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»

На тему: «Детский сад»

Студент-заочник 2 курса

Группы № 581072

ФИО: Руденко Максим

А

Адрес: г. Минск

ул. Улица

Тел. +375 29

Минск, 2017

**СОДЕРЖАНИЕ**

[ВВЕДЕНИЕ 3](#_Toc473816083)

[1 АНАЛИЗ ПРЕДМЕТНОЙ ОБЛАСТИ 5](#_Toc473816084)

[1.1 Анализ требований к автоматизированной системе 6](#_Toc473816085)

[2 СИСТЕМНОЕ ПРОЕКТИРОВАНИЕ ПРОГРАММНОГО СРЕДСТВА 7](#_Toc473816086)

[3 ЛОГИЧЕСКОЕ МОДЕЛИРОВАНИЕ 10](#_Toc473816087)

[3.1 Диаграмма вариантов использования 10](#_Toc473816088)

[3.2 Диаграмма классов 11](#_Toc473816089)

[3.3 Диаграмма деятельности 12](#_Toc473816090)

[3.4 Диаграмма кооперации 14](#_Toc473816091)

[3.5 Диаграмма последовательности 15](#_Toc473816092)

[3.6 Диаграмма развертывания 16](#_Toc473816093)

[3.7 Диаграмма компонентов 17](#_Toc473816094)

[4 ГЕНЕРАЦИЯ КОДА 18](#_Toc473816095)

[ЗАКЛЮЧЕНИЕ 25](#_Toc473816096)

[СПИСОК ИСПОЛЬЗОВАНЫХ ИСТОЧНИКОВ 26](#_Toc473816097)

## ВВЕДЕНИЕ

Как тип педагогического учреждения первый детский сад был организован в начале XIX века в Нью-Ланарке (Шотландия) социалистом-утопистом Р. Оуэном — так называемая «школа для маленьких детей».

Само название — «Детский сад» пришло из Германии и было придумано в 1837 году педагогом Фридрихом Вильгельмом Августом Фрёбелем. Им же было создано учреждение для игр и занятий детей младшего возраста в городе Бад-Бланкенбурге. Хотя данное учреждение существовало всего около двух лет. Название «Детский сад» он придумал из соображений, что дети — цветы жизни, требующие умелого и тщательного ухода и выращивать их должны садовницы.

В России первые детские сады были открыты в 60-х гг. XIX века. Они были частными и дорогостоящими, поэтому были не доступны для простых людей. Впервые упоминается о детских садах в 1859 г. (г. Гельсингфорс, ныне — столица Финляндии Хельсинки). В Москве же первый детский сад был открыт только в 1866 г. при пансионе девиц Герке.

В период с 1866 по 1870 г. было открыто несколько платных детских садов частными лицами в Иркутске, Воронеже, Москве, Смоленске, Тбилиси, Санкт-Петербурге. В 1868—1869 годах в Москве были открыты четыре платных детских сада, принадлежавших Мамонтовой, Левенштерн, Соловьевой и Римской-Корсаковой. В 1893 году в Москве было 7 платных частных детских садов для детей обоего пола (35 девочек и 21 мальчик). Все они находились при учебных заведениях и представляли собой подготовительные школы для детей самого младшего возраста.

В данные детские сады принимались дети от 3 до 8 лет. Там с ними занимались педагоги, дети играли в подвижные игры. Кроме этого Симонович начала издавать журнал «Детский сад», в котором рассказывалось о дошкольном образовании.

Первый бесплатный детский сад был открыт в России в 1866 г. Это было благотворительное учреждение при «Обществе дешевых квартир для детей работниц Петербурга». Там имелась швейная мастерская для пошива детского белья, кухня, прачечная, школа для детей. Более взрослые дети обучались священному писанию, молитвам, проводились различные ручные работы, такие как плетение, рисование, вырезание и многое другое. Но в виду отсутствия средств к существованию, бесплатный детский сад был вскоре закрыт.

Система дошкольных образовательных учреждений активно развивалась, и через три десятилетия в России появилось несколько десятков детских садов: платных и бесплатных, для дворян и интеллигенции, рабочих, а также для приютов для сирот.

В это время начали организовываться образовательные курсы для воспитателей, проводились лекции и «тренинги», выпускалась соответствующая литература.

20 ноября 1917 года была принята официальная «Декларация по дошкольному воспитанию». Этот документ гарантировал бесплатное образование и воспитание детей дошкольного возраста.

Первый педагогический факультет с дошкольным отделением был открыт в 1918 году в Московском государственном университете. Первая «Программа работы детского сада» вышла в 1934 году, а в 1938 году были опубликованы «Устав детского сада», определявший задачи работы, структуру и особенности функционирования дошкольных учреждений, и «Руководство для воспитателей детского сада», содержавшее методические указания по разделам работы с детьми.

К середине ХХ века более двух миллионов детей уже посещали детские сады. В послевоенное время в СССР появились первые ясли, где родители могли оставлять малышей, начиная с двух месяцев. В начале 60-х годов был разработан единый для всех учреждений дошкольного образования документ, определяющий их программу работы.

В начале XXI века в России насчитывается более 45 тыс. детских дошкольных учреждений. Современная система дошкольного образования состоит из яслей, детских садов, групп кратковременного пребывания детей, центров дошкольного образования.

Темой данной контрольной работы является разработка базы данных детского сада. Эта тема является весьма актуальной. Согласование работы детского сада с родителями воспитанников требует большого количества времени и человеческих ресурсов, причем, зачастую, не только работников сада, но и внешних. Разрабатываемое программное средство должно значительно упростить его деятельность, предоставив полный функциональный комплекс для упрощения процедуры зачисления детей в учреждение.

## 1 АНАЛИЗ ПРЕДМЕТНОЙ ОБЛАСТИ

Дошкольное образование — воспитание, обучение и развитие, а также присмотр, уход и оздоровление детей дошкольного возраста от 2 месяцев до 7 лет. Дошкольное образование осуществляется, как правило, в учреждениях дошкольного образования, учреждениях общего образования (предшкола), учреждениях дополнительного образования детей (центры и объединения раннего развития ребёнка), но может осуществляться и дома в семье.

Учреждение дошкольного образования – учреждение образования, которое реализует образовательную программу дошкольного образования, программу воспитания и защиты прав и законных интересов детей, находящихся в социально опасном положении, образовательную программу специального образования на уровне дошкольного образования, образовательную программу специального образования на уровне дошкольного образования для лиц с интеллектуальной недостаточностью.

