Министерство образования Республики Беларусь

Учреждение образования

БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ

ИНФОРМАТИКИ И РАДИОЭЛЕКТРОНИКИ

Специальность: «Программное обеспечение информационных технологий»

**КОНТРОЛЬНАЯ РАБОТА**

По курсу: «Технология разработки программного обеспечения»

Тема: «Создание базы данных на основе генерации кода»

Студент-заочник 2 курса

Группы № 581072

Швед Артем Олегович

Тел. 8-025-927-50-72

Проверила: Бакунова О.М.

Минск 2017г.

СОДЕРЖАНИЕ

[ВВЕДЕНИЕ 3](#_Toc472736322)

[1 ОПИСАНИЕ ПРЕДМЕТНОЙ ОБЛАСТИ 5](#_Toc472736323)

[1.1 Анализ требований к автоматизированной системе 5](#_Toc472736324)

[1.2 Описание и обоснование выбранного языка программирования 5](#_Toc472736325)

[2 СИСТЕМНОЕ ПРОЕКТИРОВАНИЕ ПРОГРАММНОГО СРЕДСТВА 8](#_Toc472736326)

[3 ЛОГИЧЕСКОЕ МОДЕЛИРОВАНИЕ 10](#_Toc472736327)

[3.1 Диаграмма вариантов использования 10](#_Toc472736328)

[3.2 Диаграмма классов 11](#_Toc472736329)

[3.3 Диаграмма последовательности 13](#_Toc472736330)

[3.4 Диаграмма коопераций 13](#_Toc472736331)

[3.5 Диаграмма развёртывания 14](#_Toc472736332)

[3.6 Диаграмма состояний 15](#_Toc472736333)

[3.7 Диаграмма компонентов 15](#_Toc472736334)

[СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ 17](#_Toc472736335)

[Генерация кода 18](#_Toc472736336)

# ВВЕДЕНИЕ

Автоматизация различных этапов технологических и бизнес процессов в настоящее время является привычным инструментом оптимизации деятельности предприятия, позволяющим существенно улучшить экономические и производственные показатели. Большинство универсальных разработок в этой области связано с автоматизацией управления финансовыми потоками компаний, в частности различные 1С-приложения.

Программные системы для управления денежными ресурсами развиты настолько, что их можно внедрить практически на любом предприятии. Однако кроме финансов, в любой организации существует множество задач, автоматизация которых может существенно повысить эффективность ее деятельности. И для подобных целей универсальных продуктов существенно меньше, т.к. необходимо учитывать специфику деятельности предприятий.

В результате либо требуется создание программной системы специально для заданного предприятия, либо серьезная настройка и адаптация уже готового продукта под его особенности.

Кроме того, зачастую возникают проблемы с внедрением приложения в уже существующую информационную систему, в частности возникает задача синхронизации данных с системой управления финансовыми потоками. Поэтому проблема автоматизации работы сервисного центра является довольно актуальной.

Объектом исследования является общество с ограниченной ответственностью «Автосервис», которое сервисным обслуживанием автомобилей. Целью работы является автоматизация работы сервисного центра.

Пользователю системы предоставляются определенные функциональные возможности, которые указываются в виде диаграммы вариантов использования, реализованной на основе синтаксиса языка UML. Построение программного обеспечения после предварительного моделирования аспектов его работы с помощью графических языков моделирования гораздо проще, чем создание приложения на основе исключительно текстовой документации.

Диаграммы вариантов использования описывают функциональное назначение системы или то, что система должна делать. Разработка диаграммы преследует следующие цели:

1. определить общие границы и контекст моделируемой предметной области;
2. сформулировать общие требования к функциональному поведению проектируемой системы;
3. разработать исходную концептуальную модель системы для ее последующей детализации в форме логических и физических моделей;
4. подготовить исходную документацию для взаимодействия разработчиков системы с ее заказчиками и пользователями.

Диаграмма вариантов использования представляет совокупность прецедентов и актеров. Они описывают систему с точки зрения возможностей ее использования. Наиболее часто данный вид диаграмм используют при организации и моделировании поведения системы. Она позволяет создать список операций, которые должна выполнять система. Часто эти диаграммы называют диаграммами функций т.к. на их основе определяется список требований к системе и множество, выполняемых системой функций. На диаграмм6е отображаются объекты предметной области с выполняемыми ими задачами.

# 1 ОПИСАНИЕ ПРЕДМЕТНОЙ ОБЛАСТИ

# 1.1 Анализ требований к автоматизированной системе

Кратко рассмотрим предметную область, задачи которой подлежат автоматизации в ходе выполнения курсового проекта.

Клиент приезжает на сервис, сначало через форму заводится его автомобиль потом через другую форму заводится сам клиент.

Есть документ заказ наряд который оформляется на клиента и на автомобиль, в форме нужно сделать текстовые причина обращения

и рекомендации.

