МОСКОВСКИЙ ИНСТИТУТ ЭЛЕКТРОНИКИ И МАТЕМАТИКИ НИУ ВШЭ

Кафедра ИТАС

**ПОЯСНИТЕЛЬНАЯ ЗАПИСКА**

К ДИПЛОМНОЙ РАБОТЕ

На тему\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Студент\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Руководитель проекта\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Допущен к защите\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_20\_\_г.

КОНСУЛЬТАНТЫ ПРОЕКТА:

Специальная часть\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

*Зав. кафедрой\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_*

МОСКВА

ВВЕДЕНИЕ 3

ГЛАВА 1: ПОСТАНОВКА ЗАДАЧИ 6

ГЛАВА 2: АНАЛИЗ ПРЕДМЕТНОЙ ОБЛАСТИ 8

ГЛАВА 3: АРХИТЕКТУРА СИСТЕМЫ 11

3.1 Выбор типа приложения 11

3*.*2 Архитектура. 13

ГЛАВА 4: АРХИТЕКТУРА БАЗЫ ДАННЫХ 17

4.1. Первоначальная инфологическая модель. 18

4.2 Схема базы данных. 21

4.3 Атрибуты 24

4.4. Нормализация 35

ГЛАВА 5: ОРГАНИЗАЦИЯ ПОЛЬЗОВАТЕЛЬСКОГО ИНТЕРФЕЙСА 39

5.1. Аналоги 40

5.2. Интерфейс оператора. 44

5.3. Интерфейс пользователя 50

ЗАКЛЮЧЕНИЕ 54

Список литературы 56

ПРИЛОЖЕНИЯ 57

# ВВЕДЕНИЕ

Современные технические устройства характеризуются сложностью исполнения, а также значительным разнообразием комплектации, что обеспечивает возможность выбора оборудования для конкретных задач, исполнение которых необходимо заказчику.

Предприятия, производящие сложное высокотехнологичное оборудование (например, промышленных или строительных роботов), вынуждены пользоваться довольно обширными общими каталогами деталей и чертежей, в которых зачастую трудно ориентироваться как заказчику, так и продавцу. Также существенным недостатком имеющиеся каталогов оборудования и запчастей является необходимость их постоянного обновления при появлении новых моделей. Такие списки достаточно просты в составлении, но приводят к проблемам на этапе обсуждения конечной комплектации, так как окончательное соглашение по комплектации оборудования, в связи с его сложностью и высокой стоимостью, обычно происходит во время переговоров сторон - заказчика и поставщика. Соответственно, представление информации об оборудовании должно быть представлено в удобной форме для всех сторон процесса - заказчик должен иметь чёткое представление о возможных вариантах комплектации, а представитель производителя, в свою очередь, должен иметь возможность предоставить эти варианты в удобной для понимания форме.

Так как зачастую оборудования для строительных работ производится для выполнения конкретных задач, комплекты поставки могут достаточно сильно различаться. То, что представлено в каталоге как отдельная модель, в реальности может иметь различные опции, от выбора которых зависит соответствие поставляемого оборудования требованиям, которые предъявляет к нему заказчик. В свою очередь, заказчик обычно ориентируется на конкретную строительную задачу, но при этом сторона производителя обеспечивает ему возможность изменения комплектации имеющегося оборудования для дальнейшего использования других условиях, в определённых пределах отличных от тех, для которых оборудование приобреталось изначально.

Создание системы обработки заказов значительно упростит процесс формирования спецификации оборудования, а также обеспечит заказчику доступ к информации об уже имеющихся или интересующих его конкретных изделиях.

В случае сложного оборудования трудно применить те же решения, как для более массовых изделий, таких как, например, легковые автомобили. Они также имеют разнообразные опции, но в рамках одной модели их разнообразие гораздо меньше, и к тому же комплекты поставки автомобилей более стандартизированы.

В случае высокотехнологичного строительного оборудования нецелесообразно использовать информационную систему, аналогичную интернет - магазину автозапчастей. Для этого оборудования характерна модульная структура из нескольких уровней, менее развитая логистическая поддержка и высокая стоимость. Последние два фактора связаны с небольшим объёмом и сложностью производства (так как даже из стандартных компонентов, присутствующих в каталоге, оборудование всё равно собирается в соответствии с требованиями заказчика).

Для обеспечения удобства подбора комплектующих производителю было бы удобно иметь структурированное представление об оборудовании. А именно как о наборе из обязательных и опциональных узлов. Такое представление позволят описывать состав и функциональность готовой единицы продукции исходя из свойств отдельных модулей, что позволяет легко формировать информационные представления для новых моделей, и отражать разнообразие вариантов комплектации для уже имеющихся.

В свою очередь для заказчика посредством данных наборов будет обеспечена возможность доступа к каталогам запчастей, подходящих конкретно для приобретённых им устройств, а также сможет отслеживать изменения в комплектации и доступные опции, связанные с имеющимся у него оборудованием. Также реализация такого конструктора для формирования комплектации позволит заказчику самостоятельно подобрать себе оборудование и тем самым упростить общение между сторонами.

Основная цель работы состоит в формировании принципов построения информационной системы, которая обеспечит удобство представления специализированного оборудования в виде наборов модулей, обладающих определёнными характеристиками, с возможностью формирования взаимосвязей между этими модулями. Такая организация информации об оборудовании снизит зависимость производителя и заказчика от традиционных каталогов оборудования и повысит возможность обеспечения актуальности информации о нём.

# ГЛАВА 1: ПОСТАНОВКА ЗАДАЧИ

Создаваемая система обработки заказов должна обеспечить удобство взаимодействия заказчика и поставщика оборудования.

Основные требования к этому взаимодействию в рамках создаваемой системы таковы:

1. Наглядность информационного представления оборудования.
2. Возможность поиска информации по большому количеству критериев.
3. Обеспечение простоты создания информационного представления оборудования.
4. Удалённый доступ к системе для производителя и потребителя.
5. Подбор списка запчастей к имеющемуся у заказчика оборудованию.

В данном случае оптимальным способом реализации системы является использование web-интерфейса. Это позволит собрать весь функционал в рамках одной программы, обеспечить удалённый доступ как для операторов со стороны производителя (продавца), так и для заказчиков, а также сведёт системные требования для клиента к наличию браузера и подключения к интернету.

Наглядность информационно представления оборудования может быть достигнута путём реализации этого представления в виде структуры из нескольких уровней:

1. Деталь, её чертежи и собственные свойства, атрибуты для описания разнородных свойств.
2. Узел – состоит из совокупности деталей, помимо этого также имеет отдельный набор атрибутов
3. Изделие – чертежи, фотографии и общие характеристики узлов, плюс собственные свойства, которые не могут быть описаны через более низкоуровневые сущности.

Таким образом, при изменении какого-либо компонента из комплекта поставки, заказчик или представитель поставщика сразу увидит изменения на самом высоком уровне представления (изделие). Создание представления оборудования сводится к комбинированию модулей и установлению разрешений на связи между ними (дополняющие, взаимоисключающие и т.д.)

# ГЛАВА 2: АНАЛИЗ ПРЕДМЕТНОЙ ОБЛАСТИ

В отличие от розничной торговли через обычные интернет-магазины с возможностью немедленной оплаты, закупка оборудования для специфических строительных работ проходит в несколько этапов:

* отправка заявки со стороны заказчика
* ответное коммерческое предложение со стороны продавца.
* переговоры по комплектации оборудования

Соответственно, в заявке заказчик формирует основные требования к оборудованию. В случае специфических проектов от заказчика может потребоваться дополнительная информация и техническое задание. В коммерческом предложении поставщик даёт описание изделия, соответствующего нуждам заказчика. Далее обычно следует стадия переговоров и, возможно, формирования нового коммерческого предложения.

Набор требуемого оборудования может достаточно сильно различаться в зависимости от задачи, стоящей перед заказчиком – например, робот Conjet 324, являющийся устройством для гидродемонтажа может использоваться как для выборочного демонтажа мостового бетонного покрытия, так и, например, для очистки корпусов кораблей. Соответственно, варьировать набор опций можно в достаточно широких пределах – мощность водомёта, наличие/отсутствие радиоуправления, система электропитания и многое другое.

Также, помимо разнообразной стандартной аппаратуры, заказчику может потребоваться оборудование, специально спроектированное под его конкретную задачу (для этого как раз и нужно предоставление технического задания на строительные работы). Такое оборудование штучного производства сложно сопровождать впоследствии – фактически, оно требует создания отдельного каталога запчастей.

В процессе обсуждения заказа его комплектация может меняться, что оказывает влияние на возможное время изготовления и доставки, общую стоимость. Особенно это выражено при заказе оборудования под специфические задачи.

Во время эксплуатации оборудования у заказчика возникает потребность в запасных частях и некоторых расходных материалах. Конечному пользователю приходится ориентироваться по предоставленному производителем каталогу, информация в котором зачастую предоставлена без привязки к конкретному серийному номеру оборудования и другим узлам агрегата, что обычно приводит к потребности в обращении к стороне производителя за разъяснениями. В общем каталоге обычно отсутствует представление о взаимоисключающих узлах. В свою очередь, производитель вынужден ориентироваться на те же каталоги. Такая ситуация может приводить к ошибкам при заказе и доставке оборудования.

При использовании каталога обычного вида пользователь имеет дело с информацией, не относящейся к цели его поиска. При первоначальном заказе оборудования эта проблема не так очевидна, но при поиске замены комплектующих для существующего оборудования вызывает неудобство. Такое представление списка оборудования является основным источником ошибок при заказе.

В обычных каталогах указывается, что имеющаяся в нём информация может быть изменена, без соответствующих изменений в данном каталоге, который в классическом виде представляет собой текстовый документ с чертежами и спецификациями. Таким образом, производитель вынужден постоянно выпускать новые версии, поддерживая актуальность распространяемой информации. Постоянный выпуск новых каталогов является потенциальным источником ошибок в описаниях оборудования. Соответственно, выпуск каталога с ошибкой впоследствии потребует ещё одного переиздания.

