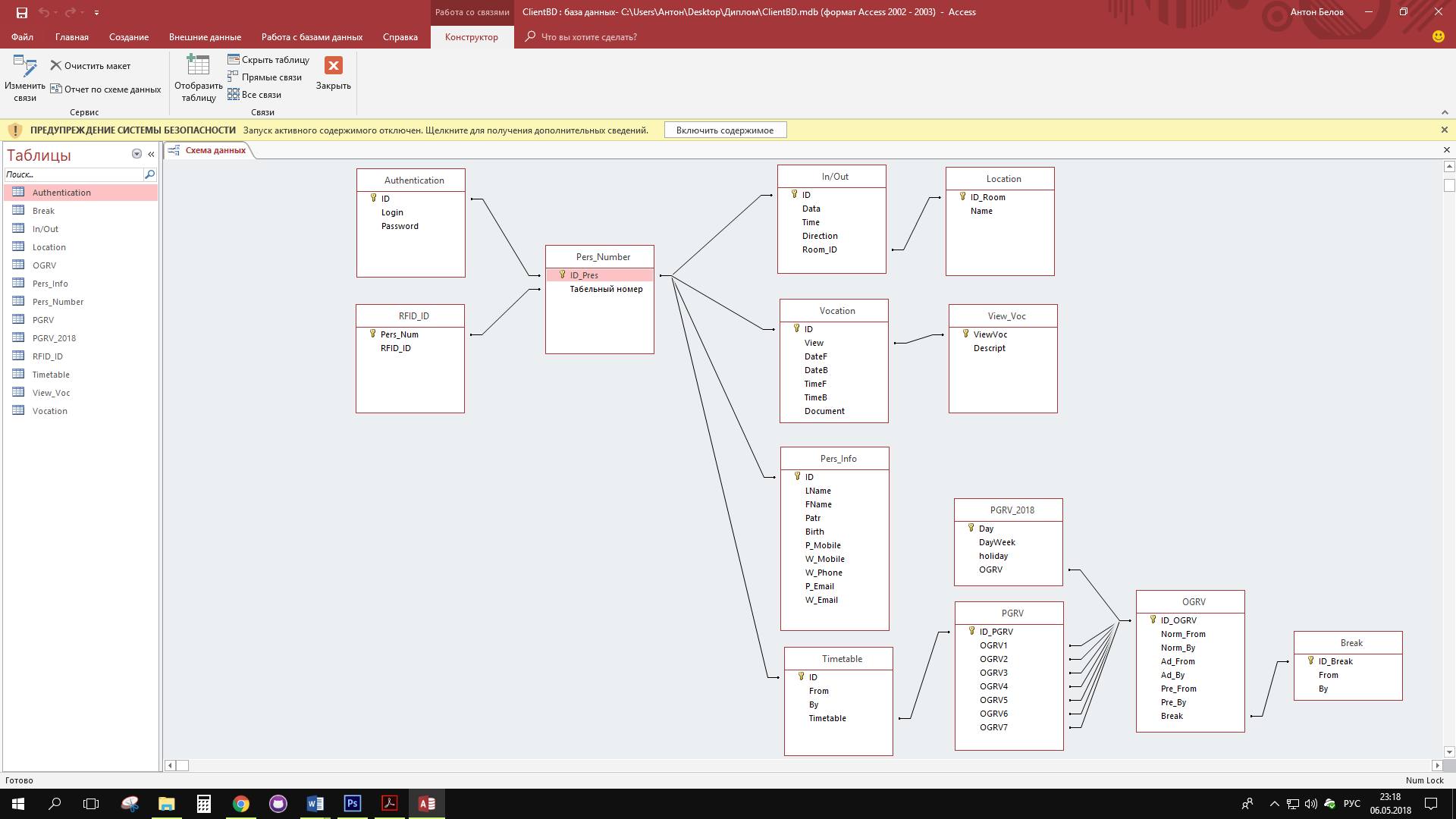
Методологическую основу ERwin составляет технология IDEF1X (моделирование данных для реляционных СУБД). Результатом построения является ER-диаграмма ("сущность-связь"). Графический подход к созданию моделей значительно упрощает процесс разработки.

ERwin имеет два уровня представления модели - логический и физический.

Логический уровень - это абстрактный взгляд на данные, на нем данные представляются так, как выглядят в реальном мире, и могут называться так, как они называются в реальном мире.

Объекты модели, представляемые на логическом уровне, называются сущностями и атрибутами. Логическая модель данных является универсальной и никак не связана с конкретной реализацией СУБД.

Логическая модель системы учёта рабочего времени и местоположения АО «СберТех» представлена на рис.\*.



В логической модели было выделено 13 сущностей:

* Pers\_Number - хранит ID сотрудника и его табельный номер.
* Authentication - хранит ID сотрудника и его данные для входа в систему (Логин и пароль будут хешироваться)
* RFID\_ID - хранит табельный номер и RFID ID пропускной карты сотрудника
* In/Out - хранит ID сотрудника, дату, время и место куда он вошел или вышел.
* Location - Хранит ID комнаты и название.
* Vocation - хранит ID сотрудника, вид отсутствия, дату с/по, премия с/по (нужно, если отсутствие было не полный день, например, обучение)
* View\_Voc - хранит вид отсутствия и название.
* Pers\_Info - хранит ФИО, ДР, Мобильный личный и рабочий, стационарный рабочий, Email личный и рабочий
* Timetable - хранит ID сотрудника, действие графика (дату с /по) и код ПГРВ (Персональный график рабочего времени)
* PGRV - Хранит ID ПГРВ и ОГРВ, которые присваиваются для каждого дня (из расчета 7 дней в неделю).
* OGRV - хранит ID ОГРВ, нормативное время, сокращенное , предпраздничное, Код перерыва.
* Break - Код перерыва и время с/по.
  1. Построение диаграмм переходов состояний

пьывьпжвы

# Проектирование системы

# Построение структуры системы и диаграмм пакетов

Анализ концептуальной модели (рис. 3) и диаграммы вариантов использования (рис.2) позволяют выделить следующие группы классов или пакеты:

* пакет пользователи – классы, реализующие сценарии вариантов использования для Пользователь;
* пакет администрирование включает в себя:
  + пакет ПГРВ (графики) - реализующие сценарий варианта использования Создание рабочих графиков;
  + пакет Управление сотрудника -реализующие сценарий варианта использования Работа с данными сотрудников;
* пакет Управление отчётами;
* пакет БД;
* обработка исключительных ситуаций - классы исключений, реализующие обработку нештатных ситуаций.

Диаграмма пакетов представлена на рис.12. Пакет Обработка исключительных ситуаций объявлен глобальным, так как его элементы могут использовать классы всех пакетов.

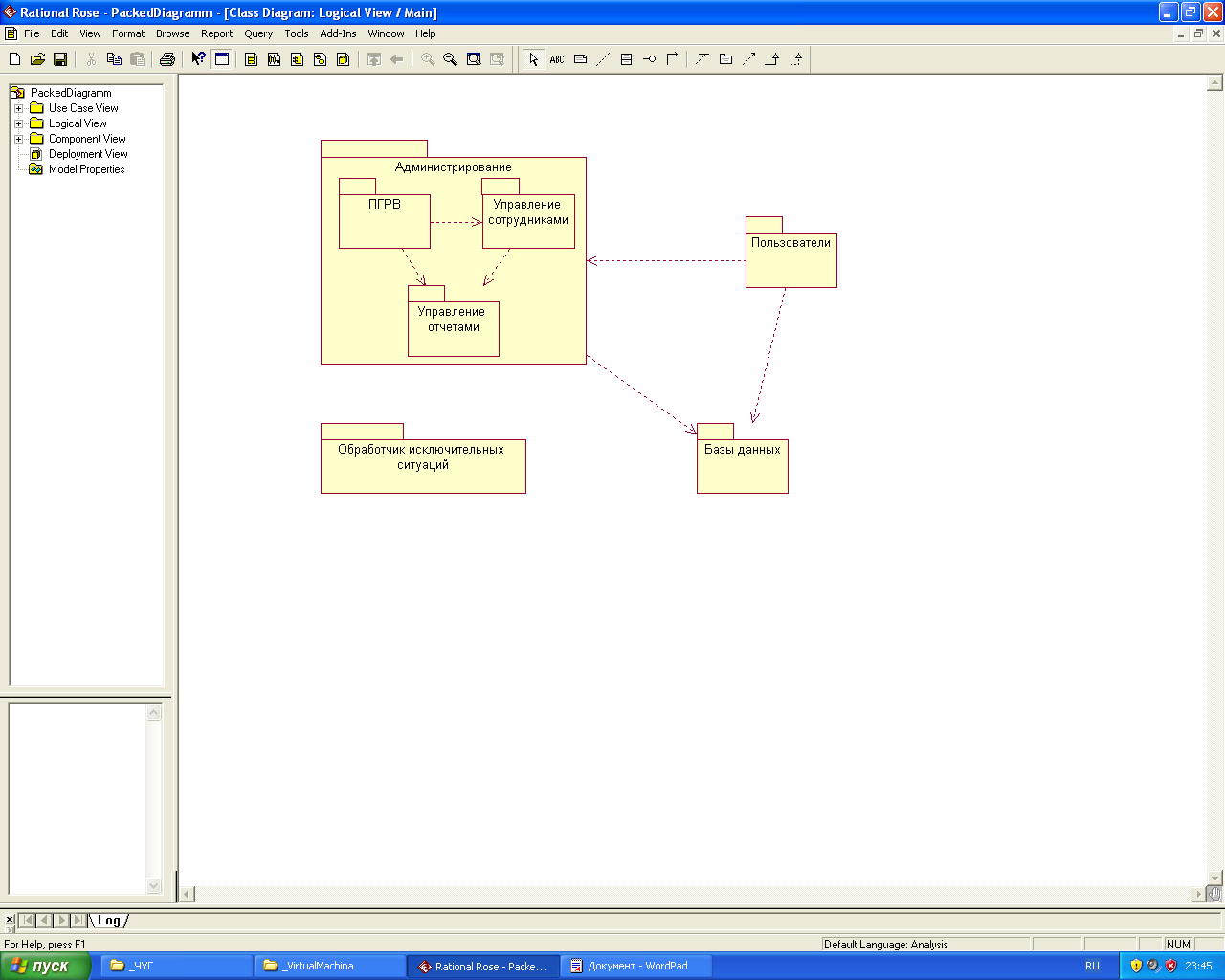


Рис. 12. Диаграмма пакетов

* 1. Проектирование функционального пакета ПГРВ
     1. Построение исходной диаграммы классов

После определения основных пакетов разрабатываемого программного обеспечения переходят к детальному проектированию классов, входящих в каждый пакет. Классы-кандидаты, которые предположительно должны войти в конкретный пакет показывают на диаграмме классов этапа проектирования и уточняют отношения между объектами указанных классов. В таб.45 представлено описание классов рассматриваемого пакета. Исходная диаграмма классов представлена на рис. 13.