Возможность выбора детали с наличия по складу или указать как деталь клиента.

три запроса в виде отчета:

1) отдача по рекламе

2) Выработка мастеров

3) и деньги в кассе

Дополнительные требования:

1. Нужна таблица в которой будут хранится реквизиты автосервиса;

2. Соответственно нужно чтобы они выводились в заказ наряде и бланке заказа;

3. Нужна авторизация в программе при запуске;

4. Отчеты с возможностью печати;

5. Нужна возможность печати прайс листа;

6. Заказ наряд новый формировался из отдельной формы;

7. В отчеты добавить возможность выбора периода;

8. Возможность сортировки и поиска.

Вывод бланка заказа и бланка наряда на работы, прайс-листов по услугам и запчастям.

# 1.2 Описание и обоснование выбранного языка программирования

Для генерации кода был выбран язык программирования Java.

Java - объектно-ориентированный язык программирования, разрабатываемый компанией Sun Microsystems с 1991 года и официально выпущенный 23 мая 1995 года. Изначально новый язык программирования назывался Oak (James Gosling) и разрабатывался для бытовой электроники, но впоследствии был переименован в Java и стал использоваться для написания апплетов, приложений и серверного программного обеспечения

Отличительной особенностью Java в сравнении с другими языками программирования общего назначения является обеспечение высокой продуктивности программирования, нежели производительность работы приложения или эффективность использования им памяти.

В Java используются практически идентичные соглашения для объявления переменных, передачи параметров, операторов и для управления потоком выполнением кода. В Java добавлены все хорошие черты C++.

Три ключевых элемента объединились в технологии языка Java:

1) Java предоставляет для широкого использования свои апплеты (applets) — небольшие, надежные, динамичные, не зависящие от платформы активные сетевые приложения, встраиваемые в страницы Web. Апплеты Java могут настраиваться и распространяться потребителям с такой же легкостью, как любые документы HTML.

2) Java высвобождает мощь объектно-ориентированной разработки приложений, сочетая простой и знакомый синтаксис с надежной и удобной в работе средой разработки. Это позволяет широкому кругу программистов быстро создавать новые программы и новые апплеты.

3) Java предоставляет программисту богатый набор классов объектов для ясного абстрагирования многих системных функций, используемых при работе с окнами, сетью и для ввода-вывода. Ключевая черта этих классов заключается в том, что они обеспечивают создание независимых от используемой платформы абстракций для широкого спектра системных интерфейсов.

Огромное преимущество Java заключается в том, что на этом языке можно создавать приложения, способные работать на различных платформах. К сети Internet подключены компьютеры самых разных типов - Pentium PC, Macintosh, рабочие станции Sun и так далее. Даже в рамках компьютеров, созданных на базе процессоров Intel, существует несколько платформ, например, Microsoft Windows версии 3.1, Windows 95, Windows NT, OS/2, Solaris, различные разновидности операционной системы UNIX с графической оболочкой X­Windows. Между тем, создавая сервер Web в сети Internet, хотелось бы, чтобы им могло пользоваться как можно большее число людей. В этом случае выручат приложения Java, предназначенные для работы на различных платформах и не зависящие от конкретного типа процессора и операционной системы.

Программы, составленные на языке программирования Java, можно разделить по своему назначению на две большие группы.

К первой группе относятся приложения Java, предназначенные для автономной работы под управлением специальной интерпретирующей машины Java. Реализации этой машины созданы для всех основных компьютерных платформ.

Вторая группа - это так называемые аплеты (applets). Аплеты представляют собой разновидность приложений Java, которые интерпретируются виртуальной машиной Java, встроенной практически во все современные браузеры.

Приложения, относящиеся к первой группе - это обычные автономные программы. Так как они не содержат машинного кода и работают под управлением специального интерпретатора, их производительность заметно ниже, чем у обычных программ, составленных, например, на языке программирования C++. Программы Java без перетрансляции способны работать на любой платформе, что само по себе имеет большое значение в плане разработок для Internet.

Аплеты Java встраиваются в документы HTML, хранящиеся на сервере Web. С помощью аплетов вы можете сделать страницы сервера Web динамичными и интерактивными. Аплеты позволяют выполнять сложную локальную обработку данных, полученных от сервера Web или введенных пользователем с клавиатуры. Из соображений безопасности аплеты (в отличие от обычных приложений Java) не имеют никакого доступа к файловой системе локального компьютера. Все данные для обработки они могут получить только от сервера Web. Более сложную обработку данных можно выполнять, организовав взаимодействие между аплетами и расширениями сервера Web - приложениями CGI и ISAPI.