В отличие от обычных каталогов, электронные каталоги несут в себе информацию обо всей продукции предприятия. При этом она разделена на соответствующие категории. В большинстве случаев это приводит к созданию крупного древовидного списка, в котором искомый объект и вся приложенная к нему информация находится на самом нижнем уровне иерархии. В некоторых вариантах пользователь может воспользоваться историей просмотров – это его единственная возможность возобновить работу с ранее посещёнными разделами, т.к. в связи с реализацией приложений страницы генерируются при запросе, следовательно, добавление адреса в избранное не поможет найти необходимую информацию позже.

Таким образом, электронный каталог представляет собой некоторую модификацию каталога обычного вида, а зачастую содержит эти каталоги в себе, в придачу к собственной информации. Вследствие этого, производитель вынужден обновлять 2 каталога вместо одного, но уже имеет некоторое преимущество в скорости и лёгкости этого обновления. Хорошим вариантом в этом случае было бы использование шаблонов автоматической генерации документов на основе информации из базы данных для автоматического обновления вложенных каталогов.

# ГЛАВА 3: АРХИТЕКТУРА СИСТЕМЫ

3.1 Выбор типа приложения

Автоматизированные системы давно и прочно обосновались во многих сферах деятельности человека, сделав хранение и обработку данных различного объёма более простыми и удобными. Развитие сети Internet позволило со временем сделать доступ к программному комплексу удалённым посредством реализации концепции «клиент-сервер», в которой централизованным хранением и обработкой данных по запросу клиента занимается серверное приложение.

Для разрабатываемой системы оптимальным вариантом будет реализация в виде web-приложения. Это позволит собрать весь функционал в рамках одной программы, обеспечить удалённый доступ, как для операторов, так и для заказчиков, а также сведёт системные требования для клиента к наличию браузера и подключения к интернету.

В рамках единой программы реализуется весь функционал приложения. Примером такой организации системы может служить Microsoft Exchange Server, обеспечивающий обработку и пересылку сообщений электронной почты, работу с календарем и многое другое. В качестве клиентского приложения используется, например, Microsoft Outlook или Outlook Express, которые представляют собой программы, установленные на компьютер пользователя, а также Outlook Web Access, представляющий из себя веб-клиент, практически аналогичный по функциональности Microsoft Outlook.

Этот пример позволяет увидеть основные различия между web-клиентом и приложением, которое устанавливается на компьютер пользователя.

Использование web-клиента даёт серьёзное преимущество пользователю – при обновлении программного комплекса не требуется переустановка клиентских компонентов программы, что ускоряет процессы внедрения программного комплекса или перехода на его новую версию. Собственно, в этом случае вообще не требуется проводить какие-то работы на машине пользователя, если не возникает проблема отображения страниц, получаемых от сервера, в браузере.

Если предполагается, что веб-приложение будет достаточно простым (например, блог или небольшой интернет-магазин), целесообразно использовать готовую систему управления содержимым, предоставляющую пользовательский интерфейс для создания и управления компонентами. Недостатком такого подхода является некоторая избыточность функций, и, соответственно, содержимого страниц сайта.

Если же приложение имеет более специфические функции (например, система электронного документооборота, в которую заложена зачастую достаточно сложная логика управления движением документов по бизнес-процессу, или разрабатываемая система управления заказами, предусматривающая возможность конструирования наборов данных, соответствующих определённому объекту), целесообразно использовать более гибкие инструменты (например, каркасы приложения (framefork) Django или Ruby on Rails).

Необходимые действия по установке обычно сводятся к обеспечению удалённого доступа посредством настройки сети или сетевых модулей (если работа с приложением возможно только из локальной сети предприятия) и создания учётной записи для работы с системой. Такой принцип подходит для систем различного вида, таких как системы электронного документооборота, управления проектами (Redmine, серверное веб-приложение), контроля версий (Git, в частности gerrit и gitweb).

Интерфейс управления может быть реализован также в виде сайта, что частично избавляет администратора системы от внесения правок в базу данных или код приложения. Примером такого подхода является Django - фреймворк для веб-приложений на языке Python, имеющий встроенное приложение для управления содержимым одного или нескольких сайтов.

Использование веб-интерфейса также обеспечивает преимущество в сотрудничестве с собственными представительствами и другими компаниями – нет необходимости использования и синхронизации нескольких баз данных, разграничение доступа к данным осуществляется с помощью набора прав, привязанных к учётной записи пользователя.

Также важным аспектом является возможность использования приложения из разных операционных систем. Нет необходимости написания различных версий, например для Windows и UNIX. Присутствуют различия в отображении элементов веб-страниц в браузерах, но в их случае они не столь фундаментальны.

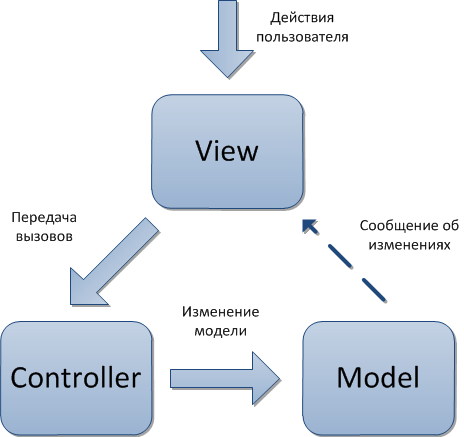
Для реализации данного приложения будет использоваться каркас приложения (framework) Ruby on Rails, созданный с помощью объектно-ориентированного языка Ruby

**3.2 Архитектура.**

В современной индустрии информационных технологий, связанных с разработкой веб-приложений часто используются «каркасы» (frameworks), которые задают общую архитектуру приложения. Например, Ruby on Rails, используемы для данного проекта, реализует архитектурный образец «модель-представление-контроллер» (model-view-controller, MVC).

«Концепция MVC позволяет разделить данные, представление и обработку действий пользователя на три отдельных компонента:

1. Модель (англ. Model). Модель предоставляет знания: данные и методы работы с этими данными, реагирует на запросы, изменяя своё состояние. Не содержит информации, как эти знания можно визуализировать.
2. Представление, вид (англ. View). Отвечает за отображение информации (визуализацию). Часто в качестве представления выступает форма (окно) с графическими элементами.
3. Контроллер (англ. Controller). Обеспечивает связь между пользователем и системой: контролирует ввод данных пользователем и использует модель и представление для реализации необходимой реакции» [1]



*Рис.1 Схема взаимодействия компонентов в рамках архитектуры MVC*

Ruby on Rails предоставляет разработчику возможность работать с объектами реляционной базы данных как с экземплярами класса в объектно-ориентированном программировании. Таблица реляционной базы данных представляется как класс, а строки таблицы, соответственно, как экземпляры класса. При вызове метода объекта, возвращающего значение его атрибутов, происходит считывание информации из базы данных. При создании экземпляра класса данные вносятся в таблицу реляционной базы данных.

Как было упомянуто выше, модель данных никак не участвует непосредственно в отображении информации, она лишь формирует необходимый объём данных по запросу контроллера.

Для вызова нужного контроллера в Rails используется специальный механизм маршрутизации, с помощью которого на основе запрашиваемого браузером URI (Uniform Resource Identifier, последовательность символов, идентифицирующая абстрактный или физический ресурс [1]) и http - метода запроса определяется, какое действие должно быть совершено. На основании этого выбирается нужный контроллер, обеспечивающий выполнение требуемого действия.

Для иллюстрации примера работы приложения, предположим, что заказчик собирается просмотреть описание оборудования, выбрав его из списка.

В упрощённом виде последовательность операций, которые совершает приложение, можно описать так:

1. От браузера поступает запрос на страницу, соответствующую определённой единице оборудовния. В нашем примере это будет /machinery/42.
2. В контроллере machinery программа находит действие show (показать).
3. С помощью действия show происходит обращение к модели machinery c целью получить информацию о сущности machinery c идентификатором 42.
4. Модель machinery выполняет соответствующий запрос к базе данных.
5. Модель возвращает контроллеру machinery полученную из запроса информацию.
6. В некой переменной контроллер передаёт представлению machinery полученные данные.
7. Представление генерирует визуализацию страницы с полученным от контроллера набором данных в HTML.
8. Контроллер возвращает браузеру сгенерированный представлением HTML-код страницы

Такая реализация приложения позволяет разделить процесс разработки – модели представления могут создаваться более-менее независимо. Для представления не важна внутренняя бизнес-логика модели, оно поучит от контроллера информацию в том виде, в котором она потребуется.

Из приведённого примера может сложиться мнение, что модель – это просто интерфейс для взаимодействия приложения с базой данных. На самом деле в моделях реализуется основная бизнес-логика приложения («совокупность правил, принципов, зависимостей поведения объектов предметной области»[1]), а как раз контроллеры являются интерфейсами для передачи данных и запросов между различными компонентами приложения. Компоненты-представлений не занимаются обработкой данных, только выводом в нужной для пользователя форме.

# ГЛАВА 4: АРХИТЕКТУРА БАЗЫ ДАННЫХ

Для реализации системы используется реляционная модель данных. В качестве одного из вариантов рассматривалась также иерархическая модель, но её использование ведёт в нашем случае к большому количеству дублирующих записей. Реляционная модель данных основывается на теории множеств и реляционной алгебре, которая является набором операций над отношениями. В реляционной модели данных используется теория нормализации, позволяющая уменьшить избыточность отношений (таблиц) с целью защиты от логических ошибок и аномалий, связанных с неоптимизированным разделением сущностей предметной области на отношения.

В качестве системы управления базами данных используется Firebird 2.5.

«Системой управления базами данных называют программную систему, предназначенную для создания на ЭВМ общей базы данных для множества приложений, поддержания её в актуальном состоянии и обеспечения эффективного доступа пользователей к содержащимся в ней данным в рамках предоставленных им полномочий»[2].

Эта СУБД поддерживает стандарт SQL-92, и при малом объёме собственно самой СУБД способна работать с объёмными базами данных (до 1ТБ ). Firebird имеет версии для многих операционных систем, база данных легко переносима и может быть восстановлена из резервной копии на любой из них. Следует отметить, что Firebird - полностью свободное программное обеспечение, не требующее лицензионных отчислений при любом варианте использования [3].