Таблица 45

Описание классов пакета ПГРВ

|  |  |
| --- | --- |
| Класс | Описание |
| break | Класс создания перерыва |
| ogrv | Класс создания ОГРВ |
| pgrv | Класс создания ПГРВ |

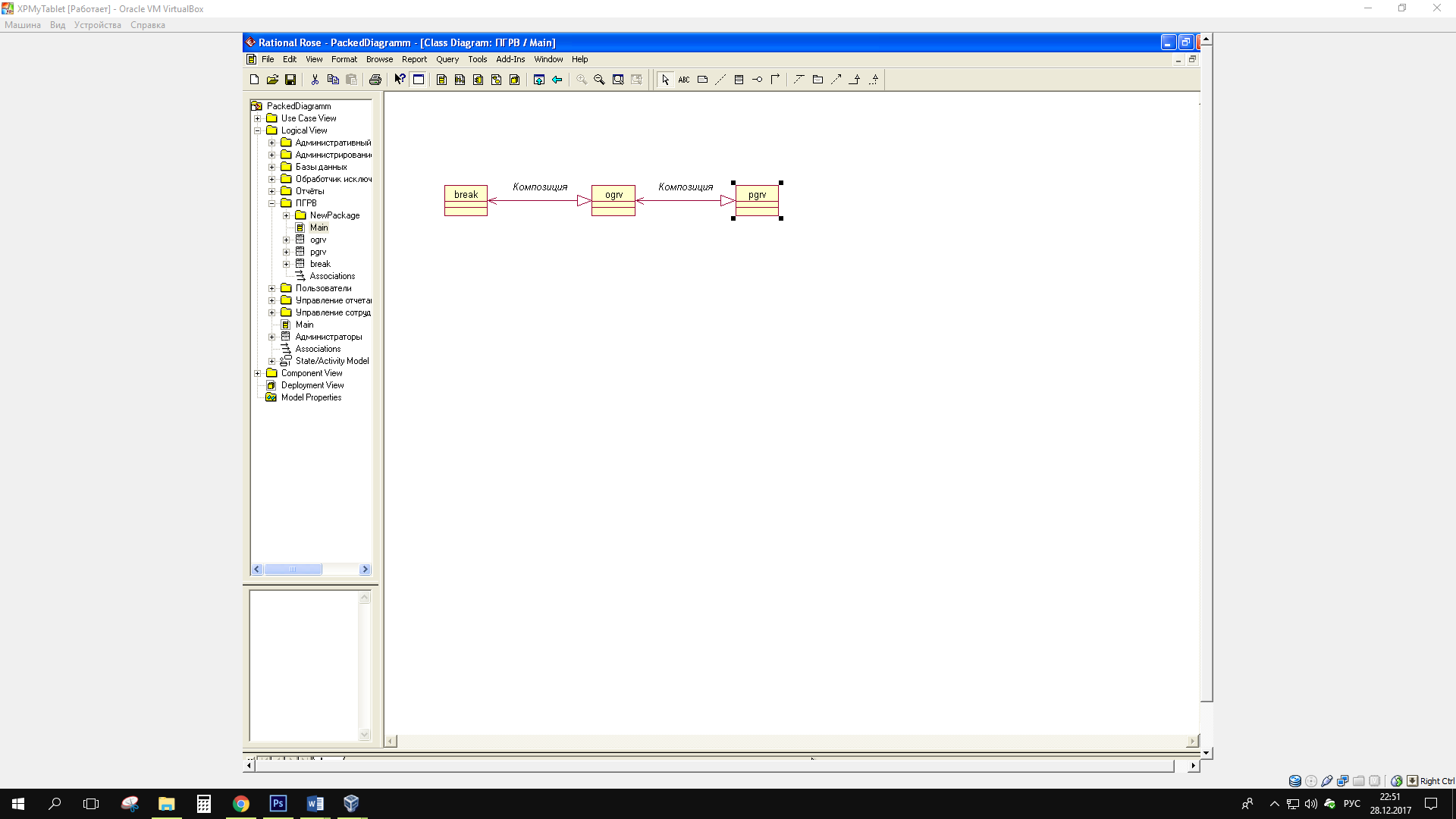


Рис.13. Классы пакета ПГРВ

### Построение диаграмм последовательностей взаимодействия объектов классов пакета ПГРВ

Диаграммы последовательностей этапа проектирования отображают взаимодействие объектов, упорядоченное по времени [1]. В отличие от диаграмм последовательности этапа анализа на ней показывают внутренние объекты, а также последовательность сообщений, которыми обмениваются объекты в процессе реализации фрагмента варианта использования, называемого сценарием.

Во время анализа вариантов использования ПГРВ было выявлены четыре ситуации:

1. нормальный ход событий;
2. повтор/ошибка в перерыве;
3. повтор/ошибка ОГРВ;
4. повтор/ошибка ПГРВ.

На рис. 14 – 17 представлены диаграммы последовательностей взаимодействия объектов классов рассматриваемого пакета. Данные диаграммы отображают взаимодействие классов в четырех ситуациях: нормальный ход событий (рис.14), повтор/ошибка в перерыве (рис.15), повтор/ошибка ОГРВ (рис.16), повтор/ошибка ПГРВ (рис.17).