Для повышения производительности приложений Java в современных браузерах используется компиляция "на лету"- Just-In-Time compilation (JIT). При первой загрузке аплета его код транслируется в обычную исполнимую программу, которая сохраняется на диске и запускается. В результате общая скорость выполнения аплета Java увеличивается в несколько раз.

Язык Java является объектно-ориентированным и поставляется с достаточно объемной библиотекой классов. Так же как и библиотеки классов систем разработки приложений на языке C++, библиотеки классов Java значительно упрощают разработку приложений, представляя в распоряжение программиста мощные средства решения распространенных задач. Поэтому программист может больше внимания уделить решению прикладных задач.

Язык Java специально ориентирован на самые передовые технологии, связанные с сетью Internet. Растущая популярность Internet и, в особенности, серверов Web, создает для программистов новые возможности для реализации своих способностей.

# 2 СИСТЕМНОЕ ПРОЕКТИРОВАНИЕ ПРОГРАММНОГО СРЕДСТВА

IDEF0 — методология функционального моделирования. С помощью наглядного графического языка IDEF0, изучаемая система предстает перед разработчиками и аналитиками в виде набора взаимосвязанных функций (функциональных блоков — в терминах IDEF0). Как правило, моделирование средствами IDEF0 является первым этапом изучения любой системы.

IDEF1X является методом для разработки реляционных баз данных и использует условный синтаксис, специально разработанный для удобного построения концептуальной схемы.

Проведем формализованное описание информационной системы в терминах IDEF0 моделирования.