Реализация базы данных для разрабатываемого приложения не требует специфических функций какой-либо конкретной СУБД, так как требуемые возможности укладываются в границы, обозначенные стандартом SQL-92.

Специфическим для различных СУБД в нашем случае могут быть несколько моментов – реализация последовательностей (генераторов) для автоинкрементации суррогатных числовых первичных ключей, а также реализация логического типа данных, которая может различаться в разных СУБД.

4.1. Первоначальная инфологическая модель.



*Рис.1 Диаграмма «Сущность-связь» (ER-диаграмма)*

На основании анализа предметной области была построена её инфологическая модель в виде диаграммы «Сущность - связь» (entity-relationship model, рис.1). Основными классами объектов (сущностями), которые были выделены в ходе анализа, являются:

1. оборудование
2. узел
3. атрибут узла
4. деталь
5. заказчик
6. заказ
7. оператор
8. чертёж

Информация о единице оборудования формируется из последовательно связанных сущностей, каждая из которых обладает как собственными, специфическими для неё связями (например, атрибут узла), так и общими (чертёж узла, заказ узла, заказ детали)

Идентифицирующим атрибутом оборудования как агрегата является название модели, для идентификации единицы оборудования используется серийный номер. Оборудование обязательно определяется как совокупность узлов. Оборудование имеет своё описание, но в общем случае его характеристики строятся из характеристик узлов. Таким образом, для создания новой модели на основе существующего описания не нужно постоянно его переписывать, достаточно задать атрибуты нового ил изменённого узла.

Характеристики оборудования вынесены в отдельные сущности «Строковый атрибут» и «Числовой атрибут», для оптимизации хранения данных и упрощения реализации поиска. Поиск по числовым атрибутам (давление, мощность, скорость, температура и т.д.) в этом случае не потребует преобразования строковых значений в числовые. Также выделение числовых атрибутов в отдельную сущность (следовательно, и таблицу базы данных) исключает возможность хранения информации с частью ошибок, появляющихся вследствие человеческого фактора – использование разных разделителей (запятая или точка), их дублирование или неверное расположение (например, 100,00,00) попадание букв и других нечисловых символов в значение числового атрибута. Атрибуты обоих типов задаются отдельно для оборудования, узлов и деталей.

Узлы в общем случае состоят из отдельных деталей, но при заказе комплектующих оборудования не исключается возможность того, что узел поставляется только в готовом виде.

Деталь является самым нижним уровнем в иерархии элементов, определяющих оборудование. Для деталей выделен специфический атрибут, определяющий минимальное количество при поставке – например, крепёжные изделия, специфические для данного типа оборудования, но использующиеся в большом количестве.

Пользователи системы рассматриваются как сущности в модели предметной области, так как для их описания нужно достаточно большое количество атрибутов, которые определяют взаимодействие с остальными выделенными сущностями.

Пользователи разделены на операторов, которые занимаются непосредственно вводом и модификацией данных об оборудовании, его комплектации и характеристиках, и заказчиков, имеющих доступ к информации об имеющемся у них оборудовании и его компонентах.

В создаваемой системе для просмотра информации о различных моделях изделий учётная запись пользователя не является строго обязательной. Она нужна для формирования набора информации, выводящегося в личный кабинет пользователя, отслеживания им состояния и состава текущих и выполненных заказов.

Заказ состоит из информации об оборудовании и деталях, которые входят в его состав, а также о его статусе и стоимости. Так как такая информация является конфиденциальной, она доступна для просмотра только зарегистрированному заказчику и представителям поставщика.

Учётные записи операторов нужны для контроля доступа к различным типам оборудования и отслеживания изменений объектов базы данных. Таким образом, если система используется для информационного сопровождения нескольких компаний, будет достаточно просто разделить доступ между ними, связав учётную запись пользователя с организацией. Для организаций формируются отдельные группы пользователей. Назначив пользователю принадлежность к такой группе, можно регулировать доступ к различным элементам по видам оборудования.

Каждая составляющая оборудования может иметь один или несколько чертежей (фотографий). Чертежи для разных уровней компонентов оборудования не имеют выраженной специфики, поэтому представлены на диаграмме как единая отдельная сущность.

4.2 Схема базы данных.

Приведённая в предыдущем разделе ER-диаграмма преобразуется в схему базы данных. На этом этапе выделенным при моделировании предметной области сущностям ставится в соответствие структура, состоящая из таблиц реляционной базы данных. В отличие от ER диаграммы здесь явно обозначены ключи, о которым осуществляется связь между таблицами. Приведённая схема не является окончательной, так как некоторые отношения (таблицы) могут изменены – объединены или разделены – на этапе нормализации отношений с целью обеспечения логической целостности хранения данных, которая позволит избежать появления большинства аномалий и ошибок при работе с базой данных.

Схема базы данных является формальным описанием её структуры, определяющим таблицы, их поля и связи между ними.



*Рис.2 Схема базы данных*

На этом этапе вводятся дополнительные связующие таблицы. Необходимость введения такой таблицы зависит от степени отношения между сущностями, а именно в случае отношения «многие ко многим»

Отношения «один ко многим» реализуются с помощью внешних ключей – ссылок на первичный ключ (атрибут, однозначно идентифицирующий строку таблицы). Отношения «один к одному» в базе данных отсутствуют – они равносильны объединению двух отношений в одно.

Примером реализации такой связи может служить таблица «*заказанные детали*». Эта таблица с помощью внешних ключей ссылается на две другие – Деталь и Заказ, используя их уникальные идентификаторы – модель детали и номер заказа. Дополнительные связующие таблицы помимо ключей могут содержать дополнительную информацию. В случае таблицы «*заказанные детали*» такой дополнительной информацией будет количество деталей.

Так как таблицы для разных уровней элементов оборудования разделены, необходимо также ввести связующие таблицы, с помощью которых можно определить, к какому компоненту относится чертёж или фотография. В дальнейшем для упрощения доступа к чертежам определённого типа целесообразно будет создать представление (хранимый запрос), опирающееся на это сочетание.

Один чертёж или атрибут может принадлежать только одной сущности (узел, деталь, оборудование), но в нашем случае связь этих отношений реализуется с помощью связующих таблиц, а не внешнего ключа, так как сущности компонентов и оборудования идентифицируются по отдельности, следовательно множества числовых идентификаторов могут пересекаться.

Для обеспечения нормальной логики работы СУБД те связи, которые на ER-диаграмме (*рис.1*) были обозначены как обязательные в обе стороны, заменены на обязательные по отношению только к одной из связанных таблиц. Это нужно для того чтобы убрать логическое противоречие. В качестве примера можно рассмотреть связь между сущностями «Оборудование» и «Тип оборудования». Если связь обязательна по отношению к обеим таблицам, невозможно было бы создать оборудование без типа и тип без оборудования. Поэтому необходимо сделать возможным раздельное добавление элементов. В случае типа и оборудования сначала добавляется тип, а затем при создании нового оборудования используется ссылка на этот ранее созданный тип

4.3 Атрибуты

Атрибуты – это свойства отношения, характеризующие его.

В скобках указаны названия таблиц и атрибутов на английском языке, которые будут использоваться в готовой базе данных.

Для таблиц, в которых потенциальным первичным ключом является строка или отсутствует потенциальный первичный ключ, введён суррогатный числовой первичный ключ.

Используемые типы атрибутов:

1. N(n,m) – NUMERIC, числовой тип. n – количество разрядов до запятой, m – после запятой.

2. V(n) – VARCHAR, символьный тип. Это строка переменной длины. В отличие от типа CHAR, для которого оставшиеся ячейки символов заполняются пробелами, VARCAR дополнительно хранит только текущее количество символов в строке. В скобках указано максимальное количество символов. Этот тип следует использовать для достаточно длинных строк

3.BOOLEAN – логический тип. Этого типа нет в некоторых СУБД (например, в СУБД Oracle, или используемой нами СУБД Firebird), обычно вместо него принято использовать числовой тип минимальной длины, или строковый CHAR размера 1 (т.е. строка, не содержащая более одного символа).

1. Схема отношения Оборудование (MACHINERY)

|  |  |  |  |
| --- | --- | --- | --- |
| Содержание поля | Имя поля | Тип поля | Примечание |
| Наименование  модели | M\_TITLE | V(50) | уникальное обязательное поле |
| Код оборудования | M\_ID | N(10,0) | суррогатный первичный ключ |
| Описание | M\_DESCRIPTION | V(500) |  |
| Стоимость | M\_COST | N(8,2) | обязательное поле |
| Идентификатор типа | TYPE\_ID | N(5,0) | внешний ключ к MACHINERY\_TYPE |

Стоимость оборудования не тождественна сумме стоимостей составляющих его деталей и узлов. Эти параметры тоже используются для формирования окончательной стоимости, но служат прежде всего для расчёта стоимости их отдельного заказа, поэтому общая стоимость законченного агрегата указывается отдельно.

1. Схема отношения «Узел» (NODE)

|  |  |  |  |
| --- | --- | --- | --- |
| Содержание поля | Имя поля | Тип поля | Примечание |
| Наименование  модели | N\_TITLE | V(50) | уникальное обязательное поле |
| Код узла | N\_ID | N(10,0) | суррогатный первичный ключ |
| Описание | N\_DESCRIPTION | V(500) |  |
| Стоимость | N\_COST | N(8,2) | обязательное поле |

1. Схема отношения «Деталь» (DETAIL)

|  |  |  |  |
| --- | --- | --- | --- |
| Содержание поля | Имя поля | Тип поля | Примечание |
| Наименование  модели | D\_TITLE | V(50) | уникальное обязательное поле |
| Код детали | D\_ID | N(10,0) | суррогатный первичный ключ |
| Описание | D\_DESCRIPTION | V(500) |  |
| Стоимость | D\_COST | N(8,2) | обязательное поле |
| Минимальное  количество | MIN\_COUNT | N(4,0) | обязательное поле |

1. Схема отношения Чертёж (PICTURE)

|  |  |  |  |
| --- | --- | --- | --- |
| Содержание поля | Имя поля | Тип поля | Примечание |
| Чертёж | P\_LINK | V(50) | ссылка на файл, а не само изображение |
| Код чертежа | P\_ID | N(10,0) | суррогатный первичный ключ |
| Описание | P\_DESCRIPTION | V(200) |  |

Файлы можно хранить в базе данных с помощью типа BLOB –binary large object, предназначенного для хранения объектов большого размера разных типов, но этот способ имеет свои недостатки – при кэшировании (сохранении результатов) выполненного запроса объект занимает память. Кроме того, непосредственный доступ к файловой системе часто может оказаться быстрее.