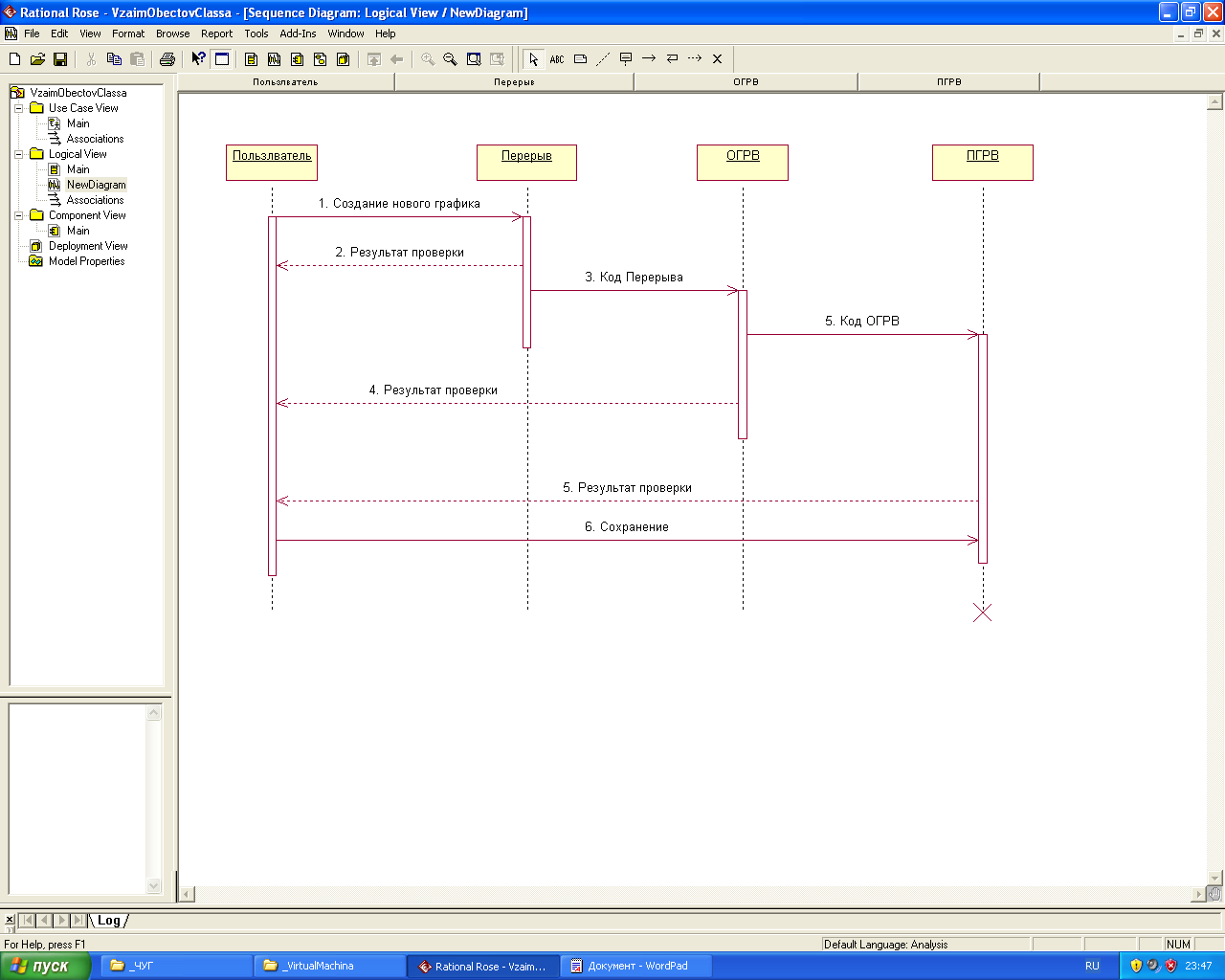


Рис. 14. Нормальный ход событий

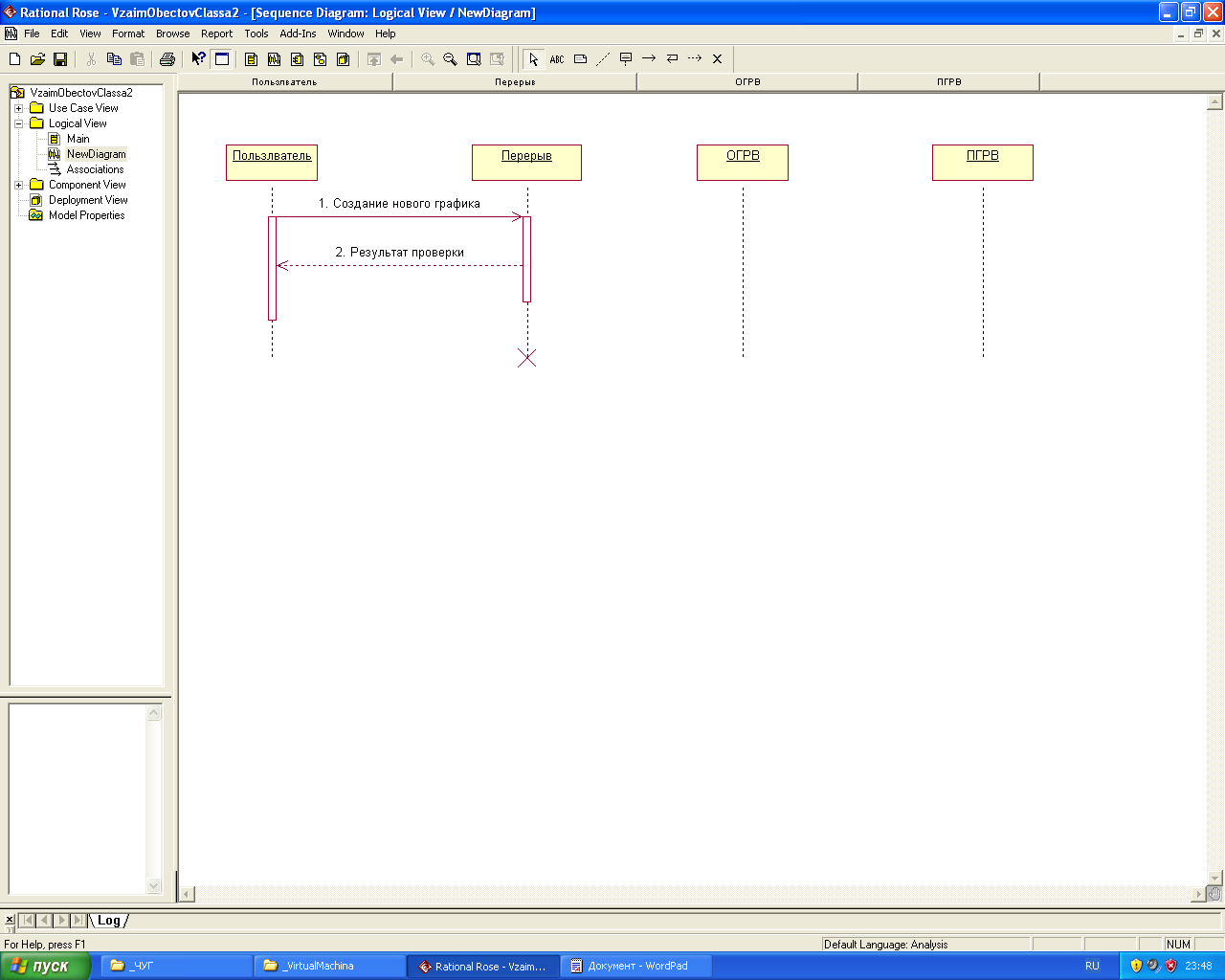


Рис.15. Прерывание процесса системой из-за ошибки в Перерыве

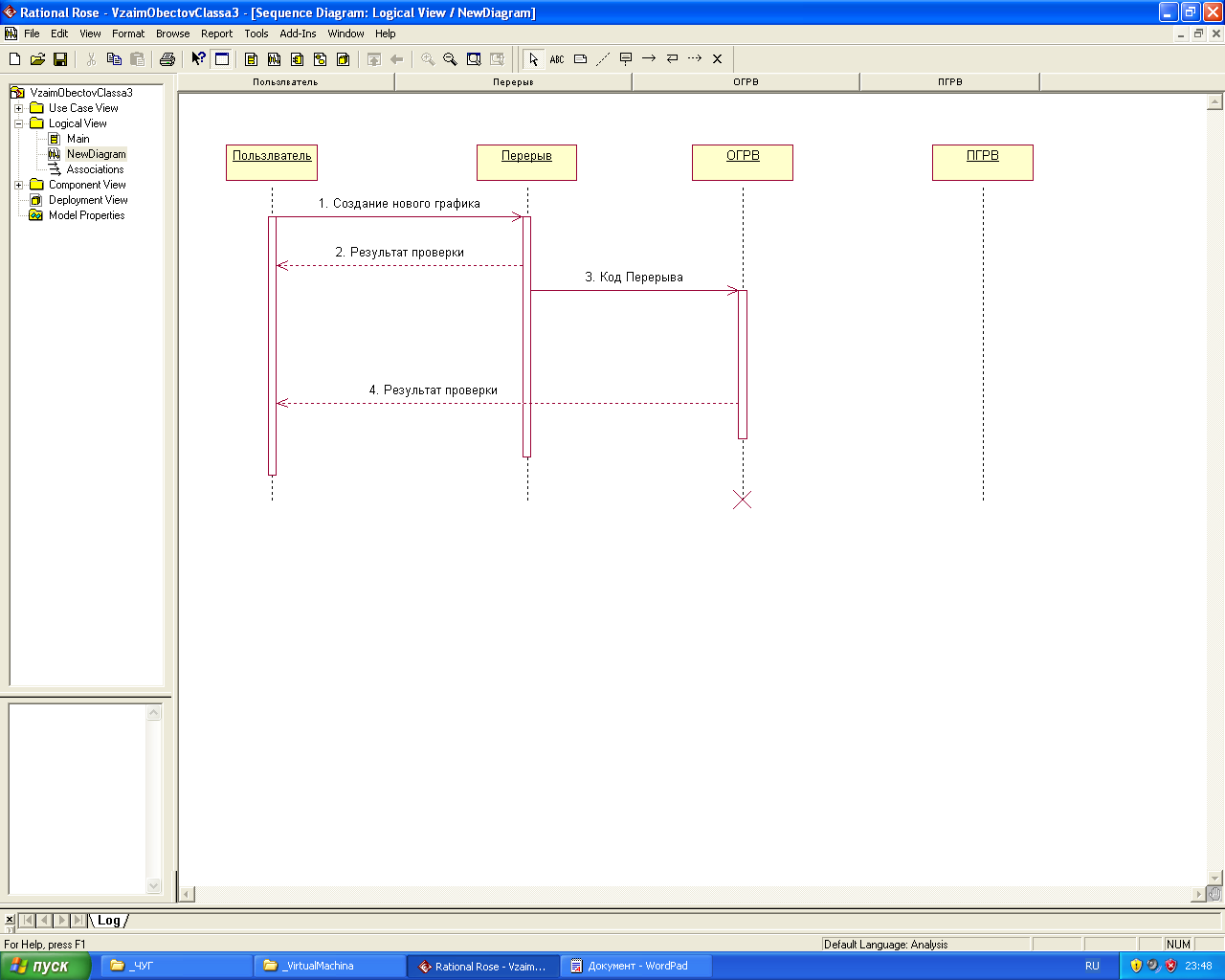


Рис. 16. Прерывание процесса системой из-за ошибки в ОГРВ

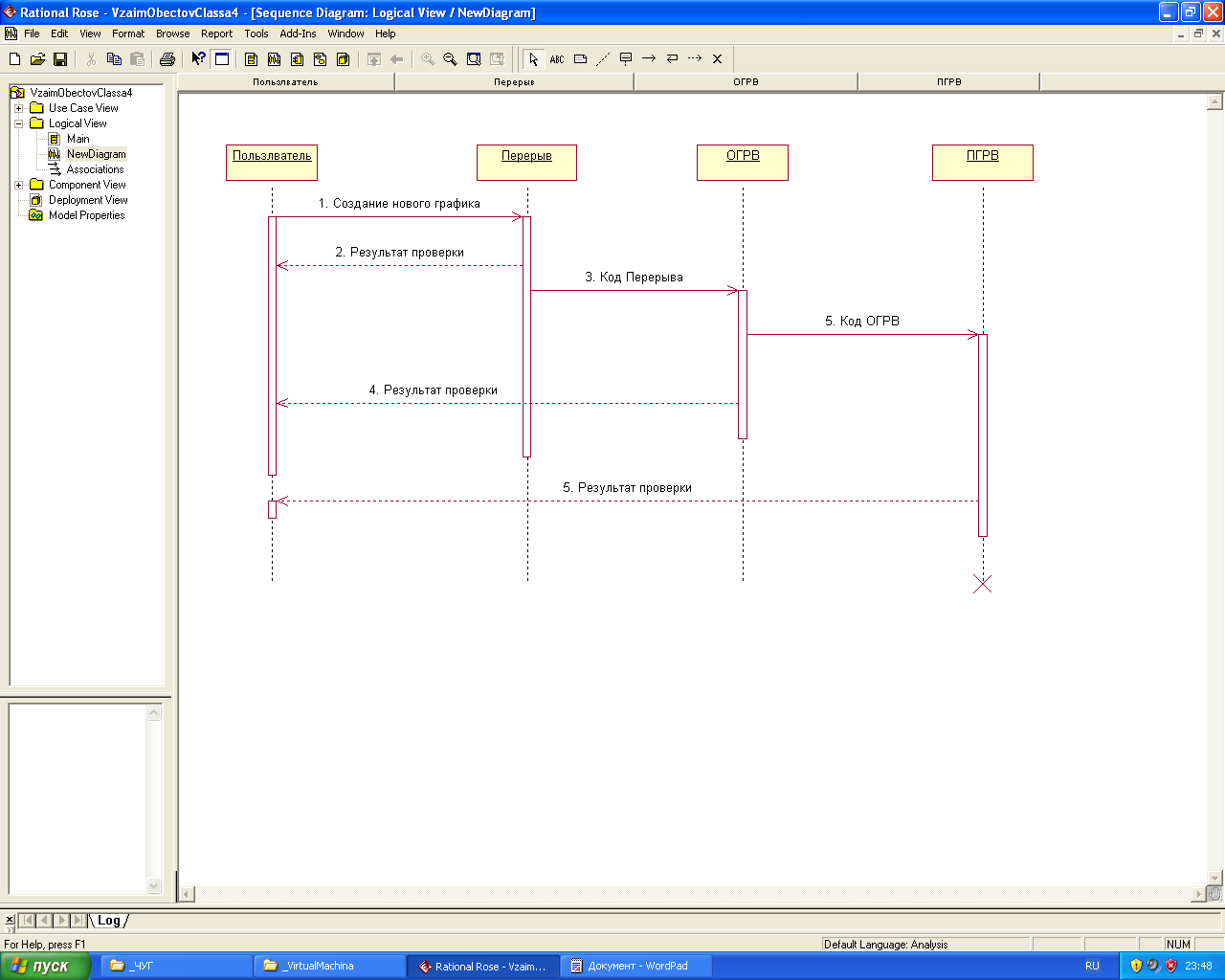


Рис.17. Прерывание процесса системой из-за ошибки в ПГРВ

### Построение диаграммы коопераций пакета ПГРВ

Диаграмма кооперации - это альтернативный способ представления взаимодействия объектов в процессе реализации сценария, который позволяет по-другому взглянуть на ту же информацию [1]. В отличие от диаграмм последовательностей диаграммы кооперации показывают потоки данных между объектами классов, что позволяет уточнить связи между ними.

Диаграмма коопераций пакета ПГРВ представлена на рис. 18.



Рис. 18. Диаграмма коопераций

### Построение уточненной диаграммы классов пакета ПГРВ

Класс ПГРВ (pgrv, персональный график рабочего времени) – класс, который содержит объект класса ОГРВ (ogrv, однодневный график рабочего времени). Класс ОГРВ содержит объект класса Перерыв (break). Таким образом класс ПГРВ зависит от двух других классов – Перерыв и ОГРВ.

Уточненная диаграмма классов пакета ПГРВ представлена на рис.19.