Учреждения дошкольного образования могут быть следующих видов:

* ясли;
* ясли-сад;
* детский сад;
* санаторный ясли-сад;
* санаторный детский сад;
* дошкольный центр развития ребенка.

Ясли – учреждение дошкольного образования, в котором воспитанники раннего возраста получают дошкольное или специальное образование и могут получать оздоровление.

Ясли-сад – учреждение дошкольного образования, в котором воспитанники раннего и дошкольного возраста получают дошкольное или специальное образование и могут получать оздоровление.

Детский сад – учреждение дошкольного образования, в котором воспитанники дошкольного возраста получают дошкольное или специальное образование и могут получать оздоровление.

Санаторный ясли-сад – учреждение дошкольного образования, в котором воспитанники раннего и дошкольного возраста получают дошкольное или специальное образование и оздоровление.

Санаторный детский сад – учреждение дошкольного образования, в котором воспитанники дошкольного возраста получают дошкольное или специальное образование и оздоровление.

Дошкольный центр развития ребенка – учреждение дошкольного образования, в котором воспитанники раннего и дошкольного возраста получают дошкольное или специальное образование и оздоровление, а также развивают творческие способности.

### 1.1 Анализ требований к автоматизированной системе

В результате изучения особенностей разрабатываемой базы данных для программного средства и ознакомления с требованиями, предъявляемыми к ним, было составлено следующее функциональное назначение.

Программное средство детского сада должно обеспечивать исполнение процесса приема воспитанников в учреждение дошкольного образования на всех его этапах: постановку на учет, обеспечение направлением в детский сад, прием воспитанника в детском саду. Постановка на учёт детей, нуждающихся в определении в учреждение образования для получения дошкольного образования, осуществляется путем внесения сведений о ребенке в списки нуждающихся в определении в учреждение образования для получения дошкольного образования. Также программа должна предоставлять информацию о детском саде, проводимых административных процедурах, организации работы с воспитанниками и их родителями, реализуемых проектах, службах учреждения, организации учебно-воспитательного процесса, графиках приема граждан и посетителей, администрации, специалистах. Программное средство должно содержать все необходимые нормативно-правовые документы, регламентирующие постановку на учёт, приём, перевод и отчисление воспитанников в учреждении дошкольного образования.

## 2 СИСТЕМНОЕ ПРОЕКТИРОВАНИЕ ПРОГРАММНОГО СРЕДСТВА

Описание системы с помощью IDEF0 называется функциональной моделью. Функциональная модель предназначена для описания существующих бизнес-процессов, в котором используются как естественный, так и графический языки. Для передачи информации о конкретной системе источником графического языка является сама методология IDEF0.

Методология IDEF0 предписывает построение иерархической системы диаграмм - единичных описаний фрагментов системы. Сначала проводится описание системы в целом и ее взаимодействия с окружающим миром (контекстная диаграмма), после чего проводится функциональная декомпозиция - система разбивается на подсистемы, и каждая подсистема описывается отдельно (диаграммы декомпозиции). Затем каждая подсистема разбивается на более мелкие и так далее до достижения нужной степени подробности.

Каждая IDEF0-диаграмм а содержит блоки и дуги. Блоки изображают функции моделируемой системы. Дуги связывают блоки вместе и отображают взаимодействия и взаимосвязи между ними.

Функциональные блоки (работы) на диаграммах изображаются прямоугольниками, означающими поименованные процессы, функции или задачи, которые происходят в течение определенного времени и имеют распознаваемые результаты. Имя работы должно быть выражено отглагольным существительным, обозначающим действие. IDEF0 требует, чтобы в диаграмме было не менее трех и не более шести блоков. Эти ограничения поддерживают сложность диаграмм и модели на уровне, доступном для чтения, понимания и использования.

Каждая сторона блока имеет особое, вполне определенное назначение. Левая сторона блока предназначена для входов, верхняя - для управления, правая - для выходов, нижняя - для механизмов. Такое обозначение отражает определенные системные принципы: входы преобразуются в выходы управление ограничивает или предписывает условия выполнения преобразований, механизмы показывают, что и как выполняет функция.

Блоки в IDEF0 размещаются по степени важности, как ее понимает автор диаграммы. Этот относительный порядок называется доминированием. Доминирование понимается как влияние, которое один блок оказывает на другие блоки диаграммы. Наиболее доминирующий блок обычно размещается в верхнем левом углу диаграммы, а наименее доминирующий - в правом углу.

Расположение блоков на странице отражает авторское определение доминирования. Таким образом, топология диаграммы показывает, какие функции оказывают большее влияние на остальные. Чтобы подчеркнуть это, аналитик может перенумеровать блоки в соответствии с порядком их доминирования. Порядок доминирования может обозначаться цифрой, размещенной в правом нижнем углу каждого прямоугольника: 1 будет указывать на наибольшее доминирование, 2 - на следующее и т. д.

Взаимодействие работ с внешним миром и между собой описывается в виде стрелок, изображаемых одинарными линиями со стрелками на концах. Стрелки представляют собой некую информацию и именуются существительными. Различают пять типов стрелок: вход, управление, выход, механизм, вызов.

В методологии IDEF0 требуется только пять типов взаимодействий между блоками для описания их отношений: управление, вход, обратная связь по управлению, обратная связь по входу, выход-механизм. Связи по управлению и входу являются простейшими, поскольку они отражают прямые воздействия, которые интуитивно понятны и очень просты.

IDEF1 — одна из методологий семейства IDEF. Применяется для построения информационной модели, которая представляет структуру информации, необходимой для поддержки функций производственной системы или среды. В настоящее время на основе совершенствования методологии IDEF1 создана её новая версия — методология IDEF1X. IDEF1X разработана с учетом таких требований, как простота изучения и возможность автоматизации. IDEF1X–диаграммы используются рядом распространённых CASE–средств (в частности, ERwin, Design/IDEF).

Хотя терминология IDEF1X практически совпадает с терминологией IDEF1, существует ряд фундаментальных отличий в теоретических концепциях этих методологий. Сущность в IDEF1X описывает собой совокупность или набор экземпляров похожих по свойствам, но однозначно отличаемых друх от друга по одному или нескольким признакам. Каждый экземпляр является реализацией сущности. Таким образом, сущность в IDEF1X описывает конкретный набор экземпляров реального мира, в отличие от сущности в IDEF1, которая представляет собой абстрактный набор информационных отображений реального мира. Связи в IDEF1X представляют собой ссылки, соединения и ассоциации между сущностями. Связи это суть глаголы, которые показывают, как соотносятся сущности между собой.