1. Схема отношения *чертёж-деталь* (LNK\_PIC\_DETAIL)

|  |  |  |  |
| --- | --- | --- | --- |
| Содержание поля | Имя поля | Тип поля | Примечание |
| Код детали | D\_ID | N(10,0) | внешний ключ к DETAIL |
| Код чертежа | P\_ID | N(10,0) | внешний ключ к PICTURE |

1. Схема отношения *чертёж-узел* (LNK\_PIC\_NODE)

|  |  |  |  |
| --- | --- | --- | --- |
| Содержание поля | Имя поля | Тип поля | Примечание |
| Код узла | N\_ID | N(10,0) | внешний ключ к NODE |
| Код чертежа | P\_ID | N(10,0) | внешний ключ к PICTURE |

1. Схема отношения *чертёж*-*оборудование* (LNK\_PIC\_MACHINERY)

|  |  |  |  |
| --- | --- | --- | --- |
| Содержание поля | Имя поля | Тип поля | Примечание |
| Код оборудования | M\_ID | N(10,0) | внешний ключ к MACHINERY |
| Код чертежа | P\_ID | N(10,0) | внешний ключ к PICTURE |

1. Схема отношения *узел-деталь* (LNK\_DETAIL\_NODE)

|  |  |  |  |
| --- | --- | --- | --- |
| Содержание поля | Имя поля | Тип поля | Примечание |
| Код детали | D\_ID | N(10,0) | внешний ключ к DETAIL |
| Код узла | N\_ID | N(10,0) | внешний ключ к NODE |
| Количество  деталей в составе  узла | QUANTITY | N(2,0) | обязательное поле. |

1. Схема отношения *узел-оборудование* (LNK\_NODE\_MACHINERY)

|  |  |  |  |
| --- | --- | --- | --- |
| Содержание поля | Имя поля | Тип поля | Примечание |
| Код оборудования | M\_ID | N(10,0) | внешний ключ к MACHINERY |
| Код узла | N\_ID | N(10,0) | внешний ключ к NODE |
| Признак обязательности узла | IS\_NODE\_REQUIRED | BOOLEAN | обязательное поле |

Признак обязательности узла указывает, должен ли он непременно входить в состав оборудования, с которым связан посредством этого отношения.

1. Схема отношения Тип оборудования (MACHINERY\_TYPE)

|  |  |  |  |
| --- | --- | --- | --- |
| Содержание поля | Имя поля | Тип поля | Примечание |
| Код типа | ID\_TYPE | N(5,0) | суррогатный первичный ключ |
| Название типа | TYPE\_TITLE | V(30) | обязательное поле |

1. Схема отношения Группы пользователей (OPER\_GROUPS)

|  |  |  |  |
| --- | --- | --- | --- |
| Содержание поля | Имя поля | Тип поля | Примечание |
| Код группы | ID\_GROUP | N(5,0) | суррогатный первичный ключ |
| Название группы | TYPE\_TITLE | V(30) | обязательное поле |
| Право на редактирование | IS\_EDITOR | BOOLEAN | обязательное поле |
| Организация | COMPANY\_ID | N(5,0) | внешний ключ к COMPANY |

1. Схема отношения *группы-оборудование* (LNK\_TYPES\_GROUPS)

|  |  |  |  |
| --- | --- | --- | --- |
| Содержание поля | Имя поля | Тип поля | Примечание |
| Код группы операторов | OPER\_ID | N(5,0) | внешний ключ  к OPERATORS |
| Код типа оборудования | TYPE\_ID | N(5,0) | Внешний ключ на MACHINERY\_TYPE |

1. Схема отношения Организации (COMPANIES)

|  |  |  |  |
| --- | --- | --- | --- |
| Содержание поля | Имя поля | Тип поля | Примечание |
| Код организации | COMPANY\_ID | N(5,0) | Суррогатный первичный ключ |
| Название организации | COMPANY\_TITLE | V(30) | обязательное поле |
| Страна | COUNTRY | V(20) | обязательное поле |
| Адрес | ADDRESS | V(150) | обязательное поле |

1. Схема отношения Операторы (OPERATORS)

|  |  |  |  |
| --- | --- | --- | --- |
| Содержание поля | Имя поля | Тип поля | Примечание |
| Код оператора | OPER\_ID | N(5,0) | суррогатный первичный ключ |
| Имя оператора | NAME | V(60) | обязательное поле |
| Адрес электронной почты | EMAIL | V(20) | обязательное поле |

1. Схема отношения oper\_companies

|  |  |  |  |
| --- | --- | --- | --- |
| Содержание поля | Имя поля | Тип поля | Примечание |
| Код оператора | OPER\_ID | N(5,0) | Внешний ключ на OPERATORS |
| Код компании | COMPANY\_ID | N(5,0) | Внешний ключ на COMPANIES |

1. Схема отношения Строковый атрибут (STRING\_ATTR)

|  |  |  |  |
| --- | --- | --- | --- |
| Содержание поля | Имя поля | Тип поля | Примечание |
| Код атрибута | STRING\_ATTR\_ID | N(5,0) | суррогатный первичный ключ |
| Название атрибута | STRING\_ATTR\_TITLE | V(30) | обязательное поле |
| Значение атрибута | STRING\_ATTR\_VALUE | V(30) | обязательное поле |
| Код узла | N\_ID | N(10,0) | внешний ключ на NODE |

1. Схема отношения Числовой атрибут (NUM\_ATTR)

|  |  |  |  |
| --- | --- | --- | --- |
| Содержание поля | Имя поля | Тип поля | Примечание |
| Код атрибута | NUM\_ATTR\_ID | N(5,0) | суррогатный первичный ключ |
| Название атрибута | NUM\_ATTR\_TITLE | V(30) | обязательное поле |
| Минимльное  значение атрибута | MIN\_ATTR\_VALUE | N(5,2) | обязательное поле |
| Максимальное  Значение атрибута | MAX\_ATTR\_VALUE | N(5,2) |  |
| Единица измерения | NUM\_ATTR\_UNIT | V(10) | обязательное поле |
| Код узла | N\_ID | N(10,0) | внешний ключ на NODE |

1. Схема отношения Заказ (ORDER)

|  |  |  |  |
| --- | --- | --- | --- |
| Содержание поля | Имя поля | Тип поля | Примечание |
| Номер заказа | ORDER\_ID | N(10,0) | первичный ключ |
| Статус заказа | ORDER\_STATUS | V(15) | обязательное поле |
| Стоимость доставки | DELIVERY\_COST | N(6,2) | обязательное поле |
| Код заказчика | ID\_CLIENT | N(6,2) | внешний ключ на CLIENT |

1. Схема отношения *заказанное оборудование* (LNK\_ORDER\_MACHINERY)

|  |  |  |  |
| --- | --- | --- | --- |
| Содержание поля | Имя поля | Тип поля | Примечание |
| Код оборудования | M\_ID | N(10,0) | внешний ключ на MACHINE |
| Номер заказа | ORDER\_ID | N(10,0) | внешний ключ на ORDER |
| Серийный номер | SERIAL\_NUM | V(18) | уникальный атрибут |

Серийный номер в данном случае характеризует конкретный экземпляр оборудования.

1. Схема отношения *заказанные детали* (LNK\_ORDER\_DETAILS)

|  |  |  |  |
| --- | --- | --- | --- |
| Содержание поля | Имя поля | Тип поля | Примечание |
| Код детали | D\_ID | N(10,0) | внешний ключ на DETAIL |
| Номер заказа | ORDER\_ID | N(10,0) | внешний ключ на ORDER |

1. Схема отношения *заказанные узлы* LNK\_ORDER\_NODE

|  |  |  |  |
| --- | --- | --- | --- |
| Содержание поля | Имя поля | Тип поля | Примечание |
| Код узла | N\_ID | N(10,0) | внешний ключ на DETAIL |
| Номер заказа | ORDER\_ID | N(10,0) | внешний ключ на ORDER |

1. Схема отношения Заказчик (CLIENT)

|  |  |  |  |
| --- | --- | --- | --- |
| Содержание поля | Имя поля | Тип поля | Примечание |
| Код заказчика | CLIENT\_ID | N(5,0) | суррогатный первичный ключ |
| Наименование  заказчика | CLIENT\_COMPANY | V(30) | обязательное поле |
| Контактный e-mail | CLIENT\_EMAIL | V(20) | обязательное поле |
| Контактный телефон | CLIENT\_PHONE | V(25) |  |
| Страна | CLIENT\_COUNTRY | V(25) | обязательное поле |

1. Схема отношения *несовместимые узлы* (incompatible\_nodes)

|  |  |  |  |
| --- | --- | --- | --- |
| Содержание поля | Имя поля | Тип поля | Примечание |
| Код узла 1 | N\_ID1 | N(10) | внешний ключ на NODES |
| Код узла 2 | N\_ID2 | N(10) | внешний ключ на NODES |

Таблица несовместимых узлов содержит в себе ссылки на пары узлов, которые не могут быть установлены одновременно.

4.4. Нормализация

Нормализация – это преобразование отношений (таблиц) реляционной базы данных с целью избавления от противоречивости хранящейся информации, которая может приводить к ошибкам и аномалиям при работе с базой данных.

«Различают три вида аномалий: аномалии обновления, удаления и добавления. Аномалия обновления может возникнуть в том случае, когда информация дублируется. Другие аномалии возникают тогда, когда две и более независимые сущности объединены в одно отношение»[2]

Всего выделяют 6 видов нормальных форм. Конечной целью нормализации отношений в БД является такая ситуация, при которой в таблицах хранятся только те данные, которые невозможно вывести из других. Обычно достаточным для нормального функционирования считается приведение к 4 или 5 нормальной форме.

Проверим полученную схему БД и атрибуты её отношений на соответствие нормальным формам.