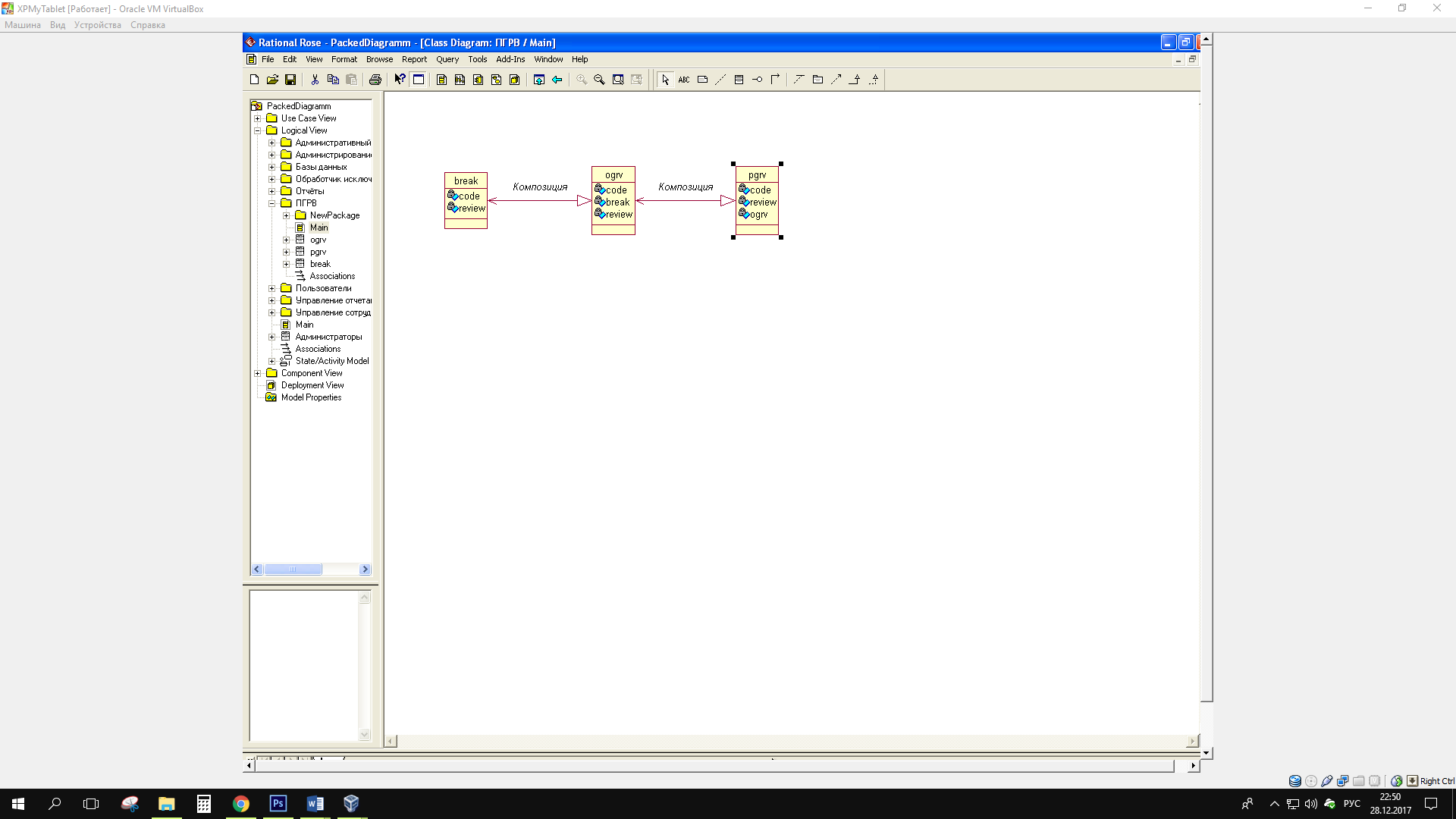


Рис. 19. Уточненная диаграмма классов пакета ПГРВ

### Построение детальной диаграммы классов

Построение детальной диаграммы классов предполагает проектирование окончательной структуры и поведения его объектов. Структура объектов определяется совокупностью атрибутов и операций класса. Каждый атрибут — это поле или совокупность полей данных, содержащихся в объекте класса.

На рис.20 представлена детальная диаграмма классов рассматриваемого пакета. В табл. 46 - 51 представлено описание полей и методов классов.

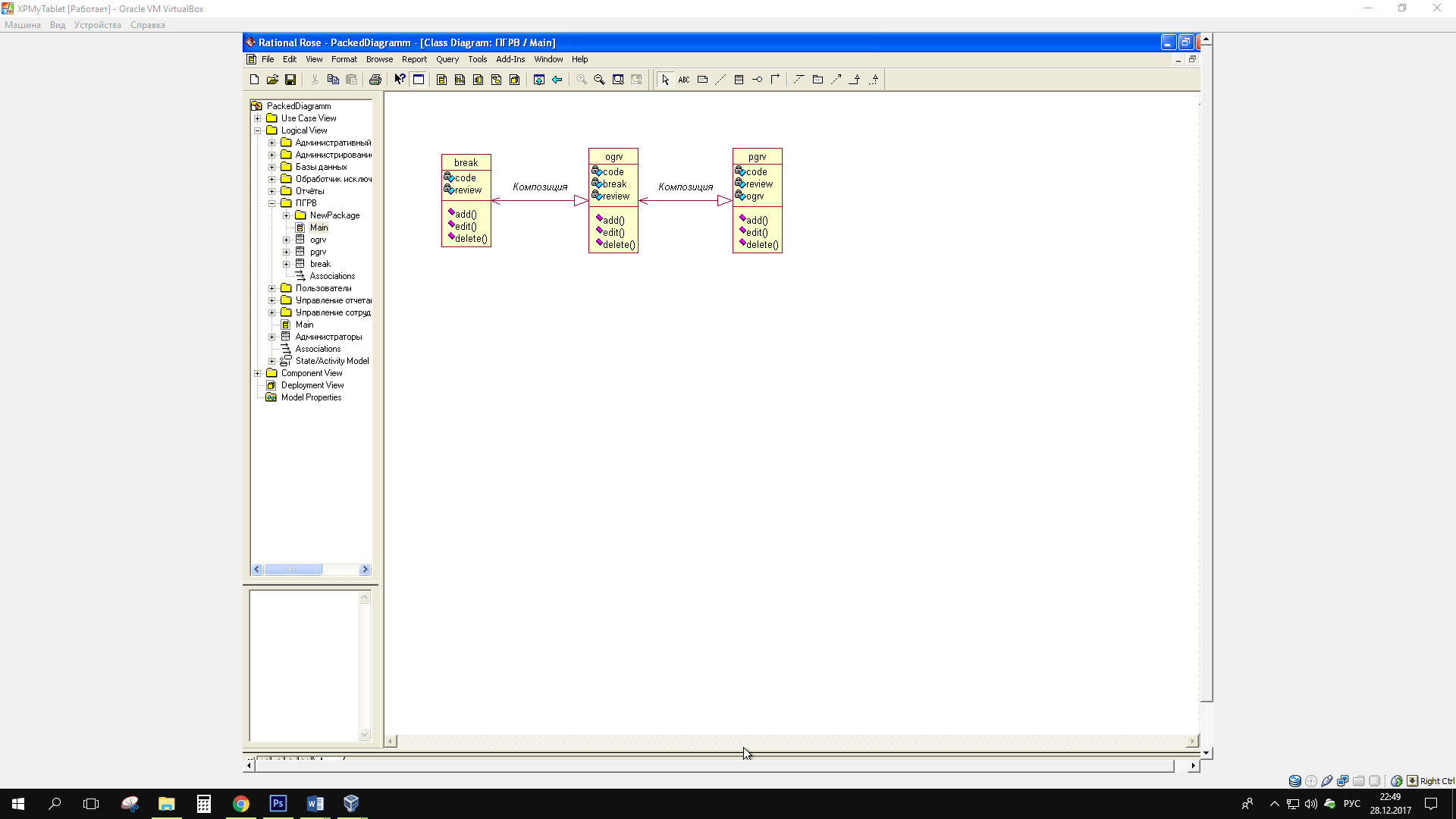


Рис. 20. Детальная диаграмма классов пакета ПГРВ

Таблица 46

Описание полей класса break

|  |  |
| --- | --- |
| Название поля | Назначение |
| code | Код перерыва |
| review | Описание перерыва |

Таблица 47

Описание методов класса break

|  |  |
| --- | --- |
| Название метода | Назначение |
| Add() | Создание перерыва |
| Edit() | Редактирование перерыва |
| Delete() | Удаление перерыва |

Таблица 48

Описание полей класса ogrv

|  |  |
| --- | --- |
| Название поля | Назначение |
| code | Код ОГРВ |
| break() | Объект класса break |
| review | Описание ОГРВ |

Таблица 49

Описание методов класса ogrv

|  |  |
| --- | --- |
| Название метода | Назначение |
| Add() | Создание ОГРВ |
| Edit() | Редактирование ОГРВ |
| Delete() | Удаление ОГРВ |

Таблица 50

Описание полей класса pgrv

|  |  |
| --- | --- |
| Название поля | Назначение |
| code | Код ПГРВ |
| ogrv() | Объект класса ОГРВ |
| review | Описание ПГРВ |

Таблица 51

Описание методов класса pgrv

|  |  |
| --- | --- |
| Название метода | Назначение |
| Add() | Создание ОГРВ |
| Edit() | Редактирование ОГРВ |
| Delete() | Удаление ОГРВ |

## Построение диаграмм компонентов

Диаграммы компонентов применяют при проектировании физической структуры разрабатываемого программного обеспечения. Эти диаграммы показывают, как выглядит программное обеспечение на физическом уровне, т.е. из каких частей оно состоит и как эти части связаны между собой [1]. В данной работе две подсистемы: HRSaveTime.exe и Server.exe, которые работают с БД. Диаграмма компонентов представлена на рис.21, описание компонентов предоставлено в табл. 52.



Рис. 21. Диаграмма компонентов

Таблица 52

Описание компонентов

|  |  |  |  |
| --- | --- | --- | --- |
| Наименование | Назначение | Входные данные | Выходные данные |
| HRSaveTime.exe | Для получения и обработки информации со считывающих устройств | ID карты | ID, время считывания, место, возможные ошибки |
| Server.exe | Для получения информации с пропускных карт и занесении данных в БД | Новые данные | Изменение в БД |
| База данных (БД) | Предназначена для хранения данных о сотрудниках | Данные, которые заполняются сотрудниками и HR Администратором (временные данные, пользовательские данные) | БД, содержащая информацию о сотрудниках |

## Диаграмма размещения

При физическом проектировании распределенных программных систем необходимо определить наиболее оптимальный вариант размещения программных компонентов на реальном оборудовании в локальной или глобальной сетях. Для этого используют специальную модель UML - диаграмму размещения (рис. 22).



Рис. 22. Диаграмма размещения

Компоненты ПО связаны между собой локальной сетью. Приложение Server.exe расположено на сервере, вместе с БД. Клиенты (сотрудники АО «СберТех») работают с ПО HRSaveTime.exe на своих персональных компьютерах.

# Кодирование

Кодирование клиентского и серверного ПО осуществлялось на языке высоко уровня C# в среде разработки Visual Studio 17. Текст программы представлен в Прил.2.

Кодирование микроконтроллеров – Arduino IDE. Текст прошивки микроконтроллера представлен в Прил.3.

# Тестирование

Тестирование – это ручной или автоматический процесс многократного выполнения программы с целью обнаружения ошибок [5].

Целью испытаний является выявление как можно большего количества ошибок, допущенных при проектировании и кодировании системы.

Тестирование проводится на основе чек-листа, приведенного в табл.53, используя «Ручной метод проверка за столом» [1]. Это проверка исходного текста или сквозные просмотры, выполняемые одним человеком, который читает текст программы, проверяет его по списку и пропускает через программу тестовые данные.

Таблица 53.