Контекстная диаграмма модели показана на рисунке 2.1. Декомпозиция контекстной диаграммы функциональной модели показана на рис. 2.2. Для построения модели использовался продукт BPwin 4.0 фирмы Computer Associates.

Субъектом моделирования является программное средство детского сада, цель – описать функциональность работы программного средства, точка зрения – пользователь, администратор.

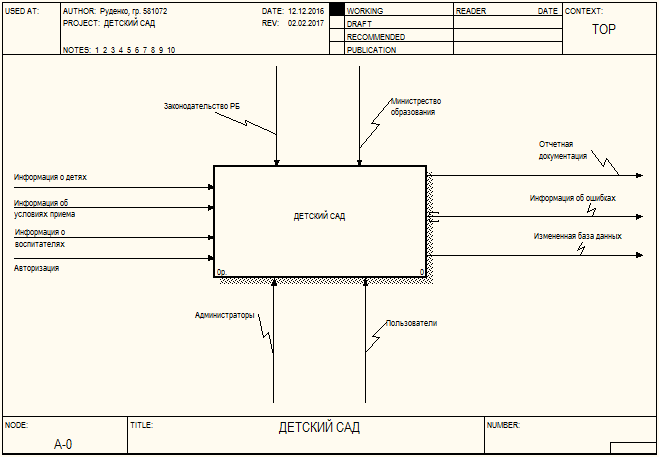


Рисунок 2.1– Контекстная диаграмма модели

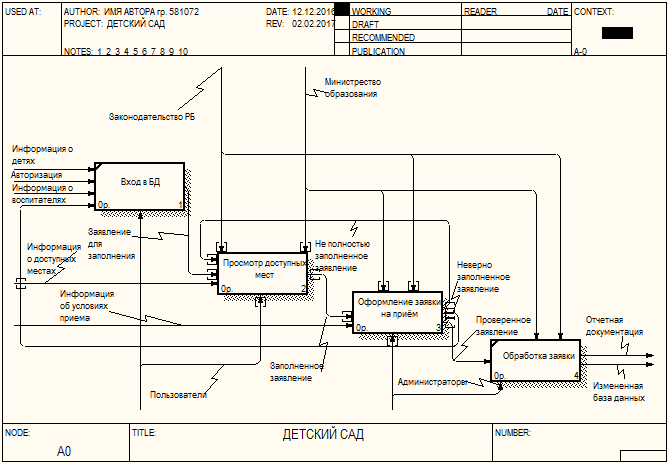


Рисунок 2.2 – Декомпозиция контекстной диаграммы

функциональной модели

## 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.

### 3.1 Диаграмма вариантов использования

Модель вариантов использования предназначается для определения требований к системе. Она включает в себя актеров, варианты использования и связи между ними. Для отображения этой модели язык UML предлагает использовать диаграммы Use Case (вариант использования) совместно с моделями State Diagram (диаграммы состояний) и Activity Diagram (диаграммы деятельности/активности). Последние используются для конкретизации вариантов использования системы

Разработка диаграммы вариантов использования преследует цели:

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

Диаграмма вариантов использования представлена на рисунке 3.1.1.

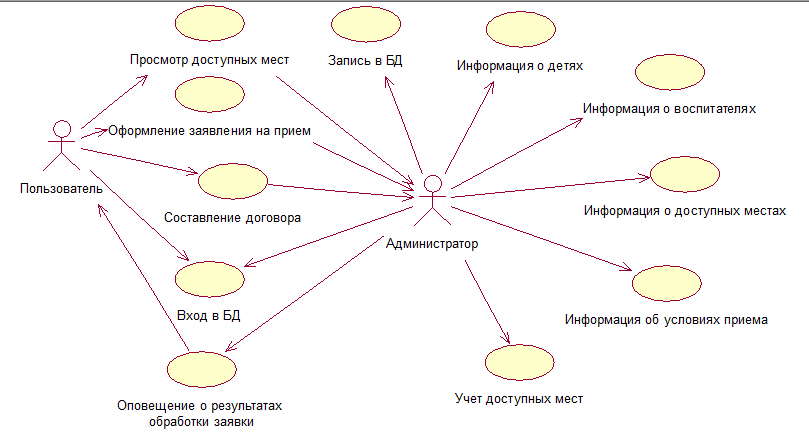


Рисунок 3.1.1 – Диаграмма вариантов использования

### 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.

Существуют разные точки зрения на построение диаграмм классов в зависимости от целей их применения:

* концептуальная точка зрения — диаграмма классов описывает модель предметной области, в ней присутствуют только классы прикладных объектов;
* точка зрения спецификации — диаграмма классов применяется при проектировании информационных систем;
* точка зрения реализации — диаграмма классов содержит классы, используемые непосредственно в программном коде (при использовании объектно-ориентированных языков программирования).

Диаграмма классов представлена на рисунке 3.2.1.

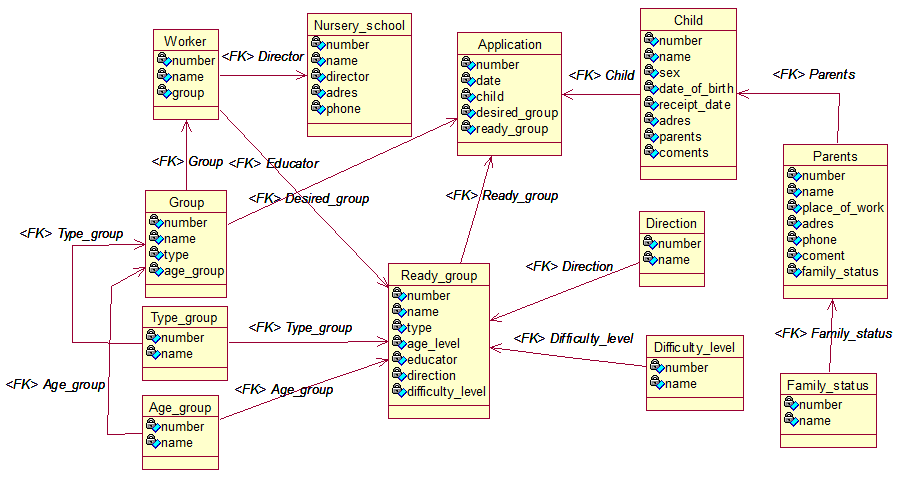


Рисунок 3.2.1 – Диаграмма классов

### 3.3 Диаграмма деятельности

Диаграмма деятельности — UML-диаграмма, на которой показано разложение некоторой деятельности на ее составные части. Под деятельностью (англ. activity) понимается спецификация исполняемого поведения в виде координированного последовательного и параллельного выполнения подчиненных элементов — вложенных видов деятельности и отдельных действий англ. action, соединенных между собой потоками, которые идут от выходов одного узла ко входам другого.