Отношение считается соответствующим первой нормальной форме в том случае, если все его атрибуты являются простыми.

В отношениях, описанных в таблицах 1-23, сложных атрибутов, агрегирующих несколько значений, нет. Таким образом, считаем, что отношения в базе данных соответствуют первой нормальной форме.

«Отношение находится во второй нормальной форме, если оно приведено к 1НФ и каждый неключевой атрибут функционально полно зависит от составного ключа» [2]. Это можно интерпретировать как то, что любая колонка таблицы определяется не частью ключа, а только ключом целиком.

В имеющейся схеме базы данных не используются составные ключи. На основании описания второй нормальной формы делаем вывод, что отношения с простыми ключами уже приведены к этой форме, а таковыми отношениями являются все, представленные выше.

Нахождение отношения в третьей нормальной форме означает отсутствие так называемых транзитивных зависимостей. Фактически это означает, что каждый неключевой атрибут должен зависеть от ключа.

Явно не удовлетворяет требованиям третьей нормальной формы отношения «Заказчик» и «Компания», в которых присутствует атрибут «Страна», а так же отношение «Числовой атрибут» и его колонка «Единицы измерения». Для нормализации необходимо вынести эти атрибуты в отдельные отношения такого вида:

1. Таблица «Страна» (COUNTRY)

|  |  |  |  |
| --- | --- | --- | --- |
| Содержание поля | Имя поля | Тип поля | Примечание |
| Идентификатор страны | ID\_COUNTRY | N(5,0) | суррогатный первичный ключ |
| Название страны | COUNTRY\_NAME | V(25) | обязательное уникальное поле |

Соответственно, в таблицах «Заказчик» и «Компания» в соответствующих столбцах будет содержаться внешний ключ к отношению «Страна», что позволит избежать аномалий обновления и записи.

Аналогично для отношения «Числовой атрибут» выносится отдельно «Единица измерения»:

1. Отношение «Единица измерения» (UNIT\_OF\_MEASURE)

|  |  |  |  |
| --- | --- | --- | --- |
| Содержание поля | Имя поля | Тип поля | Примечание |
| Идентификатор единицы измерения | UNIT\_ID | N(5,0) | суррогатный первичный ключ |
| Название единицы измерения | UNIT\_TITLE | V(30) | обязательное уникальное поле |

После произведённой декомпозиции можно считать, что отношения в базе данных приведены к третьей нормальной форме.

Соответствующим четвёртой нормальной форме считается то отношение, в котором независимые друг от друга, но зависимые от ключа неключевые атрибуты отсутствуют. Такие атрибуты должны быть вынесены в отдельные отношения.

В нашей базе данных таких атрибутов отношений не выявлено, следовательно считаем, что отношения соответствуют четвёртой нормальной форме. Это соответствие считается достаточным для сохранения приемлемой сложности логической организации базы данных. В то же время соответствие этой форме означает уменьшение количества аномалий удаления, обновления и записи, что в свою очередь даёт право быть уверенными в логической целостности данных.

# ГЛАВА 5: ОРГАНИЗАЦИЯ ПОЛЬЗОВАТЕЛЬСКОГО ИНТЕРФЕЙСА

Сложное оборудование собирается на заказ и состоит из множества узлов и компонентов. Сходным случаем можно считать различные комплектации автомобилей (базовая, премиум и т.д.), но комплектация оборудования для строительных работ более различна, так как оно выпускается для более узких областей применения. В связи с этим каталоги запчастей и комплектующих для оборудования этого рода собой довольно обширный общий список, содержащий спецификации и чертежи, не всегда понятный конечному пользователю.

Разрабатываемая система обработки заказов позволит набирать из имеющихся объектов готовую комплектацию. При заказе запчастей это позволит сопоставлять конкретному устройству набор доступных для него деталей.

Каждое устройство состоит из некоторого числа узлов. Часть узлов является базовыми, т.к. посредством них формируется основное предназначение устройства, часть – опциональными. Некоторые узлы также могут быть взаимоисключающими. В свою очередь каждый узел может содержать различный набор компонентов.

Таким образом, получается трёхуровневая модель оборудования – на уровне деталей, на уровне совокупности деталей, формирующих узел, и на уровне единицы оборудования как совокупности узлов. После первичного заполнения базы данных станет возможным быстро формировать общий вид оборудования на основании ранее внесённой информации о составе узлов.

Также необходимо предусмотреть возможность ведения отчётности по изменениям в комплектации оборудования, для отслеживания изменений с конкретной конфигурацией и возможности исправления возможных ошибок путём возвращения к предыдущей версии комплектации. В частности, для этого отслеживается, какой оператор и в какое время сделал последнее изменение.

Доступ к системе будет осуществляться посредством web-интерфейса.

Это позволит реализовать все функции в рамках одного приложения без отдельной установки клиентской части (в её роли фактически выступает браузер), что значительно упрощает работу с системой для конечного пользователя, так как технические требования к аппаратуре клиента аналогичны таковым для браузера. Пользователь не будет привязан к какому-либо определённому устройству.

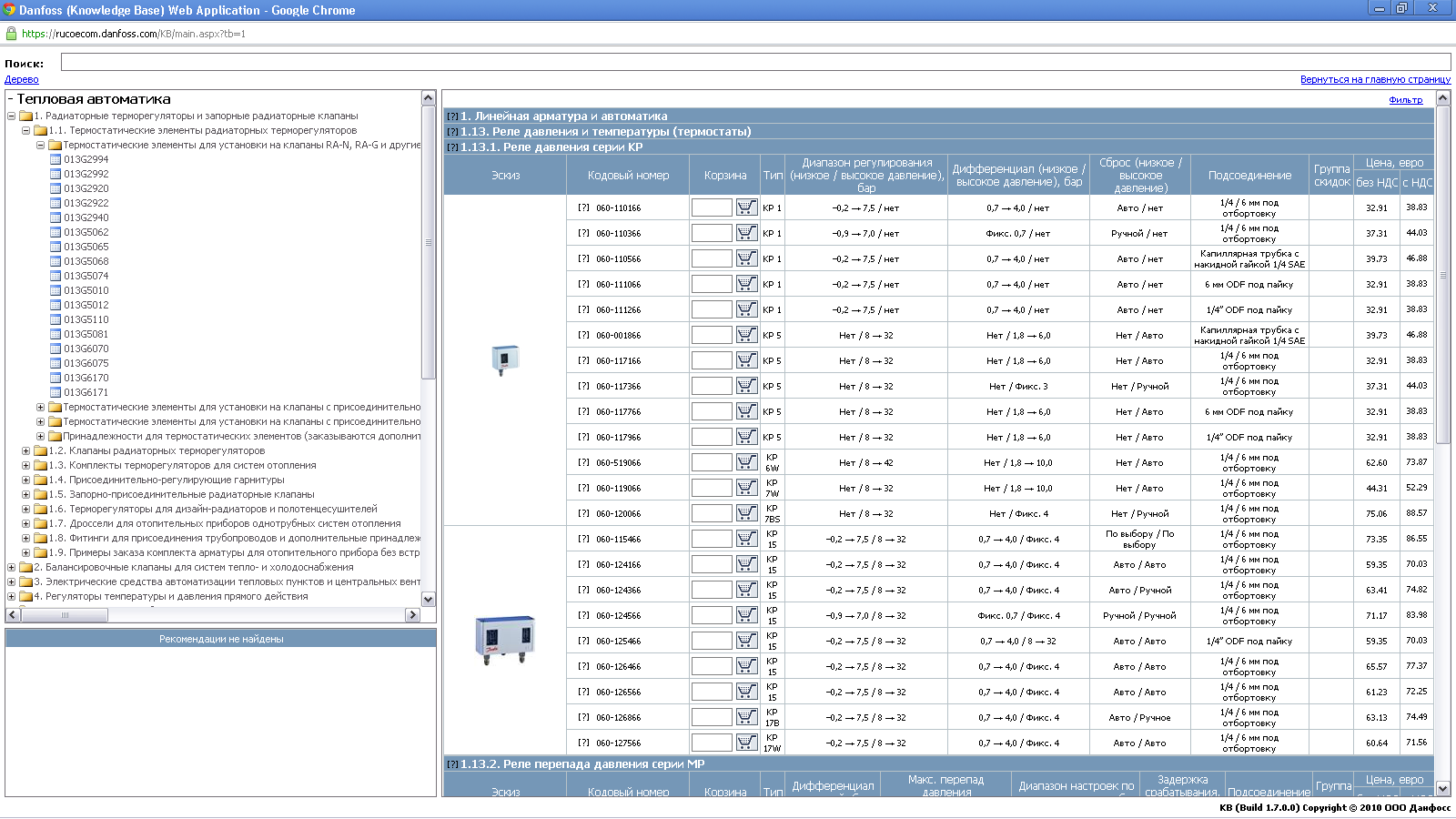
Для специалистов, занятых продажей оборудования это открывает возможность иметь доступ к системе из любого места, где есть подключение к сети internet.

Пользовательский интерфейс разделяется на несколько частей – административную и клиентскую. С помощью административной части операторы компаний-поставщиков вносят записи об оборудовании, а с помощью клиентской осуществляется просмотр списков оборудования, узлов и деталей. С помощью этого интерфейса необходимый элемент можно внести в заказ. Если у пользователя уже имеется какое-либо оборудование, он сможет просмотреть списки деталей и узлов, относящихся конкретно к его экземпляру.

5.1. Аналоги

**Danfoss Web Application**

Веб-приложение, используемое компанией Danfoss, производящей оборудование для теплоснабжения.