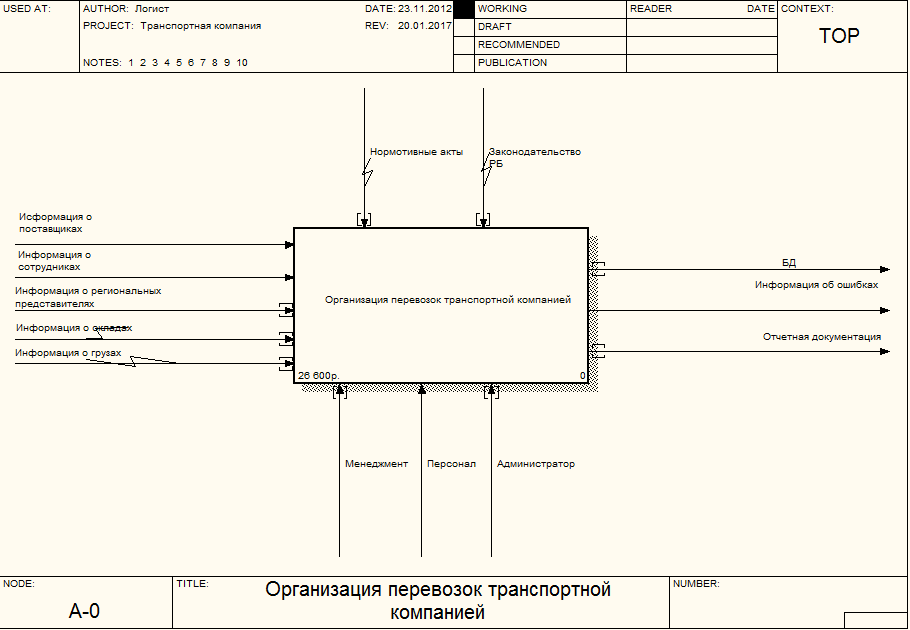


Рисунок 2.1 − Функциональная модель работы предприятия

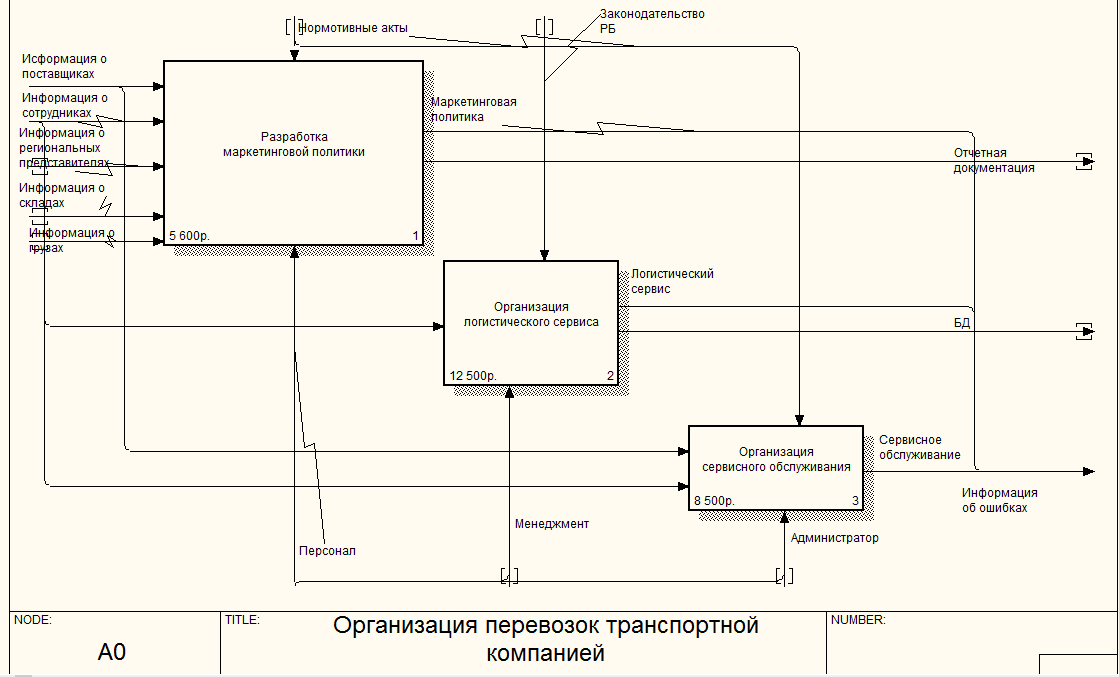


Рисунок 2.2 — Декомпозиция модели

На данный момент полностью автоматизированным является процесс управления финансовыми потоками предприятия на основе 1С Бухгалтерия 7.7, а также процесс учета имеющихся запчастей и товаров и их заказа на основе 1С Торговля и склад 7.7. Таким образом, автоматизированы практически все бизнес-процессы предприятия, кроме сервисного обслуживания. Передача информации между отделами производится посредством локальной вычислительной сети, в которую объединены все компьютеры предприятия. Каждый из компьютеров оснащен сетевым оборудованием, установленной операционной системой Windows XP Professional, стандартным набором офисных программ и приложениями 1С Бухгалтерия 7.7, 1С Торговля и склад 7.7.

Вся информация о продажах, закупках, заказах и поставках запчастей и товаров автоматически заносится в единую базу и доступна с любого компьютера. Различные формы: отчета, заказа, договора купли-продажи или сервисного обслуживания формируются системой также -автоматически. Поэтому можно считать, что перечисленные ниже бизнес-процессы автоматизированы:

* продажа автомобилей;
* продажа запчастей;
* заказ автомобилей;
* заказ запчастей;
* прием (от поставщика);
* прием от покупателя (расторжения договора купли-продажи);
* прием оплаты сервисного обслуживания (ремонта);
* различные бухгалтерские задачи (начисление заработной платы, отчисления в бюджет, налоги и пр.);

# 3 ЛОГИЧЕСКОЕ МОДЕЛИРОВАНИЕ

Основное назначение логического представления состоит в анализе структурных и функциональных отношений между элементами модели системы. Различные элементы логического представления, такие как классы, ассоциации, состояния, сообщения, не существуют материально или физически. Они лишь отражают наше понимание структуры физической системы или аспекты ее поведения.

Rational Rose представляет собой CASE средство проектирования и разработки информационных систем и программного обеспечения для управления предприятиями.

Как и другие  [*CASE средства*](http://www.kpms.ru/Automatization/CASE_tools.htm) ([*ARIS*](http://www.kpms.ru/Automatization/ARIS.htm), *[BPwin](http://www.kpms.ru/Automatization/BPwin.htm)*, *[ERwin](http://www.kpms.ru/Automatization/ERwin.htm)*) его можно применять для анализа и моделирования бизнес процессов. Первая версия этого продукта была выпущена компанией Rational Software . В дальнейшем Rational Rose был куплен IBM.

Принципиальное отличие Rational Rose от других средств заключается в объектно-ориентированном подходе. Графические модели, создаваемые с помощью этого средства, основаны на объектно-ориентированных принципах и языке [*UML*](http://www.kpms.ru/Automatization/BPM.htm#BPM_UML) (Unified Modeling Language). Инструменты моделирования Rational Rose позволяют разработчикам создавать целостную архитектуру процессов предприятия, сохраняя все взаимосвязи и управляющие воздействия между различными уровнями иерархии.

# 3.1 Диаграмма вариантов использования

Диаграммы вариантов использования описывают функциональное назначение системы или то, что система должна делать. Разработка диаграммы преследует следующие цели:

1. определить общие границы и контекст моделируемой предметной области;
2. сформулировать общие требования к функциональному поведению проектируемой системы;
3. разработать исходную концептуальную модель системы для ее последующей детализации в форме логических и физических моделей;
4. подготовить исходную документацию для взаимодействия разработчиков системы с ее заказчиками и пользователями.

Диаграмма вариантов использования представляет совокупность прецедентов и актеров. Они описывают систему с точки зрения возможностей ее использования. Наиболее часто данный вид диаграмм используют при организации и моделировании поведения системы. Она позволяет создать список операций, которые должна выполнять система. Часто эти диаграммы называют диаграммами функций т.к. на их основе определяется список требований к системе и множество, выполняемых системой функций. На диаграмм6е отображаются объекты предметной области с выполняемыми ими задачами.

