< **Межконтинентальный конвейер средств индивидуального перевоплощения «Иллюзия»**>

Software Architecture Document

Version <1.9>

Revision History

|  |  |  |  |
| --- | --- | --- | --- |
| **Date** | **Version** | **Description** | Author |
| <20/03/14> | <0.1> | Создание документа | Кобцев С.В. |
| <27/03/14> | <0.2> | Добавление таблицы соотношения типов диаграмм и слоев | Грудина А.М. |
| <04/06/14> | <0.3> | Добавление структуры Use Case View и описаний для добавленных блоков и блоков «Введение» | Грудина А.М. |
| <05/06/14> | <0.4> | Добавление логических уровней Добавление диаграммы прецедентов и описания диаграмм Добавление диаграмм базы данных для Use Case View и Implementation View | Гапонов А.И. Грудина А.М. Кобцев С.В. |
| <06/06/14> | <0.5> | Добавление диаграммы состояний для ключевой сущности «Заказ» («Order») в Use Case View Изменение ссылок на другие источники в References Добавление информации о реализации прецедентов в Use Case Realization Добавлено описание в Logical View Изменение Use Case Diagram | Грудина А.М. |
| <11/06/14> | <0.6> | Изменена диаграмма слоев | Гапонов А.И. |
| <11/06/14> | <0.7> | Добавлена крупноблочная диаграмма слоев и общее описание Logical View | Кобцев С.В. |
| <11/06/14> | <0.8> | Изменение диаграммы состояний для статусов сущности «Заказ»  Добавлено соотношение состояний и статусов «Заказов». | Грудина А.М. |
| <13/06/14> | <0.9> | Изменена диаграмма прецедентов | Грудина А.М. |
| <13/06/14> | <1.0> | Маркировка несделанных частей  Добавление подразделов для соответствующих диаграмм в представлениях  Добавлена диаграмма состояний для запроса на изменение состояния заказа  Добавление диаграммы деятельности для прецедента «Создание заказа» | Грудина А.М. |
| <13/06/14> | <1.1> | Добавлены все диаграммы классов для трех представлений  Добавлено описание слоев системы | Гапонов А.И. |
| <13/06/14> | <1.2> | Добавлено описание некоторых разделов документа Изменения в диаграммах классов | Грудина А.М.  Гапонов А.И. |
| <14/06/14> | <1.3> | Добавлена диаграмма последовательностей в Use Case View | Грудина А.М. |
| <14/06/14> | <1.4> | Добавлена диаграмма коммуникации в Use Case View | Грудина А.М. |
| <14/06/14> | <1.5> | Добавлена диаграмма деятельности в Logical View  Изменена диаграмма прецедентов | Грудина А.М. |
| <16/06/14> | <1.6> | Корректировка всех имеющихся диаграмм | Грудина А.М. |
| <19/06/14> | <1.7> | Удалена Timeline Diagram секция за ненадобностью | Грудина А.М. |
| <24/06/14> | <1.8> | Добавление диаграммы пакетов в Logical View  Добавление диаграмм последовательностей и взаимодействия в Implementation View Добавлена диаграмма развертывания в Deployment View  Добавление диаграммы последовательностей и взаимодействия в Logical View | Кобцев С.В.  Грудина А.М |
| <24/06/14> | <1.9> | Добавление диаграммы состояний в Implementation View  Добавление диаграммы деятельности в Implementation View | Грудина А.М.  Кобцев С.В. |