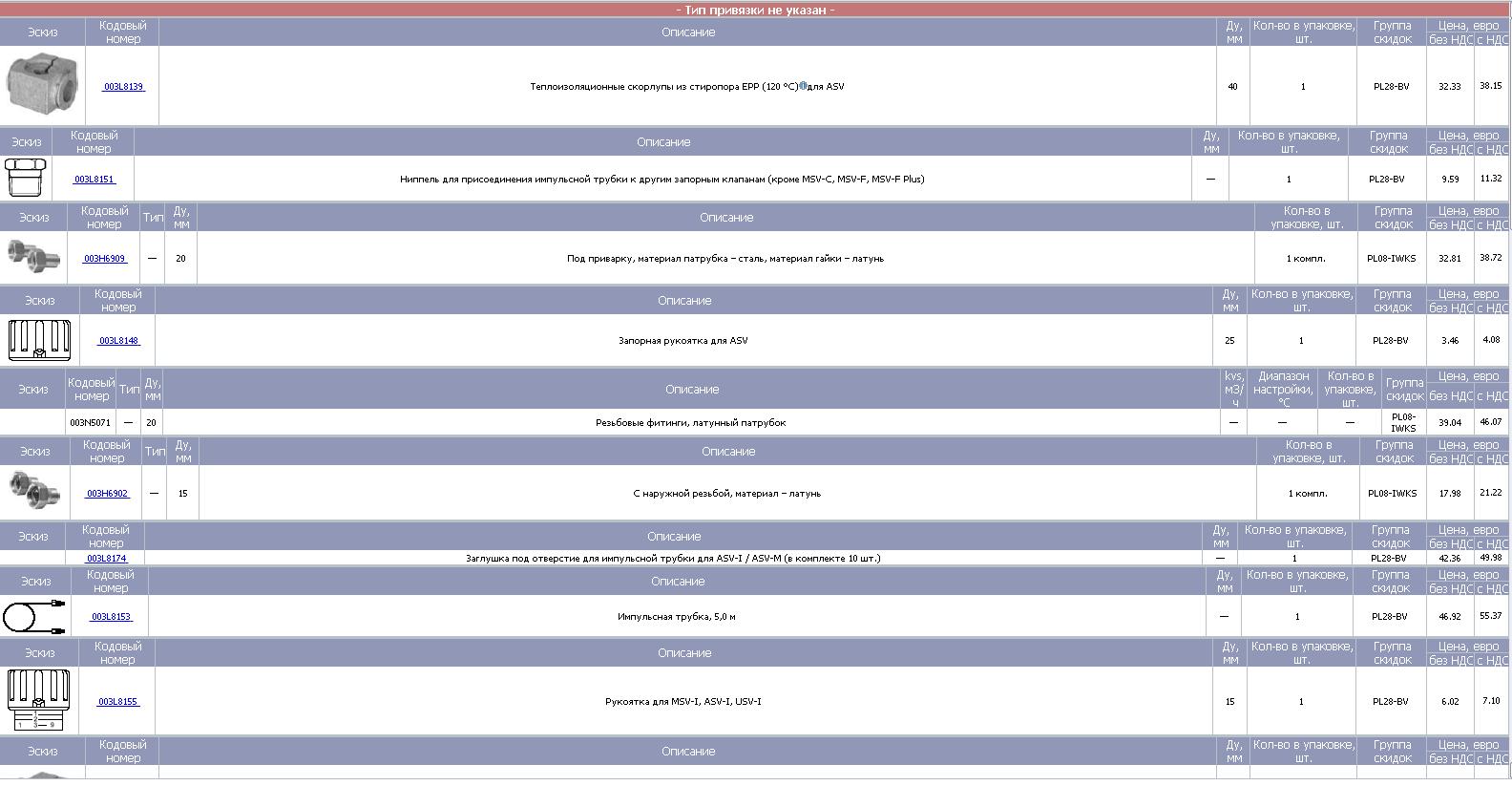


*Рис.3 Общий вид интерфейса*

Оборудование можно найти с помощью встроенного поиска, или дойдя до нужного пункта в древовидном списке. На самом нижнем уровне этого списка открывается либо спецификация (рис.3) с возможностью фильтрации по параметрам оборудования, либо pdf-файл, представляющий из себя отдельный каталог. Также есть возможность отслеживания связи между несколькими продуктами. Связанные объекты можно просмотреть на отдельной странице (рис.4).

Вид общего списка (и, соответственно, фильтр) меняется в зависимости от выбранного типа оборудования.

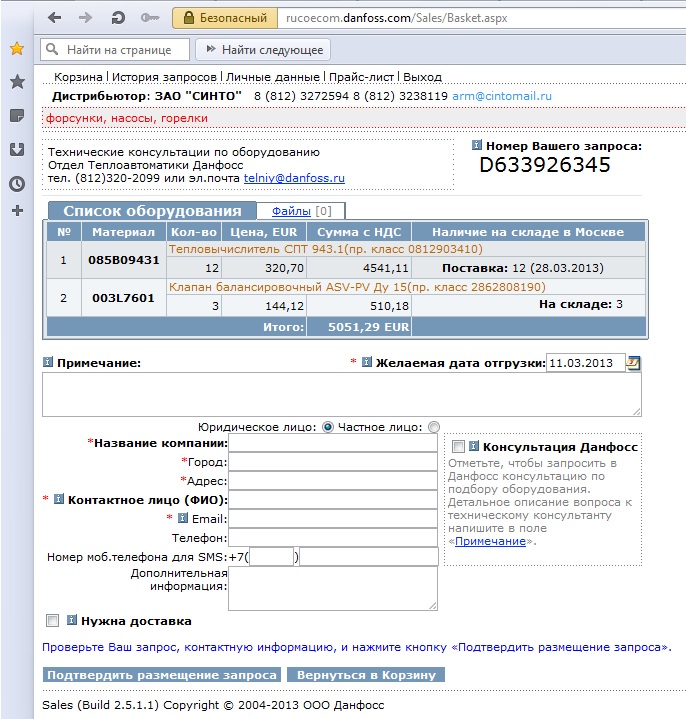
Чертежи и подробное описание оборудования находятся в отдельном файле. Чтобы перейти к конкретному файлу, нужно раскрыть древовидный список до нижнего уровня иерархии.



*Рис. 4 Связи между объектами*

Список связей отражает вхождение выбранного оборудования в комплекты поставки, с указанием типа связи. Каждый связанный элемент отображается в виде строки из соответствующей данному типу таблицы.

Табличное представление разнородных данных вносит некоторую путаницу в отображении.

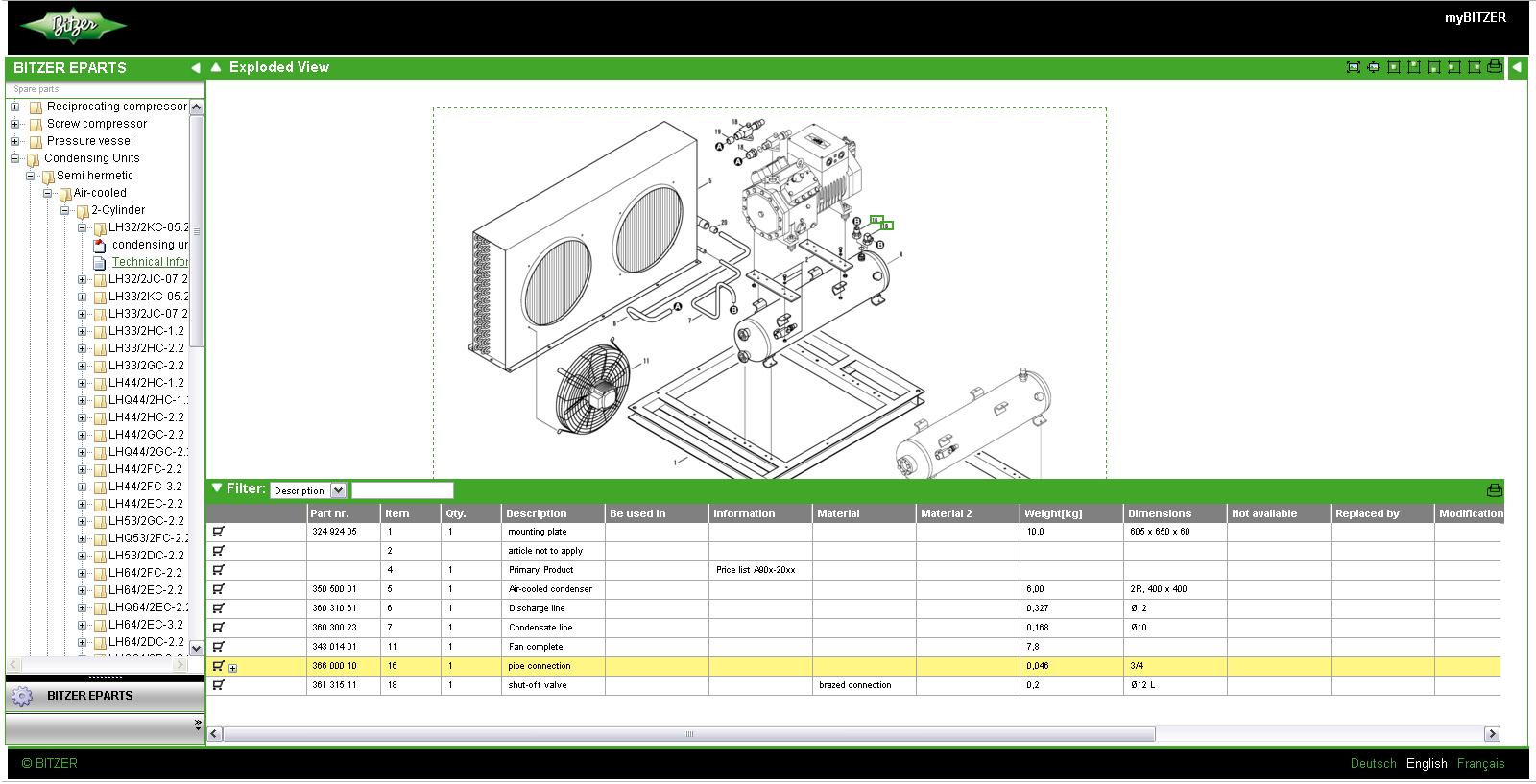


*Рис.5 Форма заказа*

Идентификаторы оборудования вносятся в форму заказа автоматически, пользователю предоставляется информация о наличии оборудования на складе или о возможных сроках поставки оборудования

**BITZER EPARTS**

Интерфейс того же типа, что и в Danfoss Web Application, но отсутствует поиск по всей продукции. Выбор оборудования осуществляется также путём поиска по древовидному списку.