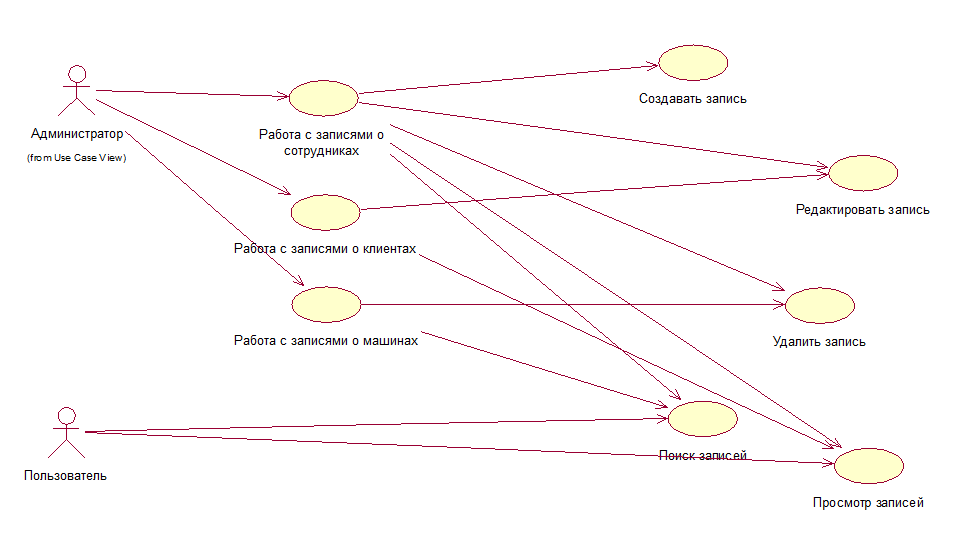


Рисунок 3.1 – Диаграмма вариантов использования

# 3.2 Диаграмма классов

Диаграмма классов — диаграмма, демонстрирующая классы системы, их атрибуты, методы и взаимосвязи между ними. Входит в UML.

Существует два вида:

* статический вид диаграммы рассматривает логические взаимосвязи классов между собой;
* аналитический вид диаграммы рассматривает общий вид и взаимосвязи классов, входящих в систему.

На диаграмме отображаются следующие классы и атрибуты:

Company – сведения о компании. Класс содержит следующие атрибуты:

* NameComp – название логистической компании;
* DateOf – дата закрытия заказа;
* SizeGruzz – размеры перевозимых грузов;
* PId – штрих код груза.

Transporter – транспортируемый груз. Класс содержит следующие атрибуты:

* CcrId – номер груза;
* SityName – страна-заказчик;
* Customer – клиент;
* DateDost – дата доставки;
* DateZagr – дата загрузки.

Putevka – путевой лист. Класс содержит следующие атрибуты:

* Voditel – ФИО водителя;
* StranaOtpr – страна-отправитель;
* StranaPriema – страна-заказчика;
* Gruz – груз;
* Ves - вес.

Transporting – транспортная накладная. Класс содержит следующие атрибуты:

* Zakazchik – название заказчика;
* DiD – номер заказчика;
* Kolichestvo – количество груза;
* Nakladnaya – накладная;
* Symma – сумма груза

Cars – машины компании. Класс содержит следующие атрибуты:

* Nomer – рег.номер машины;
* Gruzopodem –грузоподъёмность машины;
* Color – цвет;
* Marka - марка.

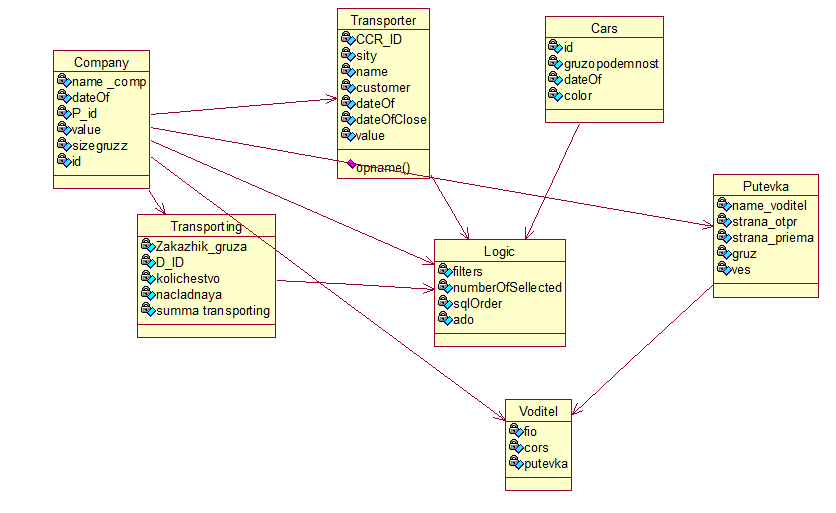


Рисунок 3.2 – Диаграмма классов

# 3.3 Диаграмма последовательности

Диаграмма последовательности — диаграмма, на которой показано взаимодействие объектов, упорядоченное по времени, с отражением продолжительности обработки и последовательности их проявления. Используется в языке UML.