Диаграммы деятельности используются при моделировании бизнес-процессов, технологических процессов, последовательных и параллельных вычислений.

Диаграмма деятельности представлена на рисунке 3.3.1.

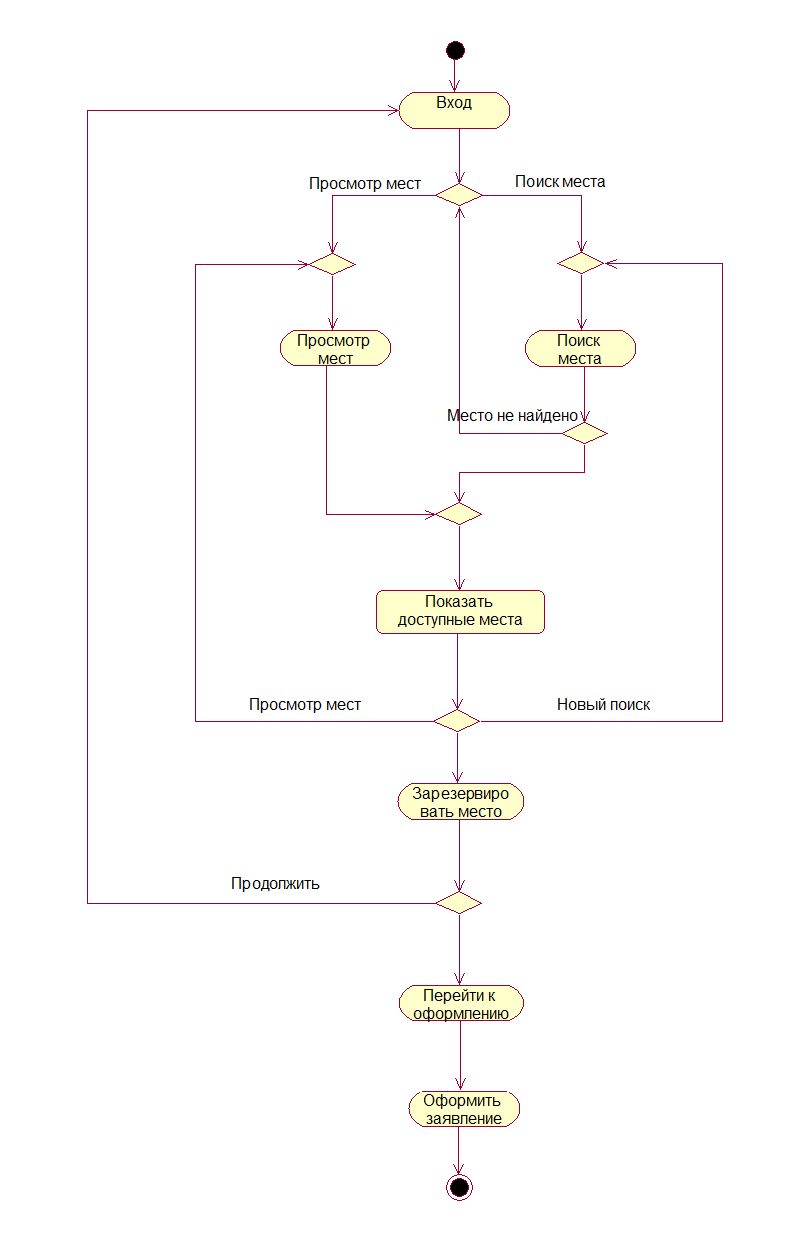


Рисунок 3.3.1 – Диаграмма деятельности

### 3.4 Диаграмма кооперации

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

Прежде всего, на диаграмме кооперации в виде прямоугольников изображаются участвующие во взаимодействии объекты, содержащие имя объекта, его класс и, возможно, значения атрибутов. Далее, как и на диаграмме классов, указываются ассоциации между объектами в виде различных соединительных линий. При этом можно явно указать имена ассоциации и ролей, которые играют объекты в данной ассоциации. Дополнительно могут быть изображены динамические связи - потоки сообщений. Они представляются также в виде соединительных линий между объектами, над которыми располагается стрелка с указанием направления, имени сообщения и порядкового номера в общей последовательности инициализации сообщений.

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.

Диаграмма коопераций представлена на рисунке 3.4.1.

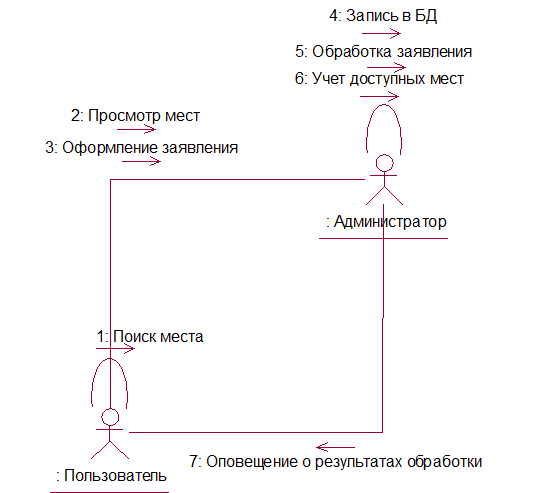


Рисунок 3.4.1 – Диаграмма коопераций

### 3.5 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.

Основными элементами диаграммы последовательности являются обозначения объектов (прямоугольники с названиями объектов), вертикальные «линии жизни», 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.

Диаграмма последовательности представлена на рисунке 3.5.1.