Чек-лист

|  |  |
| --- | --- |
| Название | Тестируемый модуль/функция |
| Интерфейс | MainWindow.xaml.cs/ private void Button\_Click |
| Профиль | Profile.cs/ private void myProfile\_Loaded/ public void Add/ |

Результаты тестирования ПО, разработанное в соответствии с требованиями технического задания (прил.1), приведены в табл.54-55

Таблица 54

Тестирование данных

|  |  |  |
| --- | --- | --- |
| Исходные данные | Тестируемый модуль или функция | Ожидаемый результат |
| Файл, хранящий данные для отображения профиля | MainWindow.xaml.cs | Данные из файла должны отобразиться на форме при отображения профиля |
| Profile.cs | Должен создаться объект класса с данными, загруженными из файла |

Таблица 55

Результаты тестирования

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Дата  тестирования | Тестируемый модуль или  функция | Кто  проводил  тестирование | Описание теста | Результаты  тестирования |
| 15.12.2017 | private void Button\_Click(object sender, RoutedEventArgs e) | Разработчик | Нажатие кнопки «Мой профиль» для открытия формы профиля сотрудника | Успех |
| 15.12.2017 | private void myProfile\_Loaded(object sender, RoutedEventArgs e) | Разработчик | Нажатие кнопки «Мой профиль» для отображения данных о сотруднике | При компиляции среда разработки выдавала ошибку о не возможности прочитать файл |
| 19.12.2017 | public void Add(string name, string pos, string birth, string wPhone, string mPhone, string hPhone, string sID, string vID) | Разработчик | Данные из файла загружаются в объект класса Profile | Не все данные из файла загружаются |
| 21.12.2017 | private void myProfile\_Loaded(object sender, RoutedEventArgs e) | Разработчик | Нажатие кнопки «Мой профиль» для отображения данных о сотруднике | Успех |
| 21.12.2017 | public void Add(string name, string pos, string birth, string wPhone, string mPhone, string hPhone, string sID, string vID) | Разработчик | Данные из файла загружаются в объект класса Profile | Успех |

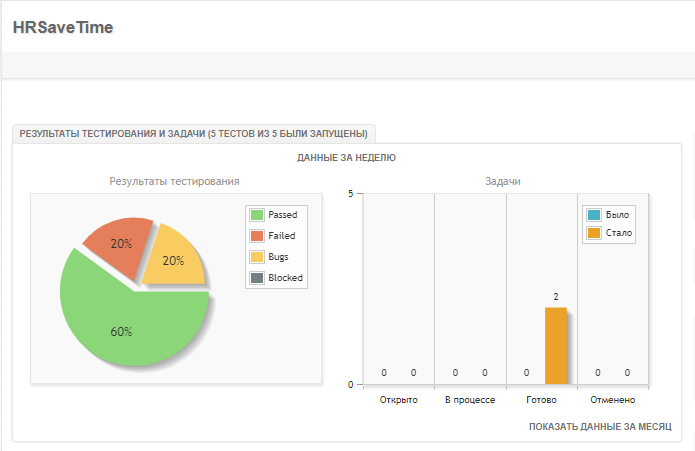


Рис.\* Результаты тестирования ПО в виде диаграмм

1. Администрирование

Основной целью администрирования информационных ресурсов ИС является поддержание информационных ресурсов в соответствии с заданными требованиями. Для достижения этой цели администрирование должно быть построено таким образом, чтобы минимизировать время и ресурсы, направляемые на управление информационными ресурсами и, в то же время, максимизировать доступность, производительность, продуктивность и защищенность информационных ресурсов автоматизированных систем [8].

В соответствии с рекомендациями ITU-T X.700 и стандартом ISO 7498-4 задачи администрирования информационных ресурсов автоматизированных систем можно разделить на пять функциональных групп [8]:

* управление конфигурацией;
* управление производительностью;
* управление использованием ресурсов;
* управление обработкой неисправностей;
* управление безопасностью.

Каждая ИС, в зависимости от назначения и средств ее организации, является индивидуальным компьютерным продуктом. Поэтому правила и условия администрирования жестко зависят от исходной ИС. Они не могут быть абсолютно универсальными или стандартными, но могут совпадать в принципиальном решении ключевых вопросов.

Компьютерная сеть представляет собой совокупность компьютеров или рабочих станций, объединенных средствами передачи информации. В зависимости от удаленности компьютеров сети условно разделяют на глобальные и локальные. Произвольная глобальная сеть может включать другие глобальные сети, локальные сети, а также отдельно подключенные компьютеры (удаленные компьютеры). В глобальной сети основным видом взаимодействия между компьютерами является обмен сообщениями. Более интенсивный обмен информацией реализуется в локальных сетях. В сетях, наряду с передачей информации, организовано управление аппаратно-программными ресурсами всех входящих в сеть компьютеров. Реализует эти функции сетевое ПО.

Необходимость администрирования сетевой ИС продиктована следующим [10].

1. ИС в простейшем случае может быть реализована в виде группы компьютеров пользователей, работающих под управлением одного или нескольких серверов (сети, базы данных, печати и т.д.). Для успешной работы ИС необходимо, чтобы кто-то нес за нее ответственность и следил за повседневным обслуживанием компьютеров, их модернизацией, добавлением оперативной памяти, увеличением объема жестких дисков и установкой ПО.

2. В ИС компьютеры-клиенты и компьютеры-серверы подключены к общей сети. Если сеть не будет работать, то передача информации между сервером и компьютерами-клиентами станет невозможной. Следовательно, кто-то должен отвечать за работу сети и управлять ею.

3. На всех компьютерах-клиентах и компьютерах-серверах используются операционные системы (ОС) Windows. Все клиенты и серверы должны быть правильно сконфигурированы. За исполнением и соответствием этого кто-то также должен следить.

4. Отображение искомой информации в ИС осуществляется на мониторах компьютеров-клиентов, а ее документирование – с помощью принтеров и плоттеров, находящихся под управлением сервера печати. Сервер печати (СП) нуждается в общем техническом обслуживании, создании резервных копий, необходимых для предотвращения потери и модернизации данных. При этом кто-то должен следить за созданием новых приложений и поддерживать стабильность работы СП. Так как, если будет невозможно напечатать запрашиваемую информацию, то пользователи ИС не смогут выполнить свою работу.

5. Управление БД осуществляется с помощью системы управления базами данных (СУБД). Сервер БД данных необходимо корректно установить и настроить, чтобы обеспечить его оптимальную производительность. Важной задачей является также защита информации в базе, так как ее утрата или искажение может привести к значительным финансовым потерям. При эксплуатации ИС требуется предусмотреть, чтобы кто-то осуществлял восстановление потерянных данных, обеспечивал безопасность данных за счет учета имеющихся и новых пользователей, распределение прав доступа к информации.

# Заключение

В результате выполнения курсовой работы на тему «Администрирование подсистемы учета рабочего времени и местонахождения сотрудников АО «СберТех» была спроектирована информационная система, взаимодействующая с пропускными системами, работающие на базе микроконтроллера Arduino.

Помимо этого, были закреплены такие компетенции, как способность выбирать и оценивать способ реализации информационных систем и устройств (программно-, аппаратно- или программно-аппаратно-) для решения поставленной задачи (ОПК-6), выбирать и оценивать способ реализации информационных систем и устройств (программно-, аппаратно- или программно-аппаратно-) для решения поставленной задачи (ПК-2), проводить техническое проектирование (ПК-3), проводить рабочее проектирование (ПК-4), проводить выбор исходных данных для проектирования (ПК-10), разрабатывать, согласовывать и выпускать все виды проектной документации. Данная система позволяет сотрудникам оформлять отсутствия (отгул, командировку, больничный), а HR Администратору контролировать рабочее время, всего персонала.

Внедрение данной подсистемы в офис АО «СберТех» позволит HR Администратору автоматизировать процесс ведения временных данных по каждому сотурднику.

Список литературы

1. Технология программирования – Г.И. Иванова; издательство МГТУ им. Н.Э. Баумана, 2002 год; ISBN 6-57038-2077-4;
2. Моделирование и анализ систем. IDEF-технологии: практикум – С.В.Черемных; издательство Финансы и статистика; ISBN 5-279-02564-Х;
3. https://sberbank-talents.ru/ - официальный сайт АО «Сбертех»;
4. https://ru.wikipedia.org/wiki/ - Википедия;
5. http://www.protesting.ru/testing/ - тестирование программного обеспечения - основные понятия и определения;
6. https://habrahabr.ru/ - обзор IDEF0, IDEF3, DFD;
7. https://www.intuit.ru/studies/courses/2195/55/lecture/1628?page=5 – Методологии моделирования предметной области.
8. https://cyberleninka.ru/article/n/metody-administrirovaniya-informatsionnyh-resursov-avtomatizirovannyh-sistem - Методы администрирования информационных ресурсов автоматизированных систем

# ПРИЛОЖЕНИЕ 1

МИНОБРНАУКИ РОССИИ

федеральное государственное бюджетное образовательное учреждение

высшего образования

«ЧЕРЕПОВЕЦКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»

Институт информационных технологий

Кафедра математического и программного обеспечения ЭВМ

Дисциплина: «Администрирование информационных систем»

|  |
| --- |
| УТВЕРЖДАЮ |
| Зав. кафедрой МПО ЭВМ |
| д.т.н., профессор \_\_\_\_\_\_\_\_ Ершов Е.В. |
| «\_\_\_\_» \_\_\_\_\_\_\_\_\_\_\_ 2018 г. |

Администрирование подсистемы учета рабочего времени и местоположения сотрудников АО «Сбербанк-Технологии»

Техническое задание на курсовую работу

Листов \_\_\_

|  |  |  |
| --- | --- | --- |
|  | Руководитель: | доцент Селяничев О.Л. |
|  | Исполнитель: | студент гр. 1ИСб-00-41оп |
|  |  | Белов А.С. |

2018 г.

Введение

Программное обеспечение – HRSaveTime, должно автоматизировать ведение уходов и приходов сотрудников в АО «СберТех» с использованием пластиковых пропускных карт, а также оформление и согласование отгулов, больничных, командировок, отпусков и прогулов с возможностью выводить информацию в Excel-документ.

1. Основания разработки и область применения

Основанием для разработки служит задание на курсовой проект по дисциплине «Администрирование информационных систем», выданное на кафедре МПО ЭВМ ИИТ ЧГУ.

Дата утверждения: 9 февраля 2018 года.

Наименование темы разработки: Администрирование подсистемы учета рабочего времени и местоположения сотрудников АО «Сбербанк-Технологии».

1. Назначение разработки

Программа должна автоматически фиксировать с помощью пластиковых карт точное время прихода и ухода каждого сотрудника, занося информацию в соответствующую таблицу в БД.

Помимо автоматической фиксации времени, система позволит сотрудникам самостоятельно оформлять заявки на отпуск, отгул или командировку и отправлять её на согласование линейному руководителю, после чего HR Администратору.

HR Администратор сможет самостоятельно в системе заносить данные по отгулам, больничным, командировкам, отпускам и прогулам.

Система должна способна самостоятельно производить вычисления времени, такие как:

* фактическое время работы сотрудника за период (разница между временем с момента ухода и моментом прихода в офис в течении суток);
* фактическое время переработки за период (разница между фактическим и плановым временем ухода);
* количество дней, которое сотрудник отсутствовал по какой-либо причине за период;
* количество дней накопленного отпуска (из расчета 28 дней, в течении года).