На данной диаграмме объекты располагаются слева направо.

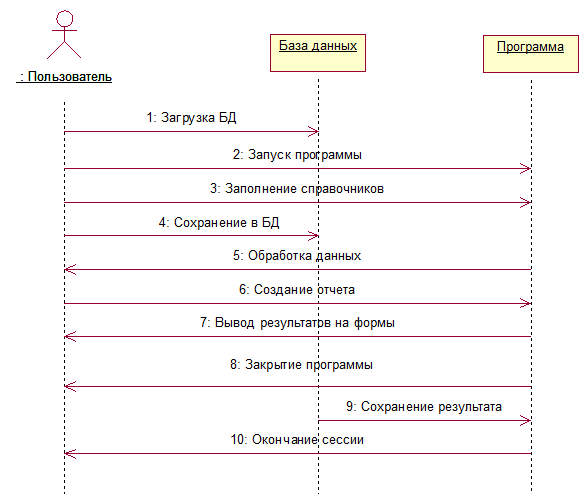


Рисунок 3.3 – Диаграмма последовательности

# 3.4 Диаграмма коопераций

Главная особенность диаграммы кооперации заключается в возможности графически представить не только последовательность взаимодействия, но и все структурные отношения между объектами, участвующими в этом взаимодействии.

В отличие от диаграммы последовательности, на диаграмме кооперации изображаются только отношения между объектами, играющими определенные роли во взаимодействии. На этой диаграмме не указывается время в виде отдельного измерения. Поэтому последовательность взаимодействий и параллельных потоков может быть определена с помощью порядковых номеров. Следовательно, если необходимо явно специфицировать взаимосвязи между объектами в реальном времени, лучше это делать на диаграмме последовательности.