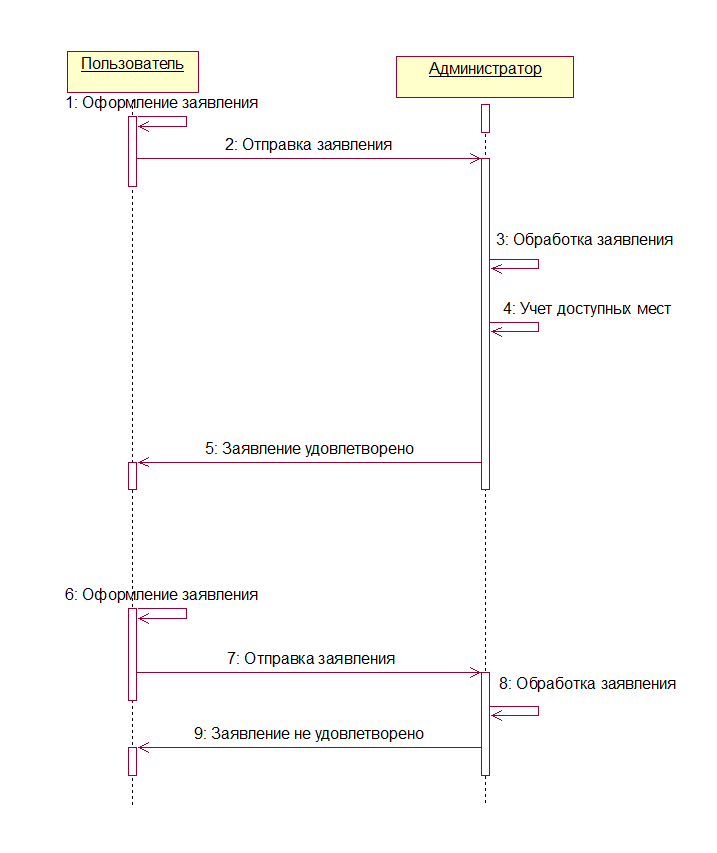


Рисунок 3.5.1 – Диаграмма последовательности

### 3.6 Диаграмма развертывания

Диаграмма развертывания 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).

Узлы представляются как прямоугольные параллелепипеды с артефактами, расположенными в них, изображенными в виде прямоугольников. Узлы могут иметь под узлы, которые представляются как вложенные прямоугольные параллелепипеды. Один узел диаграммы развертывания может концептуально представлять множество физических узлов, таких как кластер серверов баз данных.

Существует два типа узлов:

* узел устройства;
* узел среды выполнения.

Узлы устройств — это физические вычислительные ресурсы со своей памятью и сервисами для выполнения программного обеспечения, такие как обычные ПК, мобильные телефоны. Узел среды выполнения — это программный вычислительный ресурс, который работает внутри внешнего узла и который предоставляет собой сервис, выполняющий другие исполняемые программные элементы.

Диаграмма развертывания представлена на рисунке 3.6.1.

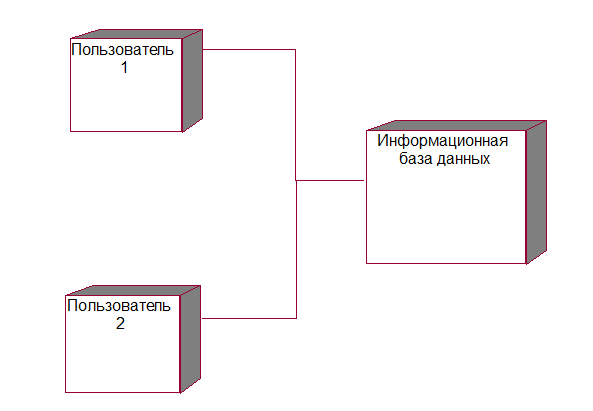


Рисунок 3.6.1 – Диаграмма развертывания

### 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 интерфейс составного компонента могут делегироваться в соответствующие интерфейсы внутренних компонентов.

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

Диаграмма компонентов представлена на рисунке 3.7.1.

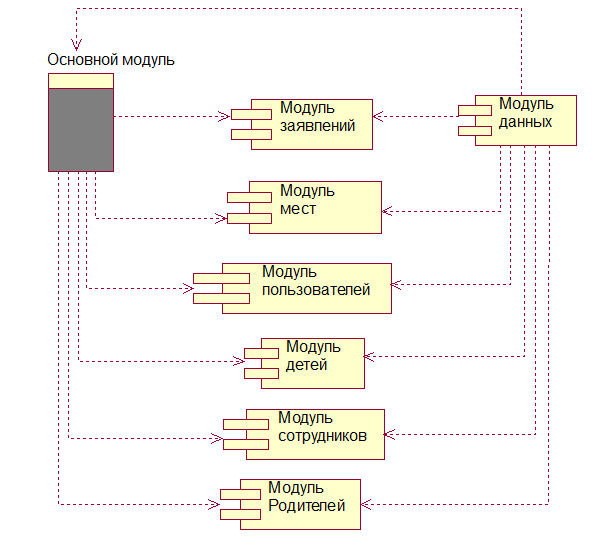


Рисунок 3.7.1 – Диаграмма компонентов

## 4 ГЕНЕРАЦИЯ КОДА

На рисунках 4.1 — 4.5 представлен процесс генерации кода на языке Java, непосредственно сам код расположен ниже.