*Рис.6 Интерфейс Bitzer Eparts*

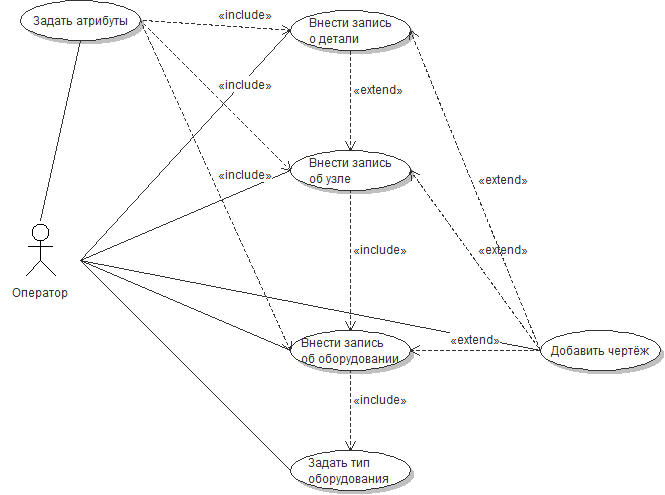
Имеется возможность просмотра чертежа оборудования в увеличенном масштабе и фильтрации списка деталей.

В остальном функционал аналогичен Danfoss Web Application, за исключением отсутствия общего поиска по всем видам оборудования.

Это приложение обладает удобным интерфейсом для просмотра данных по спецификации узла.

Bitzer Epart представляет собой java-апплет, поэтому не может функционировать в браузере с отключенной поддержкой java, и работает значительно медленнее описанного выше Danfoss Web Application.

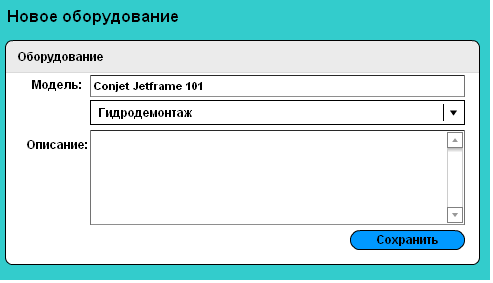
5.2. Интерфейс оператора.



*Рис.7 Диаграмма вариантов использования(Use Case Diagram) для оператора*

На верхнем уровне комплектации оператор создаёт информационный объект, соответствующий модели оборудования, внося в базу его первоначальное описание – название модели, описание, тип, стоимость.

Остальные параметры задаются после сохранения объекта в базе данных для обеспечения возможности ссылаться на только что созданное оборудование.



*Рис.8 Общий вид интерфейса для добавления нового оборудования.*

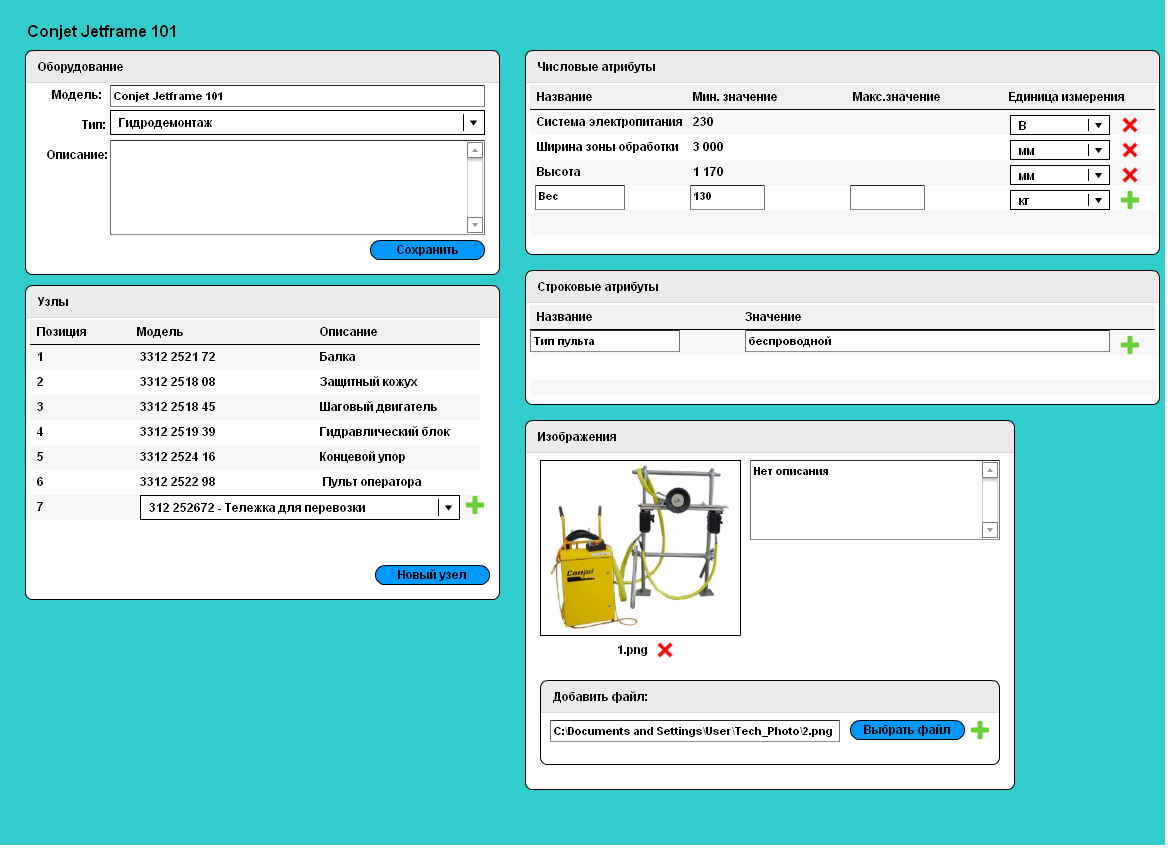
Также для оборудования задаются атрибуты – числовые или строковые. Атрибуты имеют вид «название-значение», для числовых атрибутов в этом случае добавляется единица измерения - килограммы, метры, давление и так далее. С помощью этих атрибутов можно задать параметры, которые не зависят от необязательных узлов оборудования – например, линейные размеры машины – длину, ширину и высоту.

К оборудованию можно прикреплять файлы – чертежи и фотографии.

После сохранения нового объекта оператор может добавлять к нему связанные элементы.

Различные группы элементов размещены на отдельных панелях. Возможность перемещать и сворачивать панели помогает оператору настроить вид интерфейса удобным для него образом.

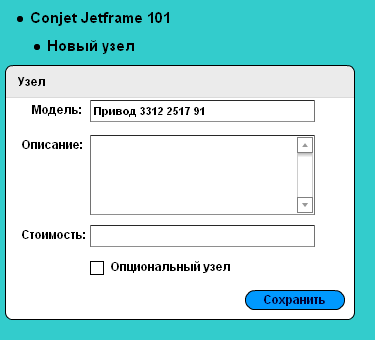
Списки элементов представлены в виде таблиц. В случае добавления узлов оператор может как выбрать один из ранее созданных, так и добавить новый. При отсутствии элементов в списке на панели отображается кнопка «Добавить». При нажатии этой кнопки в списке атрибутов появляется новая строка, которую можно внести в список после заполнения, нажав на кнопку «+».



*Рис.9 Интерфейс редактирования оборудования.*

Загрузка файлов осуществляется путём поиска расположения этого файла в доступных пользователю директориях Выбрав файл, оператор нажимает на кнопку «+». После этого изображение загружается на сервер, путь к нему записывается в базу данных и связывается с объектом, к которому относится это изображение.

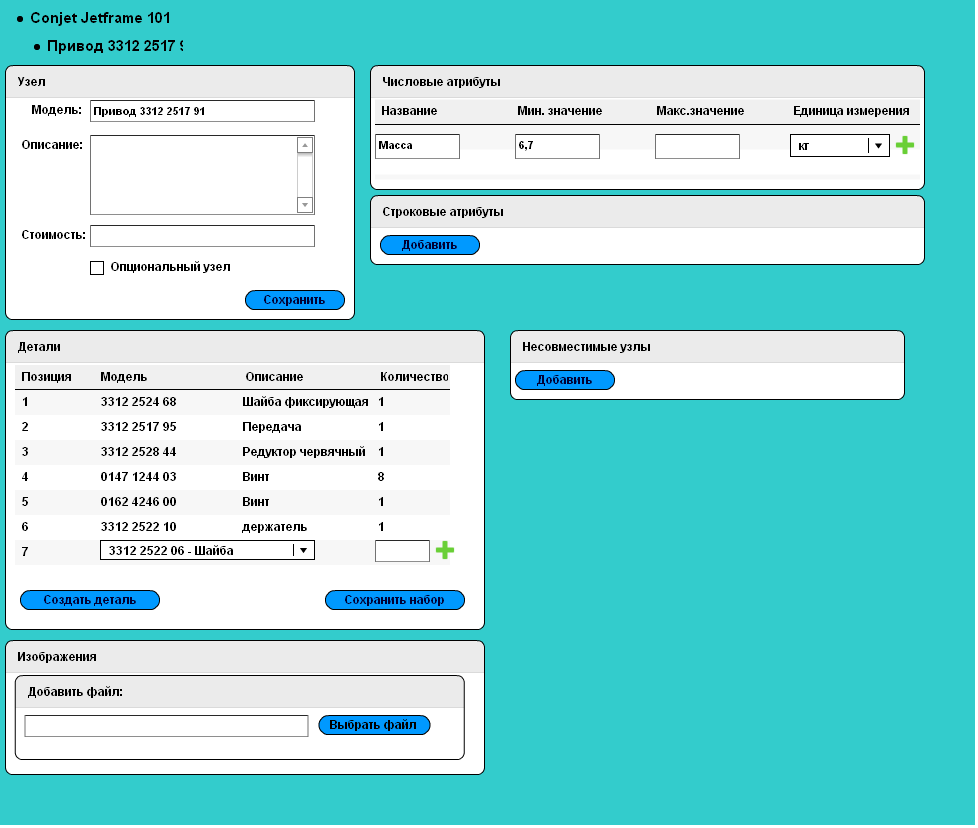
Далее оператор может добавить узлы, из которых состоит данная модель. Оператор может выбрать ранее описанный узел для оборудования того же типа, или же внести запись о новом узле. В первом варианте оператор просто выбирает узел из предложенного списка, добавляя его так же как атрибут в ранее рассмотренном примере, во втором – переходит к созданию нового узла.