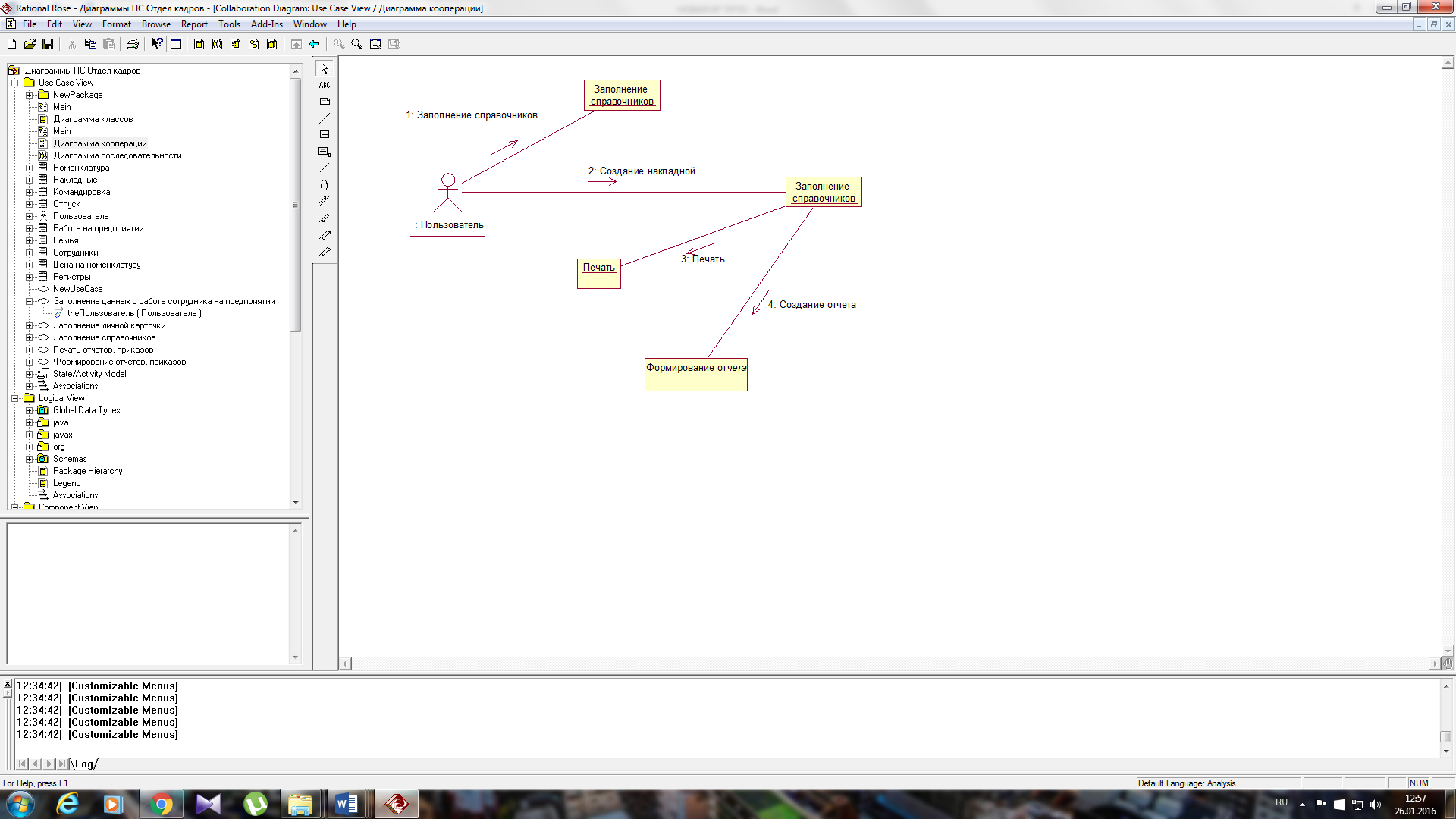


Рисунок 3.4 – Диаграмма коопераций

# 3.5 Диаграмма развёртывания

Диаграмма развёртывания в UML моделирует физическое развертывание артефактов на узлах. Например, чтобы описать веб-сайт диаграмма развертывания должна показывать, какие аппаратные компоненты («узлы») существуют (например, веб-сервер, сервер базы данных, сервер приложения), какие программные компоненты («артефакты») работают на каждом узле (например, веб-приложение, база данных), и как различные части этого комплекса соединяются друг с другом (например, JDBC, REST, RMI).

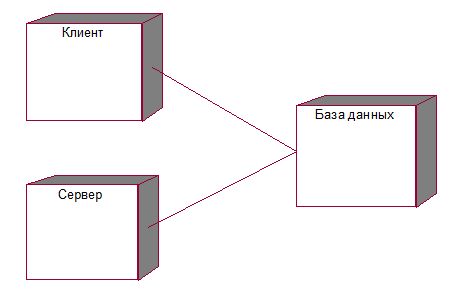


Рисунок 3.5 – Диаграмма развёртывания

# 3.6 Диаграмма состояний

Диаграмма состояний — это, по существу, диаграмма состояний из теории автоматов со стандартизированными условными обозначениями, которая может определять множество систем от компьютерных программ до бизнес-процессов.

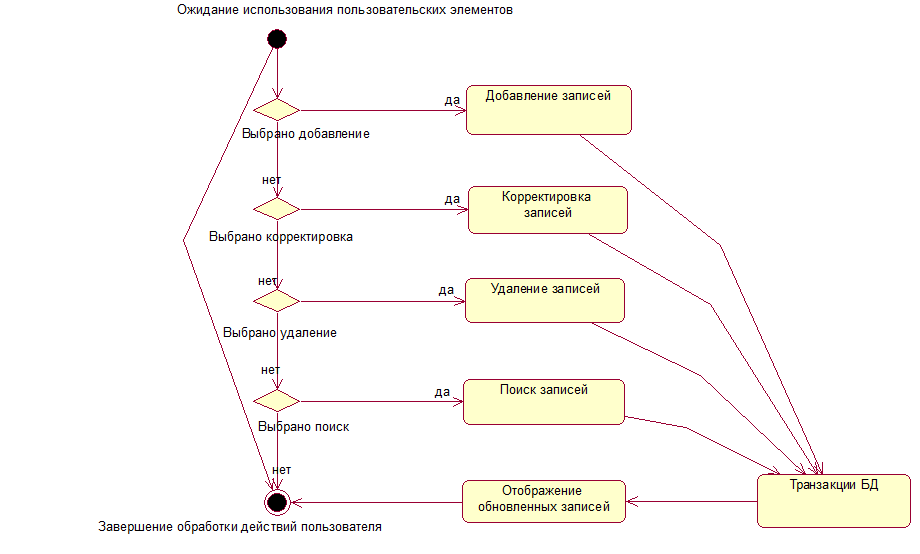


Рисунок 3.6 – Диаграмма состояний

# 3.7 Диаграмма компонентов

Диаграмма компонентов — статическая структурная диаграмма, показывает разбиение программной системы на структурные компоненты и связи (зависимости) между компонентами. В качестве физических компонентов могут выступать файлы, библиотеки, модули, исполняемые файлы, пакеты и т. п.

Компоненты связываются через зависимости, когда соединяется требуемый интерфейс одного компонента с имеющимся интерфейсом другого компонента. Таким образом иллюстрируются отношения клиент-источник между двумя компонентами.