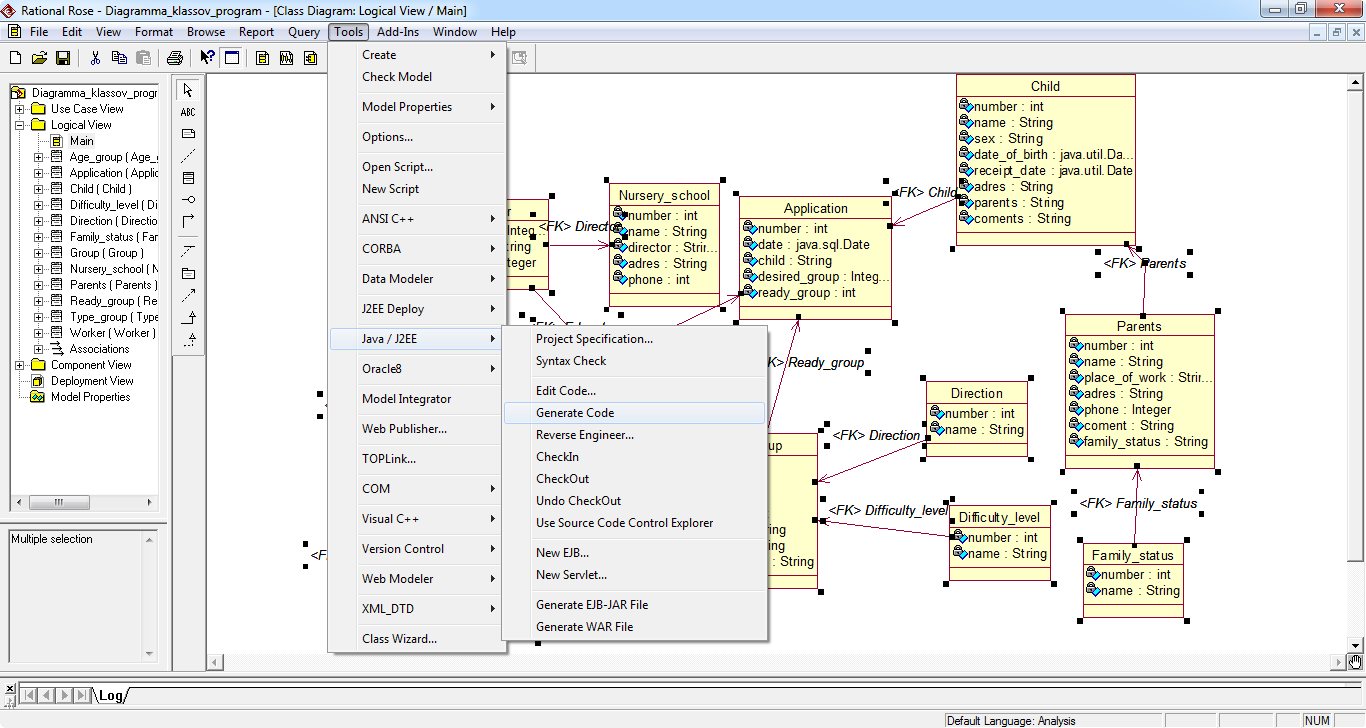


Рисунок 4.1 – Процесс генерации кода

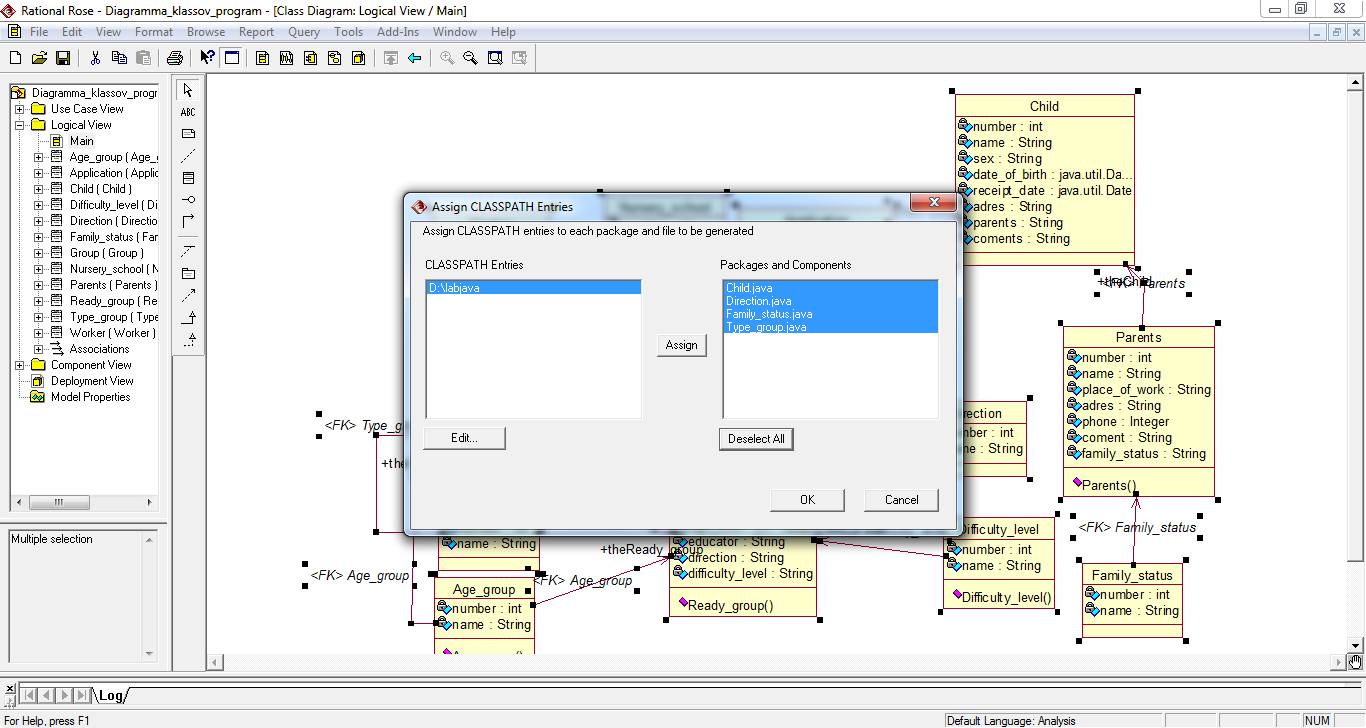


Рисунок 4.2 – Процесс генерации кода

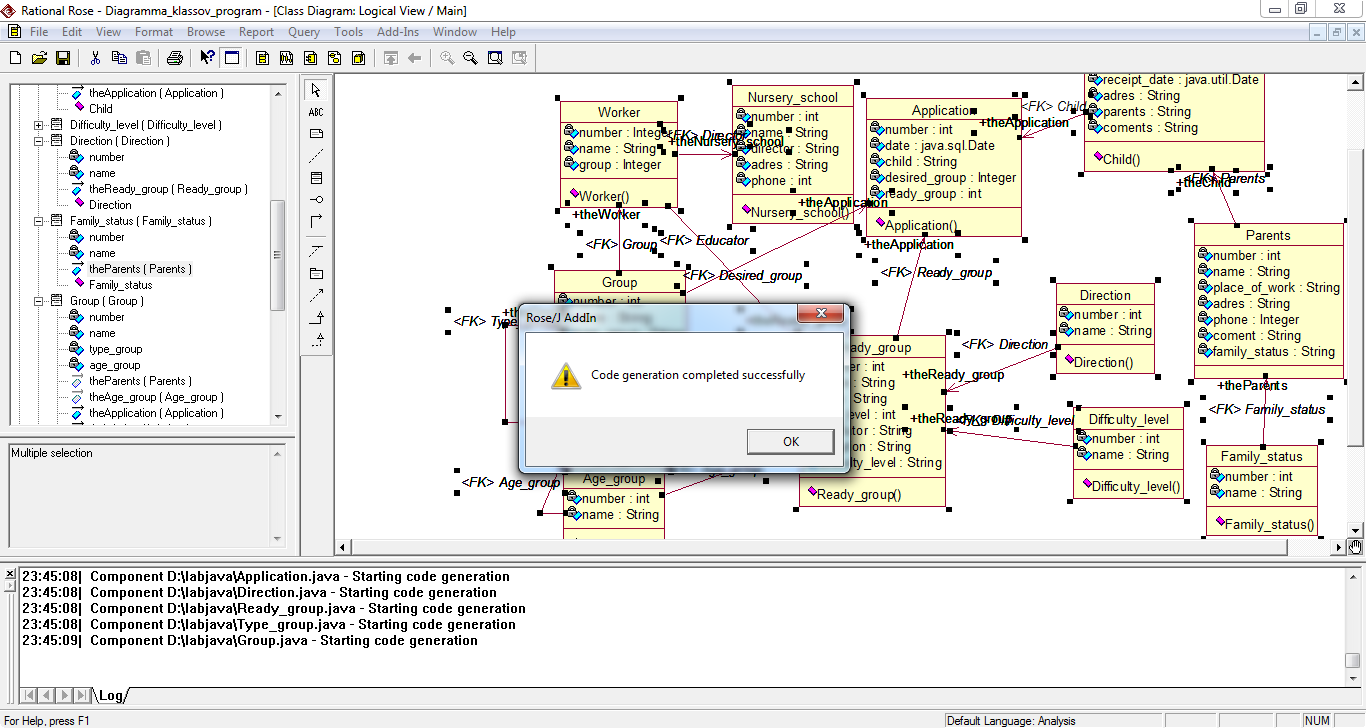


Рисунок 4.3 – Процесс генерации кода

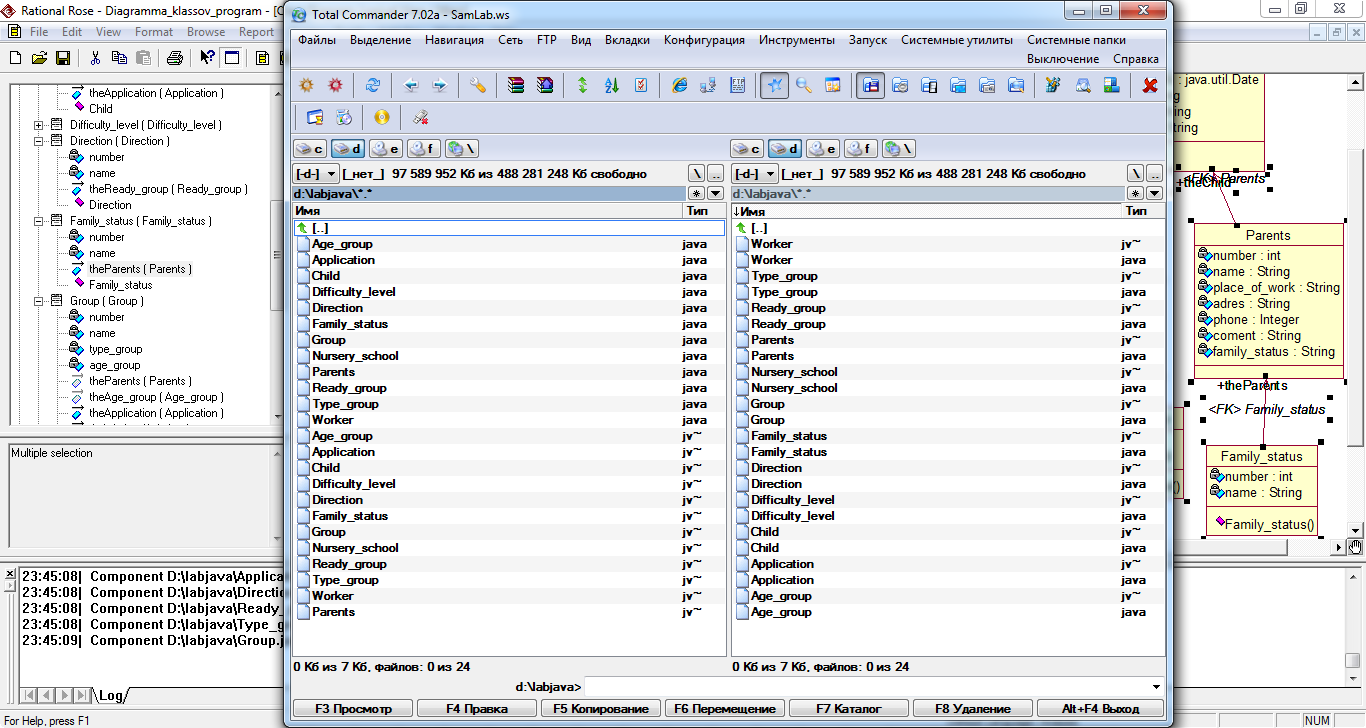


Рисунок 4.4 – Процесс генерации кода

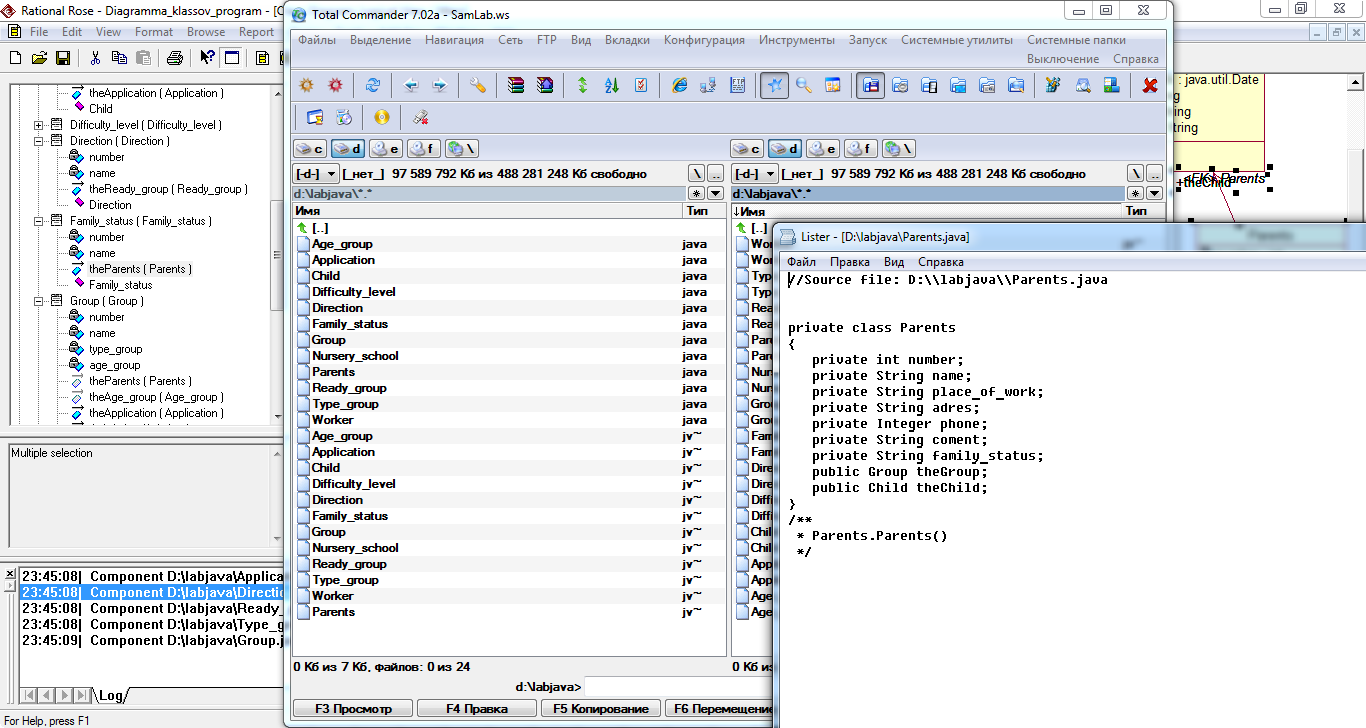


Рисунок 4.5 – Процесс генерации кода

Java — строго типизированный объектно-ориентированный язык программирования, разработанный компанией Sun Microsystems (в после-дующем приобретенной компанией Oracle). Приложения Java обычно транслируются в специальный байт-код, поэтому они могут работать на любой компьютерной архитектуре, с помощью виртуальной Java-машины. Дата официального выпуска — 23 мая 1995 года.

Язык Java активно используется для создания мобильных приложений под операционную систему Android. При этом программы компилируются в нестандартный байт-код, для использования их виртуальной ма-шиной Dalvik (начиная с Android 5.0 Lollipop виртуальная машина заменена на ART). Для такой компиляции используется дополнительный инструмент, а именно Android SDK (Software Development Kit), разработанный компанией Google.

Разработку приложений можно вести в среде Android Studio, NetBeans, в среде Eclipse, используя при этом плагин Android Development Tools (ADT), или в IntelliJ IDEA. Версия JDK при этом должна быть 5.0 или выше.

Сгенерированный код:

public class Age\_group

{

private int number;

private String name;

public Worker theWorker;

public Ready\_group theReady\_group;

public Group theGroup;

/\*\*

\* @roseuid 58926341006D

\*/

public Age\_group()

{

}

}

import java.sql.Date;

public class Application

{

private int number;

private Date date;

private String child;

private Integer desired\_group;

private int ready\_group;

/\*\*

\* @roseuid 589263410010

\*/