Вся данная информация должна попадать в отчёты за период по одному или нескольким сотрудникам (табл.П1.1. – табл.П1.5).

1. Требования к программе

Подсистема учета рабочего времени и местоположения сотрудников АО «СберТех» должна состоять из двух подпрограмм:

* Клиентское приложение, содержащее функционал для удобного ведения данных по сотрудникам. С данной программой будут работать сотрудники отдела.
* Серверное приложение, содержащее функционал для работы с пропускными картами и БД банка.
  1. Требования к функциональным характеристикам клиентского приложения

С клиентским приложением будет работать HR Администратор и другие сотрудники АО «СберТех».

Для работы HR Администратору будущая программа должна обладать следующим функционалом:

* возможность создания персонального профиля в программе для каждого сотрудника, в котором будет отражена следующая информация:
  + фамилия, имя, отчество;
  + дата рождения в формате ДД.ММ.ГГГГ;
  + текущая должность;
  + локальное местоположение на территории офиса в формате № кабинета – наименование кабинета;
  + контактная информация (рабочий телефон, мобильный телефон, e-mail адрес);
  + код персонального графика рабочего времени;
  + таблица «Присутствия», содержащая поля, приведенные в табл.П1.6, позволяющая хранить точное время прихода и ухода сотрудника с рабочего места;
  + таблица «Отсутствия», содержащая поля, приведенные в табл.П1.7, позволяющая хранить данные по временным отсутствиям, а также нетрудоспособности сотрудника.
* возможность редактирования и удаления персонального профиля сотрудника;
* возможность создание персонального графика рабочего времени, отражающего следующую информацию:
  + перерыв – время обеденного перерыва;
  + ОГРВ – однодневный график рабочего времени, который отражает суточную часовую норму, которую должен отработать сотрудник (не включая время на перерыв);
  + Вид графика – 5 через 2 (5/2), 2 через 2 (2/2) или индивидуальный;
  + Описание, в котором указывается краткая информация о графике.
* возможность редактирование и удаления графика рабочего времени;
* возможность оформления отсутствий:
  + прогул – беспричинное отсутствие на рабочем месте;
  + отгул – отсутствие на рабочем месте по договоренности с линейным руководителем;
  + больничный - отсутствие на рабочем месте в связи с болезнью;
  + отпуск:
    - Основной – за год работы накапливается 28 календарный дней;
    - Дополнительный – 3 дня в году;
    - Декрет – в связи с рождением и воспитанием ребенка до 3-ёх лет.
  + Командировка – отсутствие на рабочем месте в связи с выполнением рабочего задания вне офиса.
* регистрация пропускной карты;
* формирование отчетов:
  + отчет по одному или нескольким сотрудникам за период в формате:

Таблица П1.1

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Табельный номер | Фамилия | Имя | Отчество | ПЛАН: Норма рабочего времени (за период в часах) | ФАКТ: Отработано времени (в часах) | ПЛАН:  Норма рабочего времени (с учетом и больничных) | ФАКТ: (Норма рабочего времени (с учетом и больничных) | Отработано менее нормы | Отработано в выходные (в часах) | Количество выходов в выходные дни (в днях) | Количество опозданий (в днях) | Общее время опозданий (в часах) | Количество переработок (в днях) | Количество переработок (разница факт-план) |

Таблица П1.1. Продолжение

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Была командировка? | Количество дней в командировке | Был больничный? | Количество дней больничного | Был отпуск? | Количество дней отпуска | Было обучение? | Количество дней в обучении? | Прогул (в днях) | Отгул (в днях) |

* + отчет по больничным у одного или нескольких сотрудников за период в формате:

Таблица П1.2

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Табельный номер | Фамилия | Имя | Отчество | Количество дней | Дата с | Дата по | Документ (ссылкой на документ) |

* + отчет по отгулам у одного или нескольких сотрудников за период в формате:

Таблица П1.2

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Табельный номер | Фамилия | Имя | Отчество | Количество дней | Дата с | Дата по | Табельный номер согласующего | Фамилия | Имя | Отчество |

* + отчет по прогулов у одного или нескольких сотрудников за период в формате:

Таблица П1.3

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Табельный номер | Фамилия | Имя | Отчество | Количество дней | Дата с | Дата по |

* + отчет по командировкам у одного или нескольких сотрудников за период в формате:

Таблица П1.4

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Табельный номер | Фамилия | Имя | Отчество | Количество дней | Дата с | Дата по | Цель | Место |

* + отчет по одному или нескольким сотрудникам в формате:

Таблица П1.5

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Табельный номер | Фамилия | Имя | Отчество | Дата принятия на работу | Дата Увольнения | Должность | Код персонального графика рабочего времени |

В ПО должны быть предусмотрены роли для разграничения полномочий сотрудников:

* sysAdmin
* HRAdmin – полный доступ к функционалу программы;
* Wman – ограниченный доступ;
* возможность создавать свои роли с настраиваемым функционалом.

Роль HRAdmin присваивается HR Администратору.

Роль Wman должна присваиваться HR Администратором новым сотрудникам, которым будет доступен следующий функционал:

* При просмотре своего профиля, сотруднику доступна вся информация;
* При просмотре чужого профиля, сотруднику доступна следующая информация:
  + фамилия, имя, отчество;
  + дата рождения;
  + нынешняя должность;
  + локальное местоположение на территории офиса;
  + участником какого проекта является в данный момент;
  + контактная информация (рабочий телефон, мобильный телефон, e-mail адрес).
* Возможность редактировать свои данные:
  + рабочий телефон;
  + мобильный телефон;
  + e-mail адрес;
  + сменить пароль для входа в систему.

Таблица П1.6

Описание полей таблицы «Присутствия»

|  |  |
| --- | --- |
| Имя поля | Предполагаемый тип поля |
| Дата | Дата |
| Время с | Время |
| Время до | Время |
| Полный день | Болевая переменная |

Таблица П1.7

Описание полей таблицы «Отсутствия»

|  |  |
| --- | --- |
| Имя поля | Предполагаемый тип поля |
| Дата с | Дата |
| Дата по | Дата |
| Вид отсутствия | Текст |
| Документ | Ссылка на документ |

Для запуска нужного профиля, в системе должен быть процесс авторизации, в котором сотруднику предлагается ввести свой табельный номер и пароль, который генерируется HR Администратором случайным образом в момент создания профиля сотрудника. В последствии HR Администратор может изменить пароль у любого сотрудника лично, либо сгенерировать повторно.

После ввода данных, система обращается на сервер к БД, в которой хранятся табельный номер сотрудника, пароль и роль. В зависимости от последнего, программа отображает только те элементы интерфейса, которые доступны для данной роли.

* 1. Требования к функциональным характеристикам серверного приложения

Серверное приложение должно быть реализовано в консольном виде, которое обладает следующим функционалом:

* получение от микроконтроллера Arduino данные в виде id-адреса, передаваемые от приложенного к считывающему устройству пропускной карты;
* на основе полученного id-адреса, внести запись о приходе, уходе и местоположении по конкретному сотруднику;
* при считывании id-адреса выводить информации в реальном времени на экран в виде:

[ДД:ММ:ГГГ] [ЧЧ:ММ:СС] [ID-адрес карты] [ФИО сотрудника];

* с использованием консольных команд реализовать возможность запуска, остановки и паузы получения данных со считывающего устройства;
* с использованием консольных команд реализовать возможность выводить информацию по заданному id-адресу на экран в виде:

[ID-адрес карты] [ФИО сотрудника] [Статус «В офисе»/«Вне офиса»].

* 1. Требования к надежности

Для безопасного функционирования программ необходимо выполнить следующие требования:

* программа будет работать на рабочих станциях сотрудников, находящихся в локальной сети Сбербанка. Поэтому, передача информации от клиентов к серверу будет осуществлена по защищённой локальной сети;
* сервер должен быть подключён к сети через источник бесперебойного питания для фиксации последних транзакций;
* в программе должны быть предусмотрены всевозможные ошибки, текст которых должен выводиться пользователю в доступной форме. Все ошибки должны записываться в log-файл, хранящийся в папке с клиентским приложением;
  1. Условия эксплуатации

Климатические условия эксплуатации, при которых должны обеспечиваться заданные характеристики, должны удовлетворять требованиям, предъявляемым к техническим средствам в части условий их эксплуатации.

* 1. Требования к составу и параметрам технических средств

Для эксплуатации клиентского ПО необходимо иметь:

* персональный компьютер или ноутбук со схожими характеристиками:
  + операционная система Windows 10;
  + процессор Intel Core i3;
  + свободное место на жёстком диске 300 Мбайт;
  + объём оперативной памяти 4 Гбайта;
  + клавиатура;
  + мышь;
  + монитор с разрешением 1280 на 720 пикселей.

Для эксплуатации серверного ПО необходимо иметь:

* сервер со схожими характеристиками:
  + операционная система Windows Server 2016;
  + процессор Intel Xenon E3 1200;
  + свободное место на жёстком диске 100 Мбайт;
  + объём оперативной памяти 4 Гбайта;
  + клавиатура;
  + мышь;
  + монитор с разрешением 600 на 480 пикселей.
* считывающие устройства пропускных карт;
* пропускные пластиковые карты.
  1. Требования к информационной и программной совместимости
     1. Требования к исходным кодам и языкам программирования

Клиентское приложение для сотрудников должно быть разработано с учётом следующих рекомендаций:

* код программы реализован на языке высоко уровня C# с применением объектно-ориентированного подхода;
* интерфейс ПО выполнен с использованием WPF;
* компиляция кода выполнена в среде программирования Visual Studio 17.

Сервеное приложение должно быть разработано с учётом следующих рекомендаций:

* код программы реализован на языке высоко уровня C# с применением объектно-ориентированного подхода;
* интерфейс ПО выполнен в консольном виде;
* компиляция кода выполнена в среде программирования Visual Studio 17.

Устройства считывания пропускных кард должны быть реализованы с учётом следующих рекомендаций:

* компиляция кода прошивки микроконтроллера Arduino необходимо выполнить в среде разработки Arduino IDE;
* код программы реализован на C-подобном языке Arduino.
  + 1. Требования к программным средствам, используемым программой

Для успешного функционирования программы на клиенте и сервере должна быть установлена библиотека .NET Framwork 4.0.

* + 1. Требования к защите информации и программ
* для осуществления входа в программу, сотруднику необходимо ввести свои логин и пароль, которые должны храниться в БД в зашифрованном виде;
* в момент ввода пароля сотрудником, поле ввода не должно отображать введенный пароль. Каждый символ должен отображаться в виде снежинки «\*»;
  + 1. Требования к упаковке и маркировке

Диск с программой должен быть упакован в бумажный конверт, прикреплённый к папке с документацией. Диск и его упаковка должны содержать информацию о названии программы и ФИО разработчика программы.

* + 1. Требования к хранению

Курсовой проект необходимо хранить в сухом помещении в дали от источников воды и огня.