Зависимость показывает, что один компонент предоставляет сервис, необходимый другому компоненту. Зависимость изображается стрелкой от интерфейса или порта клиента к импортируемому интерфейсу.

Когда диаграмма компонентов используется, чтобы показать внутреннюю структуру компонентов, предоставляемый и требуемый интерфейсы составного компонента могут делегироваться в соответствующие интерфейсы внутренних компонентов.

Делегация показывается связь внешнего контракта компонента с внутренней реализацией этого поведения внутренними компонентами.

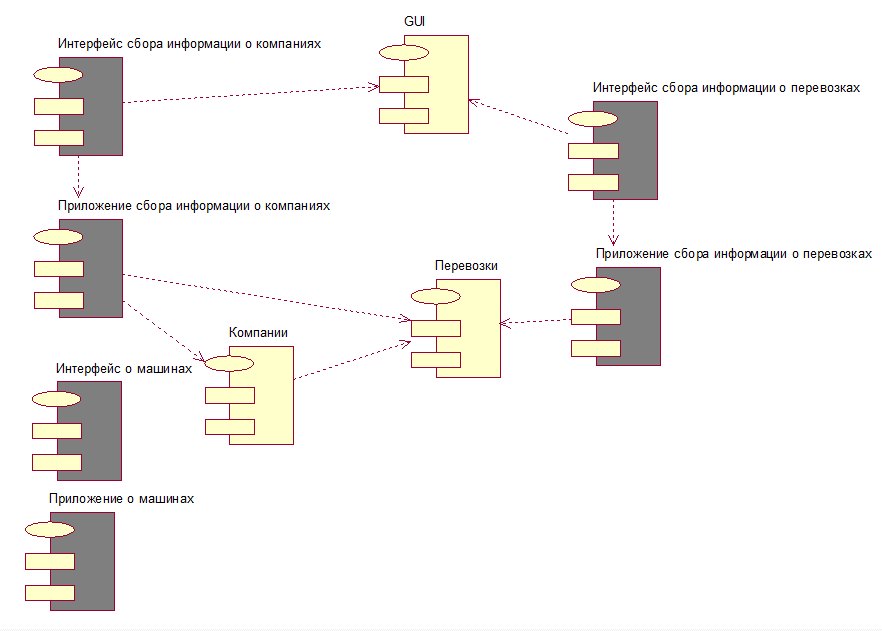


Рисунок 3.7 – Диаграмма компонентов

# СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. ГОСТ 2.105-95 ЕСКД. Введ. 1996–07–01. – М.: Изд-во стандартов, 1996.
2. ГОСТ 19.104–78 – Единая система программной документации. Основные надписи.
3. Маклаков С.В. BPwin и ERwin: CASE - средства для разработки информационных систем.
4. Федотов Д.Э., Семенов Ю.Д., Чижик К.Н. Практикум для высших учебных заведений. CASE-технологии.-157с.
5. Интернет обучение [Электронный ресурс] / Котляр Д.С. Использование CASE-средства ERwin для автоматизации проектирования и разработки базы данных – Режим доступа: http://royallib.com/read/ bezopasnost – Дата доступа: 15.12.2016.
6. Интернет обучение [Электронный ресурс] / Информационные системы и технологии – Режим доступа: <http://www.narfu.ru> – Дата доступа: 19.12.2016.
7. Интернет обучение [Электронный ресурс] / Нормализация структурны данных – Режим доступа: http://infostart.ru/public/269803/ – Дата доступа: 15.01.2016.
8. Интернет обучение [Электронный ресурс] / Программа компьютерного моделирования BpWin – Режим доступа: http://bourabai.kz/cm/bpwin.htm– Дата доступа: 11.11.2016.
9. Леоненков. «Самоучитель UML».
10. 2. Б. Хичков SYBASE: Настольная книга администратора. Лори, 2000 г. 448 с.
11. 3. Р. Мюллер. Базы данных и UML: Проектирование.– Лори, 2002г. 432 с.
12. 4. Фельдман С.К. Система программирования Java без секретов: Как создать безопасное приложение с "нуля". – Новый издательский дом" , 2005 г. , 347 с.
13. 5. Дейтел П.Дж., Дейтел Х.М. Как програмировать на Java. Книга 2. Файлы, сети, базы данных. – "Бином" · 2005 г., 672 с.
14. Интернет обучение [Электронный ресурс] / Основные методологии обследования организаций. Стандарт IDEF0. – Режим доступа: http://www.cfin.ru/vernikov/idef/idef0.shtml
15. Интернет обучение [Электронный ресурс] / Основы методологии IDEF1X. – Режим доступа: http://www.cfin.ru/vernikov/idef/idef0.shtml

# Генерация кода

# Приложение А