public Application()

{

}

}

import java.util.Date;

public class Child

{

private int number;

private String name;

private String sex;

private Date date\_of\_birth;

private Date receipt\_date;

private String adres;

private String parents;

private String coments;

public Application theApplication;

/\*\*

\* @roseuid 5892644C010E

\*/

public Child()

{

}

}

public class Difficulty\_level

{

private int number;

private String name;

public Worker theWorker;

public Ready\_group theReady\_group;

/\*\*

\* @roseuid 589263400320

\*/

public Difficulty\_level()

{

}

}

public class Direction

{

private int number;

private String name;

public Ready\_group theReady\_group;

/\*\*

\* @roseuid 589264BC00CA

\*/

public Direction()

{

}

}

public class Family\_status

{

private int number;

private String name;

public Parents theParents;

/\*\*

\* @roseuid 589264BC0006

\*/

public Family\_status()

{

}

}

private class Group

{

private int number;

private String name;

private String type\_group;

private int age\_group;

public Application theApplication;

public Worker theWorker;

}

/\*\*

public Group.Group()

\*/

public class Nursery\_school

{

private int number;

private String name;

private String director;

private String adres;

private int phone;

/\*\*

\* @roseuid 5892634003BC

\*/

public Nursery\_school()

{

}

}

private class Parents

{

private int number;

private String name;

private String place\_of\_work;

private String adres;

private Integer phone;

private String coment;

private String family\_status;

public Group theGroup;

public Child theChild;

public Parents.Parents()

\*/

{

}

}

public class Ready\_group

{

private int number;

private String name;

private String type;

private int age\_level;

private String educator;

private String direction;

private String difficulty\_level;