* + 1. Специальные требования

Вместе с документацией и диском, содержащим две программы, необходимо предоставить прототип системы считывания пропускных карт с инструкцией по подключению.

1. Требования к программной документации
   1. Структурные требования к расчётно-пояснительной записке

Титульный лист

Оглавление

Введение

1. Описание предметной области

2. Выбор технологии, среды и языка программирования

2.1. Выбор жизненного цикла

2.2. Выбор подхода к разработке

2.3. Выбор инструментальных средств

3. Выбор структур данных для решения поставленной задач

4. Разработка спецификаций проектируемой системы

4.1. Построение диаграмм вариантов использования

4.2 Построение контекстной диаграммы классов

4.3 Построение диаграмм последовательностей системы

4.4. Построение диаграмм деятельностей сценариев вариантов использования

5. Проектирование системы

5.1. Построение структуры системы и диаграмм пакетов

5.2. Построение диаграмм компонентов

5.3. Диаграмма размещения

6. Кодирование

7. Тестирование

Заключение

Список литературы

Приложение 1. Техническое задание

Приложение 2. Текст программы

Приложение 3. Руководство пользователя

* 1. Требования к оформлению программной документации

При оформлении текстовых и графических материалов, следует придерживаться общих требований ЕСПД.

Текстовые документы оформляются на белых листах формата А4 (210х297 мм). В соответствии с общими требованиями поля листа определяются следующим образом: левое – 30 мм, правое – 10 мм, верхнее – 20 мм, нижнее – 20 мм. Формат текста: Word for Windows, через один интервала, шрифт – Times New Roman Cyr, размер шрифта – 14, отступ первой строки абзаца – 0,75 см, выравнивание – по ширине. Количество знаков в строке, считая пробелы, не должно превышать 60 символов. Текст программы может быть расположен в две колонки, шрифт – Times New Roman Cyr, размер шрифта – 8.

Номер страницы проставляется в середине верхнего поля страницы арабской цифрой. Первая страница – титульный лист, вторая страница – аннотация, с третьей страницы начинается оглавление. Номер страницы на титульном листе не проставляется.

Наименование разделов, подразделов, пунктов должно быть кратким и соответствовать содержанию. Каждая новая глава печатается с новой страницы. Наименование разделов (основных частей) пишется прописными буквами по центру строки. Расстояние между заголовками и текстом, а также между заголовками разделов и подразделов должно быть равно двум интервалам.

Наименования подразделов и пунктов размещаются с абзацного отступа (0,75 см) и печатаются с прописной буквы, без подчеркивания и без точки в конце.

При использовании ссылок на пункты, разделы и подразделы указывается порядковый номер раздела или пункта (например: «в разд. 2», «в п. 2.3.1»).

Рисунки, таблицы и формулы нумеруются арабскими цифрами последовательно, так называемая сквозная нумерация, или в пределах раздела (относительная нумерация). В приложении нумеруются в пределах приложения. Каждый рисунок имеет порядковый номер и название, помещаемые под рисунком по центру.

Таблицы в основном применяются для оформления цифрового материала. Шрифт – Times New Roman Cyr, размер шрифта – 14. Иногда возможен 10-й размер шрифта. Обозначения единиц физических величин необходимо применять в системе СИ. Номер таблицы размещается в правом верхнем углу над заголовком.

5. Стадии и этапы разработки

Стадии и этапы разработки представлены в табл. П1.8

Таблица П1.8

Стадии и этапы разработки

|  |  |  |  |
| --- | --- | --- | --- |
| Наименование этапа  разработки ПО | Сроки  Разработки | Результат  выполнения | Отметка о  выполнении |
| 1 | 2 | 3 | 4 |
| Программирование микроконтроллера Arduino | 27.03.2018 | Считывание датчиком с пропускных карт их ID-адрес |  |
| Разработка серверного приложения | 30.032018 | Консольное приложение, обладающее функционалом, в соответствии с п.3.2. |  |
| Разработка интерфейса пользователя | 03.04.2018 | Программа с пользовательским интерфейсом |  |

Таблица П1.8. Продолжение

|  |  |  |  |
| --- | --- | --- | --- |
| 1 | 2 | 3 | 4 |
| Разработка клиентского приложения | 04.04.2018 | Клиентское приложение, обладающее функционалом, в соответствии с п.3.2. |  |
| Тестирование | 11.04.2018 | Отчёт о результатах тестирования ИС, с использованием тестовых данных |  |
| Внедрение | 13.04.2018 | Внедрение ИС на предприятие Заказчика |  |
| Администрирование | 16.04.20180 | Выполненные  требования по  администрированию ИС |  |
| Демонстрация работы программы | 17.04.2018 | Презентация ИС Заказчику |  |

7. Порядок контроля и приемки

Порядок контроля и приёма работ представлен в табл.П1.9.

8. Виды испытаний

После выполнения разработки программ, необходимо провести следующие тестирования с документированием результатов:

* функциональное тестирование (Functional testing);
* тестирование безопасности (Security and Access Control Testing);
* тестирование удобства пользования или Usability Testing.

Таблица П1.9

Порядок контроля и приемки

|  |  |  |  |
| --- | --- | --- | --- |
| Наименование  контрольного этапа  выполнения  курсовой работы | Сроки  контроля | Результат выполнения | Отметка о приемке  результата  контрольного этапа |
| Выдача задания | 26.03.18 | Получено задание на выполнение курсового проекта |  |
| Техническое задание | 27.03.18 | Документ «Техническое задание» |  |
| Концептуальное, логическое и физическое проектирование ИС | 30.03.18 | Подготовлены схемы и диаграммы основных процессов программ |  |
| Расчётно-пояснительная записка | 15.04.18 | Документ «Расчётно-пояснительная записка» |  |
| Защита курсовой работы | 23.04.18 | Защита курсовой работы |  |

ПРИЛОЖЕНИЕ 2

# Текст программы

Profile.cs

using System;

using System.Collections.Generic;

using System.Text;

namespace WpfApplication1

{

class Profile

{

//head

public string Name;

public string Position;

public string Birth;

//contacts

public string workPhone;

public string mobilePhone;

public string homePhone;

public string skypeID;

public string vkID;

public void Add(string name, string pos, string birth, string wPhone, string mPhone, string hPhone, string sID, string vID)

{

Name = name;

Position = pos;

Birth = birth;

workPhone = wPhone;

mobilePhone = mPhone;

homePhone = hPhone;

skypeID = sID;

vkID = vID;

}

}

}

Рис. 1. Текст класса Profile

MainWindow.xaml.cs

using System;

using System.Collections.Generic;

using System.Text;

using System.Windows;

using System.Windows.Controls;

using System.Windows.Data;

using System.Windows.Documents;

using System.Windows.Input;

using System.Windows.Media;

using System.Windows.Media.Imaging;

using System.Windows.Navigation;

using System.Windows.Shapes;

using System.IO;

namespace WpfApplication1

{

/// <summary>

/// Логика взаимодействия для MainWindow.xaml

/// </summary>

Рис. 2. Текст модуля MainWindow.xaml.cs

public partial class MainWindow : Window

{

public MainWindow()

{

InitializeComponent();

}

private void Button\_Click(object sender, RoutedEventArgs e)

{

myProfile.Visibility = Visibility.Visible;

}

private void btn\_home\_Click(object sender, RoutedEventArgs e)

{

myProfile.Visibility = Visibility.Hidden;

}

private void myProfile\_Loaded(object sender, RoutedEventArgs e)

{

List<string> list = new List<string>();

Profile myProfile = new Profile();

using (StreamReader sr = new StreamReader("myProfile.txt", encoding: Encoding.GetEncoding(1251)))

{

myProfile.Add(sr.ReadLine(), sr.ReadLine(), sr.ReadLine(), sr.ReadLine(), sr.ReadLine(), sr.ReadLine(), sr.ReadLine(), sr.ReadLine());

};

l\_name.Content = myProfile.Name;

l\_position.Content = myProfile.Position;

l\_birth.Content = myProfile.Birth;

l\_work\_phone.Content = myProfile.workPhone;

l\_mobile\_phone.Content = myProfile.mobilePhone;

l\_home\_phone.Content = myProfile.homePhone;

l\_skype.Content = myProfile.skypeID;

l\_vk\_id.Content = myProfile.vkID;

}

private void Window\_Loaded(object sender, RoutedEventArgs e)

{

myProfile.Visibility = Visibility.Hidden;

}

}

}

Рис. 2. Продолжение

MainWindow.xaml

<Window x:Class="WpfApplication1.MainWindow"

xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"

xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"

Title="Главно рабочее окно" Height="700" Width="1024" ResizeMode="NoResize" Loaded="Window\_Loaded">

<Grid>

<Grid.Background>

<ImageBrush/>

</Grid.Background>

<Grid.ColumnDefinitions>

<ColumnDefinition Width="115\*"/>

<ColumnDefinition Width="168\*"/>

<ColumnDefinition Width="733\*"/>

</Grid.ColumnDefinitions>

<Frame Content="" Height="75.793" Margin="0" VerticalAlignment="Top" BorderBrush="#FF23FF00" Background="#FF00CD5D" Grid.ColumnSpan="3"/>

Рис. 3. Текст модуля MainWindow.xaml

<Image HorizontalAlignment="Right" Margin="0,11,11,0" Width="246.253" Stretch="Fill" Height="53.4" VerticalAlignment="Top" Grid.Column="2" Source="C:\Users\Антон\Desktop\Диплом\WpfApplication1\WpfApplication1\sber\_logo.png">

<Image.OpacityMask>

<ImageBrush ImageSource="C:\Users\Антон\Desktop\Диплом\WpfApplication1\WpfApplication1\sber\_logo.png"/>

</Image.OpacityMask>

</Image>

<Frame Content="" Margin="0,75.88,0,0" Height="40.49" VerticalAlignment="Top" Background="#FF009142" Grid.ColumnSpan="3"/>

<Button Content="МОЙ ПРОФИЛЬ" HorizontalAlignment="Left" Margin="10.086,81.934,0,0" VerticalAlignment="Top" Width="111.311" Height="28.029" Background="{x:Null}" BorderBrush="#FF00CD5D" Foreground="White" Grid.ColumnSpan="2" Click="Button\_Click"/>

<Button x:Name="btn\_home" Margin="0,131,0,0" VerticalAlignment="Top" Height="72" BorderBrush="White" FontSize="24" BorderThickness="0" RenderTransformOrigin="0.5,0.5" Grid.ColumnSpan="2" Background="{x:Null}" Width="280.666" Click="btn\_home\_Click" HorizontalAlignment="Left">

<StackPanel Orientation="Horizontal" Width="278.222" HorizontalAlignment="Left">

<Image Source="C:\Users\Антон\Desktop\Диплом\WpfApplication1\WpfApplication1\home.png" Margin="28,0,0,0"/>

<TextBlock Text="ГЛАВНАЯ" Width="155.407" TextAlignment="Center" HorizontalAlignment="Center" VerticalAlignment="Center" Margin="-6,0,0,0"/>

</StackPanel>

</Button>

<Button Margin="0,222.5,0,0" BorderBrush="White" FontSize="24" BorderThickness="0" RenderTransformOrigin="0.5,0.5" Background="{x:Null}" Grid.ColumnSpan="2" Height="72" VerticalAlignment="Top" Width="280.666" HorizontalAlignment="Left">