*Рис.10 Создание нового узла*

Для узла также вводится его наименование и стоимость.

После сохранения узла и привязки его к какой-либо модели оборудования, оператор имеет возможность выбрать статус узла – обязательный или не обязательный. Также имеется возможность сделать несколько узлов обязательными, но взаимоисключающими. Для этого в свойствах узла необходимо указать те узлы, которые являются несовместимыми с ним. К узлу также можно прикрепить описывающие его изображения – чертежи и фотографии.

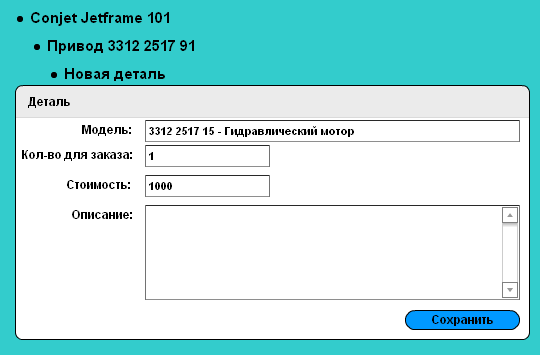


*Рис.11 Интерфейс редактирования узла*

Для упрощения формирования узла оператор может просмотреть состав других узлов, относящихся к тому же типу оборудования, и добавить в комплектацию узла детали, из которых они состоят.

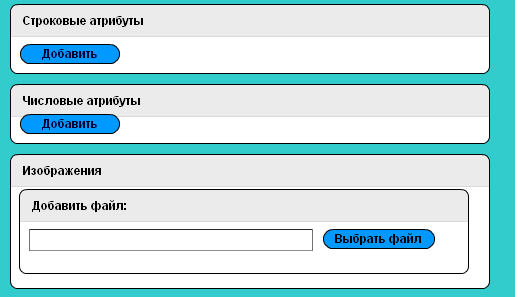
Далее в рамках каждого узла формируется набор деталей, из которых он состоит. Работа на этом, более низком уровне, в принципе аналогична предыдущему. Задаётся название, стоимость и минимальное количество для заказа.

Отображение детали или узла как зависимой сущности не означает её привязки к строго к конкретному узлу. При удалении связанного узла деталь остаётся в базе данных.



*Рис.12 Интерфейс создания детали. Деталь не привязана к узлу*

При связывании детали с узлом указывается количество этих деталей, используемое в данном узле. Деталь также имеет собственные атрибуты – как минимум линейные размеры или краткое описание, но они не являются обязательными. Для деталей также возможно добавление чертежей.



*Рис.13 Интерфейс редактирования детали, добавление атрибутов и изображений*

После первичного заполнения базы данных оператор получает различные наборы узлов, из которых потом может достаточно быстро собрать готовую комплектацию. Для этого используется функция создания нового объекта на основе существующего оборудования – например, если для робота, выполняющего работы по гидродемонтажу требуется отделить модель, в комплект поставки которой входит водомёт другой мощности, пользователь вносит информацию только об одном новом узле, а остальные остаются прежними.

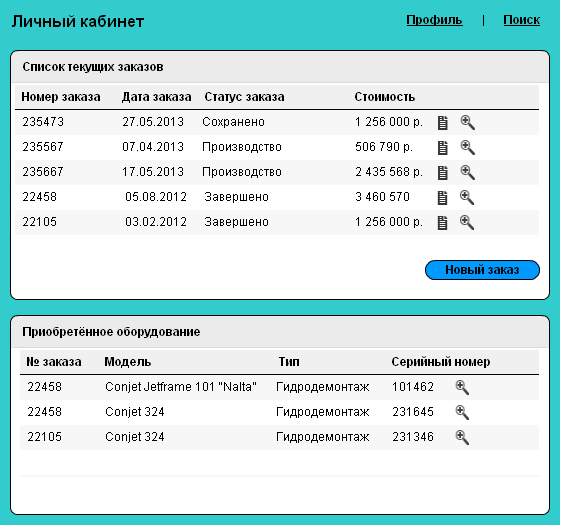
Изменения в базе данных, произведённые оператором, будут использоваться для построения отчётности – когда и кем вносилась запись о конкретном узле, информация об использования узлов/деталей в оборудовании с различными серийными номерами.

5.3. Интерфейс пользователя

Для доступа к личному кабинету пользователь должен зарегистрироваться, но для просмотра информации об оборудовании ему будет достаточно ввести серийный номер в строку поиска, без обязательной авторизации.

Пользователь имеет возможность просматривать списки оборудования и его комплектующих и вносить их в заказ. У пользователя есть свой личный кабинет, в котором хранится:

1. список совершённых заказов с указанием статуса (выполнен, ожидание)
2. список сохранённых, но ещё не совершённых заказов в виде предварительно выбранного комплекта оборудования
3. список оборудования, владельцем которого является пользователь.



*Рис.14 Личный кабинет со списками заказов и оборудования*

В общих списках оборудования, отобранных по фильтрам, пользователь может добавить само оборудование и компоненты к заказу.

При наборе в «корзину» у пользователя есть выбор – сделать заказ, сохранить для последующей доработки или очистить корзину.

При просмотре текущих и выполненных заказов пользователь может перейти к просмотру детального описания оборудования (с возможностью скрывать неиспользуемые элементы интерфейса), в который входят:

1. чертежи
2. спецификация деталей по узлам с указанием обязательности данного узла
3. технические характеристики оборудования

Интерфейс просмотра в целом по внешнему виду аналогичен интерфейсу оператора, с поправкой на невозможность редактирования каких-либо данных. Пользователь может добавлять в заказ отдельные детали и узлы оборудования.

Из данных о выполненных заказах формируется список имеющегося у пользователя оборудования, который используется для подбора деталей и узлов к определённому агрегату.

Помимо отображения на странице заказов, с которой пользователь может перейти на описание конкретного заказа, приложением генерируется как документ на основании выбранных для заказа деталей и узлов, который может быть загружен и распечатан.

Использование отдельных панелей, в которые включены функциональные блоки, хорошо проявило себя в различных классах программ – например, в системах электронного документооборота. Пользователь постоянно имеет доступ к большому объёму информации, но способен манипулировать этим представлением согласно своим предпочтениям.

Такой способ организации интерфейса позволяет компактно разместить большой объём данных без ущерба количеству предоставляемой информации, так как любая скрытая информация может легко быть сделана вновь доступной. Если заголовки блоков достаточно точно передают их содержимое, пользователь может быстро и с небольшим количеством ошибок переключаться между различными блоками.

Также пользователь не испытывает давление от большого количества информации, которое вынужден воспринимать единовременно – ему достаточно скрыть ненужные в данный момент блоки. Так как разрабатываемое приложение не относится к классу диспетчерских программ и не использует каких-либо сигналов, которые требуют от оператора немедленного обращения внимания, такое скрытие блоков не приносит существенного вреда при работе с программой. Для напоминания оператору о не выполненных действиях достаточно вывести ему сообщение о незаполненных полях.

Так как в программе отсутствует необходимость в сложной обработке данных, и её цель состоит более в оптимизации хранения и отображения этих данных, ошибки пользователя не влекут за собой какие-либо нарушения бизнес логики, а логическую целостность данных достаточно надёжно контролирует система управления базами данных и средства приложения, занимающиеся валидацией (проверкой) вводимых пользователем значений.

ЗАКЛЮЧЕНИЕ

Автоматизация работы во многих областях человеческой деятельности сталкивается с проблемами. В частности, производители мелкосерийного высокотехнологичного оборудования обычно не имеют возможности содержать в штате отдел разработки программного обеспечения, автоматизирующего бизнес-процессы компании, а заказ системы конкретно под свои нужды зачастую бывает достаточно дорогим.

В аспекте обработки заказов и формировании каталогов оборудования и запчастей желательно задать единые принципы представления и обработки информации, с целью дальнейшего использования несколькими предприятиями, при условии, что производимую продукцию можно представить в рамках определённой модели.

В ходе анализа и построения модели предметной области было выявлено, что различные виды сложного оборудования можно свести к единому иерархическому представлению. Такое представление корректно воспроизводится с помощью реляционной модели данных, широко используемой в различных областях информационных технологий.

Разбиение сложносоставного оборудования на отдельные элементы, формирующие его, позволяет реализовать программу-конструктор. С помощью программы такого вида производитель имеет возможность быстро формировать информационное представление новых моделей оборудования, используя готовые элементы. В свою очередь такое разделение оказывается удобным и для заказчика, который без проблем может установить соответствие между имеющимся у него оборудованием и доступными запчастями.

Для улучшения взаимодействия заказчика и производителя был проведен анализ бизнес-процесса заказа оборудования и формирования набора поставки для нужд заказчика. По результатам этого анализа будет целесообразной реализация единого программного комплекса, сочетающего в себе как интерфейс для получения данных заказчиком, так и интерфейс-конструктор для производителя.

Наиболее удобным для обеих сторон (и возможных посредников) является поставка такого программного комплекса в виде веб-приложения. В частности, такая реализация избавляет обе стороны от процесса непосредственной установки программного обеспечения, и предъявляет к пользователю только одно требование – подключение к сети Интернет. Пользователь имеет возможность получать доступ к системе с разных видов устройств, и при использовании системы всегда имеет доступ к актуальной информации.

Таким образом, предлагаемая реализация системы обработки заказов позволяет существенно упростить взаимодействие производителей и заказчиков, обеспечивает быстрый обмен информацией о каких-либо изменениях (в отличие от традиционных каталогов оборудования и запчастей) и позволяет отображать и хранить информацию в виде, удобном для обеих сторон процесса.

# Список литературы

1. [**http://ru.wikipedia.org/wiki/**](http://ru.wikipedia.org/wiki/)
2. Карпова И.П. Введение в базы данных. Учебное пособие. – Московский Государственный институт электроники и математики. – М.: 2005. – 75 с.
3. http://www.firebirdsql.org/