public Nursery\_school theNursery\_school;

public Child theChild;

public Application theApplication;

/\*\*

\* @roseuid 589263400362

\*/

public Ready\_group()

{

}

}

public class Type\_group

{

private int number;

private String name;

public Nursery\_school theNursery\_school;

public Child theChild;

public Ready\_group theReady\_group;

public Group theGroup;

/\*\*

\* @roseuid 589264BC0129

\*/

public Type\_group()

{

}

}

public class Worker

{

private Integer number;

private String name;

private Integer group;

public Application theApplication;

public Ready\_group theReady\_group;

public Nursery\_school theNursery\_school;

/\*\*

\* @roseuid 5892634100B7

\*/

public Worker()

{

}

}

## ЗАКЛЮЧЕНИЕ

Предметной областью контрольной работы является программное средство детского сада. Разработанное программное средство позволит быстро и удобно предоставить полную функциональность и информацию для дошкольного учреждения, значительно упростив его деятельность.

Пояснительная записка состоит из следующих разделов:

1. Постановка задачи. В данном разделе была поставлена задача по разработке программного средства детского сада.

2. Описание основного процесса предметной области с использованием стандарта IDEF0.

3. Описание информационной модели с использованием стандарта IDEF1.

4. Описание моделей информационной системы с использованием языка UML.

5. Генерация кода на основе моделей Rational Rose.

В ходе написания контрольной работы были закреплены теоретические знания по таким разделам как составление функциональной модели (IDEF0), разработка логической и физической модели IDEF1X, проектирование системы с помощью унифицированного языка моделирования (UML).

Контрольная работа выполнялась в соответствии с методическими требованиями и указаниями.

## СПИСОК ИСПОЛЬЗОВАНЫХ 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.