<StackPanel Orientation="Horizontal" Width="278.555" HorizontalAlignment="Left">

<Image Source="C:\Users\Антон\Desktop\Диплом\WpfApplication1\WpfApplication1\calendar.png" Margin="29,0,0,0"/>

<TextBlock Width="155.407" TextAlignment="Center" HorizontalAlignment="Center" VerticalAlignment="Center" Margin="-5,0,0,0"><Run Language="ru-ru" Text="ГРАФИКИ"/></TextBlock>

</StackPanel>

</Button>

<Button Margin="0,315,0,0" VerticalAlignment="Top" Height="77" BorderBrush="White" FontSize="24" BorderThickness="0" RenderTransformOrigin="0.5,0.5" Background="{x:Null}" Grid.ColumnSpan="2" Width="280.666" HorizontalAlignment="Left">

<StackPanel Orientation="Horizontal" Width="279.722" Height="74" HorizontalAlignment="Left">

<Image Source="C:\Users\Антон\Desktop\Диплом\WpfApplication1\WpfApplication1\document.png" Margin="31,0,0,0" Height="73" Width="65.462"/>

<TextBlock Width="155.407" TextAlignment="Center" Margin="15,0,0,0" HorizontalAlignment="Center" VerticalAlignment="Center"><Run Language="ru-ru" Text="ОТСУТСТВИЯ"/></TextBlock>

</StackPanel>

</Button>

<Button Margin="0,405,0,0" VerticalAlignment="Top" Height="72" BorderBrush="White" FontSize="24" BorderThickness="0" RenderTransformOrigin="0.5,0.5" Background="{x:Null}" Grid.ColumnSpan="2" Width="280.666" HorizontalAlignment="Left">

<StackPanel Orientation="Horizontal" Width="279.555" HorizontalAlignment="Left">

<Image Source="C:\Users\Антон\Desktop\Диплом\WpfApplication1\WpfApplication1\people.png" Margin="33,0,0,0" Height="66" Width="62.148"/>

<TextBlock Width="155.407" TextAlignment="Center" HorizontalAlignment="Center" VerticalAlignment="Center" Margin="6,0,0,0"><Run Language="ru-ru" Text="ПЕРСОНАЛ"/></TextBlock>

</StackPanel>

</Button>

<Image Grid.Column="2" Height="34" Margin="0,79,50,0" VerticalAlignment="Top" Source="C:\Users\Антон\Desktop\Диплом\WpfApplication1\WpfApplication1\push.png" HorizontalAlignment="Right" Width="34">

<Image.OpacityMask>

<ImageBrush ImageSource="C:\Users\Антон\Desktop\Диплом\WpfApplication1\WpfApplication1\push.png"/>

</Image.OpacityMask>

</Image>

<Label Content="0" Grid.Column="2" Margin="0,77.52,52.503,0" VerticalAlignment="Top" Height="18.96" FontSize="9" BorderThickness="1,0,0,0" HorizontalAlignment="Right" Width="16.993"/>

<Image Grid.Column="2" Height="34" Margin="0,79,11,0" VerticalAlignment="Top" Source="C:\Users\Антон\Desktop\Диплом\WpfApplication1\WpfApplication1\out.png" HorizontalAlignment="Right" Width="34">

<Image.OpacityMask>

<ImageBrush ImageSource="C:\Users\Антон\Desktop\Диплом\WpfApplication1\WpfApplication1\out.png"/>

</Image.OpacityMask>

</Image>

Рис. 3. Продолжение

<Grid x:Name="myProfile" Grid.Column="2" Margin="0,117,0,0" Background="WhiteSmoke" Loaded="myProfile\_Loaded">

<Image HorizontalAlignment="Right" Height="100" Margin="0,65,20,0" VerticalAlignment="Top" Width="100" Source="calendar.png">

<Image.OpacityMask>

<ImageBrush ImageSource="calendar.png"/>

</Image.OpacityMask>

</Image>

<Label Content="Мой профиль" HorizontalAlignment="Left" Margin="20,20,0,0" VerticalAlignment="Top" FontSize="16"/>

<Label x:Name="l\_name" Content="Label" HorizontalAlignment="Left" Margin="40,80,0,0" VerticalAlignment="Top" RenderTransformOrigin="0.5,0.5" FontSize="24"/>

<Label x:Name="l\_position" Content="Label" HorizontalAlignment="Left" Margin="40,160,0,0" VerticalAlignment="Top" FontSize="14"/>

<Label x:Name="l\_birth" Content="Label" HorizontalAlignment="Left" Margin="301,160,0,0" VerticalAlignment="Top" FontSize="14"/>

<Label Content="Должность:" HorizontalAlignment="Left" Margin="40,130,0,0" VerticalAlignment="Top" FontSize="14"/>

<Label Content="Дата рождения:" HorizontalAlignment="Left" Margin="301,130,0,0" VerticalAlignment="Top" FontSize="14"/>

<Label Content="Контактные данные:" HorizontalAlignment="Left" Margin="40,223,0,0" VerticalAlignment="Top" FontSize="14"/>

<Label Content="Рабочий телефон:" HorizontalAlignment="Left" Margin="40,260,0,0" VerticalAlignment="Top" FontSize="14"/>

<Label Content="Сотовый телефон:" HorizontalAlignment="Left" Margin="40,290,0,0" VerticalAlignment="Top" FontSize="14"/>

<Label Content="Домашний телефон:" HorizontalAlignment="Left" Margin="40,320,0,0" VerticalAlignment="Top" FontSize="14"/>

<Label Content="Skype:" HorizontalAlignment="Left" Margin="40,350,0,0" VerticalAlignment="Top" FontSize="14"/>

<Label Content="Вконтакте:" HorizontalAlignment="Left" Margin="40,380,0,0" VerticalAlignment="Top" FontSize="14"/>

<Label x:Name="l\_work\_phone" Content="null" HorizontalAlignment="Left" Margin="197,260,0,0" VerticalAlignment="Top" FontSize="14"/>

<Label x:Name="l\_mobile\_phone" Content="null" HorizontalAlignment="Left" Margin="197,290,0,0" VerticalAlignment="Top" FontSize="14"/>

<Label x:Name="l\_home\_phone" Content="null" HorizontalAlignment="Left" Margin="197,320,0,0" VerticalAlignment="Top" FontSize="14"/>

<Label x:Name="l\_skype" Content="null" HorizontalAlignment="Left" Margin="197,350,0,0" VerticalAlignment="Top" FontSize="14"/>

<Label x:Name="l\_vk\_id" Content="null" HorizontalAlignment="Left" Margin="197,380,0,0" VerticalAlignment="Top" FontSize="14"/>

</Grid>

</Grid>

</Window>

Рис. 3. Продолжение

ПРИЛОЖЕНИЕ 3

# Текст прошивки

#include <SPI.h>

#include <MFRC522.h>

// константы подключения контактов SS и RST

#define RST\_PIN 9

#define SS\_PIN 10

// Инициализация MFRC522

MFRC522 mfrc522(SS\_PIN, RST\_PIN); // Create MFRC522 instance.

void setup()

{

Serial.begin(9600); // инициализация последовательного порта

SPI.begin(); // инициализация SPI

mfrc522.PCD\_Init(); // инициализация MFRC522

}

void loop()

{

if ( ! mfrc522.PICC\_IsNewCardPresent())

return;

// чтение карты

if ( ! mfrc522.PICC\_ReadCardSerial())

return;

// показать результат чтения UID и тип метки

Serial.print(F("Card UID:"));

dump\_byte\_array(mfrc522.uid.uidByte, mfrc522.uid.size);

Serial.println();

Serial.print(F("PICC type: "));

byte piccType = mfrc522.PICC\_GetType(mfrc522.uid.sak);

Serial.println(mfrc522.PICC\_GetTypeName(piccType));

delay(2000);

}

// Вывод результата чтения данных в HEX-виде

void dump\_byte\_array(byte \*buffer, byte bufferSize)

{

for (byte i = 0; i < bufferSize; i++)

{

Serial.print(buffer[i] < 0x10 ? " 0" : " ");

Serial.print(buffer[i], HEX);

}

Рис. 1. Текст прошивки микроконтроллера

ПРИЛОЖЕНИЕ 4

# Руководство пользователя

Данная информационная система предназначена для учёта рабочего времени сотрудников на предприятии.

В её функционал входит:

* фиксирование фактического времени прихода/ухода с рабочего места сотрудника посредством пропусков;
* ведение отсутствий: отгул, прогул, отпуск, командировки, больничные;
* ведение рабочих графиков;
* ведение данных сотрудников.

Установка

Для установки программного обеспечения на компьютер сотрудника необходимо воспользоваться автоматической установкой.

После удачной установки, на рабочем столе появится ярлык для запуска программы.

Настройка

Прежде чем начать работу с программой, необходимо установить систему пропусков в нужных местах, а так же зарегистрировать сотрудников в системе.

По окончанию данной процедуры, каждый пропуск будет хранить уникальный идентификационный номер, соответствующий сотруднику. По данному пропуску система будет фиксировать фактический приход и уход с того или иного места в офисе.

Описание функционала

С данным ПО может работать как HR Администратор, так и сотрудник.

HR Администратору доступен такой функционал, как:

* ведение персональных данных сотрудников;
* ведение временных данных;
* ведение рабочих графиков;
* согласование отсутствий;
* формирование отчётов.

Сотрудники в данной системе могут:

* редактировать свои персональные данные;
* создавать заявки на отсутствия.

Регистрация фактического времени системой осуществляется в автоматическом режиме.

Пользовательская инструкция для сотрудников

Для оформления отсутствия в системе необходимо:

1. войти в систему, используя свой Логин и Пароль;
2. перейти во вкладку «Оформление отсутствия» в главном меню программы;
3. в открывшемся окне необходимо выбрать день (или период), когда вы будете отсутствовать на рабочем месте, указать причину, а так же можете оставить комментарий, который увидит HR Администратор. При оформлении командировки необходимо будет указать дополнительные данные, такие как:

* место командировки;
* тип транспорта;
* номер билета;
* дата отъезда из города - дата прибытия в место командировки (согласно билету);
* дата отъезда из места командировки - дата прибытия в город (согласно билету);
* стоимость билета.

Для просмотра своих персональных данных необходимо перейти во вкладку «Мой профиль». Там же вы сможете отредактировать данные, нажав на иконку «Редактировать ПД»

Пользовательская инструкция для HR Администратора

Для согласования отсутствия на главной страницы программы нажмите на иконку «Уведомления». Она будет активной, в случае если есть новый запрос на отсутствие.

Для создания нового графика перейдите в раздел «Ведение графиков рабочего времени» в главном меню программы. В открывшемся окне необходимо будет внести данные для перерыва, однодневного рабочего дня, и указать тип рабочего графика: 5/2, 2/2 или настроить индивидуальный тип.

Для ведения временных данных перейдите в разделе Персонал в главном меню программы. Из списка работников выберите нужного, после чего в открывшемся окне выберете вкладку «Временные данные». В открывшейся форме, в зависимости от типа временного события, внесите изменения.