Table of Contents

1. Introduction 6

1.1 Purpose 6

1.2 Scope 6

1.3 Definitions, Acronyms, and Abbreviations 7

1.4 References 7

1.5 Overview 7

2. Architectural Representation 7

3. Architectural Goals and Constraints 8

4. Use-Case View 8

4.1 Use Case Diagrams 8

4.2 Activity Diagram for creating order by townsman 10

4.3 Sequence Diagrams for changing warehouse info about collected nettles 11

4.4 Cooperation Diagram for writing instruction (description) for good by seamstress 12

4.5 State Machine diagram for key entity “Order” 12

4.6 Preliminary Class Diagram 14

4.7 Data Base Diagram 15

4.8 Use-Case Realizations 16

5. Logical View 17

5.1 Overview 17

5.2 Architecturally Significant Design Packages 17

5.3 Detailed Class Diagram for key-value entities 18

5.4 Package Diagram 19

5.5 Activity Diagram for suspension order running by analyst 20

5.6 State Machine Diagram for changing progress of the each order state of “In Progress” type 21

5.7 Sequence Diagram for searching order in the analytic view 22

5.8 Cooperation Diagram 23

6. Deployment View 24

6.1 Deployment Diagram 24

7. Implementation View 25

7.1 Overview 25

7.2 Layers 25

7.3 Activity Diagram for getting recipe of the good 25

7.4 State Machine Diagram for changing quality description by wizard 26

7.5 Sequence Diagram for uploading predefined values for goods in database 27

7.6 Cooperation Diagram for setting size for the good in the system 28

7.7 Class Diagram (full version) 29

7.8 Data Base Diagram 29

8. Data View 32

9. Size and Performance 32

10. Quality 32

Software Architecture Document

# Introduction

Введение описывает назначение данного документа с точки зрения цели, масштаба работы, определений, акронимов, аббревиатур, ссылок.

## Purpose

Этот документ обеспечивает всесторонний архитектурный обзор системы МКСИД «Иллюзия» для отображения различных аспектов системы. Данный документ предназначен для получения и передачи архитектурных решений, которые были приняты в системе.

Чтобы изобразить программное обеспечение максимально точно, структура этого документа основывается на модели “4+1 представление” [KRU41].



Рисунок1 - "4+1 Views" Model

## Scope

Данный документ должен отобразить архитектуру программного обеспечения системы «Иллюзия», разрабатываемой компанией «Крафт-Девелопмент».

Документ описывает те аспекты системы «Иллюзия», которые являются архитектурно значимыми; т.е. те элементы и их поведения, которые являются основной частью системы и необходимы для понимания системы в целом.

## Definitions, Acronyms, and Abbreviations

**RUP**: Rational Unified Process

**UML:** Unified Modeling Language

**SAD:** Software Architecture Document

**WWW**: World Wide Web

## References

[SRS]: Software Requirements Specification

[MedBiquitous]: Sample SAD, <http://medbiq.org/std_specs/techguidelines/softwarearchitecture.pdf>

[KRU41]: The “4+1” view model of software architecture, Philippe Kruchten, November 1995, <http://www3.software.ibm.com/ibmdl/pub/software/rational/web/whitepapers/2003/Pbk4p1.pdf>

[USECASES]: Use Case документ, описывающий поведений всей системы в целом, и набор документов, описывающий детально каждый из use cases.

## Overview

Данный документ состоит из описания 4+1 представлений, которые в совокупности описывают все значимые архитектурные аспекты разрабатываемой системы. Структуру документа смотрите вначале документа в оглавлении.

# Architectural Representation

Данный документ описывает архитектуру системы с помощью 5 представлений: Use-Case View, Logical View, Process View, Deployment View и Implementation View, которые выполнены в форме соответствующих UML-диаграмм. Соотношение представлений и диаграмм представлено в таблице ниже.

Таблица 1 - Соотношение представлений и диаграмм

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Diagram\View | Use Case View | Logical View | Implementation view | Process View | Deployment View |
| Use Case Diagram | 100% | - | - | - | - |
| Class Diagram | 30% (class analyze) | 60% (detailed class analyze) | 100% | - | - |
| Activity Diagram | 20% | 50% | 100% | - | - |
| Timeline Diagram | - | - | - | 100% (0%\*) | - |
| State Machine Diagram | + | + | + | - | - |
| Deployment Diagram | - | - | - | - | 100% |
| Sequence Diagram | + | + | + | - | - |
| Cooperation Diagram | + | + | + | - | - |
| Package Diagram | - | 100% | - | - | - |
| Data Base Diagram | 100% | - | 100% (detailed) | - | - |

\*Делается только, если в системе есть элементы, которые жестко зависят от времени.

# Architectural Goals and Constraints

В этом разделе описываются требования к программному обеспечению и задачи, которые имеют существенное влияние на архитектуру разрабатываемой системы.

Требования, которые являются ключевыми для архитектуры системы:

1. Система должна предоставлять различные функциональные возможности для различных групп пользователей.
2. Система должна обеспечивать, чтобы все запросы, созданные пользователями, немедленно передавались следующим участникам процесса и обновлялись связанные аналитические таблицы. [SRS]
3. Система должна быть доступна для рабочего персонала для использования и обслуживания - семь дней в неделю по 8 часов каждый день, исключения составляют профилактические дни (1 раз в месяц).[SRS]
4. Система должна предоставлять валидные на текущий момент времени данные о статусе заказа без задержек. [SRS]
5. Система должна иметь модуль для печати, необходимый для обеспечения возможности пользователю печатать описание своего заказа вне зависимости от его состояния и возможности аналитикам печатать аналитические таблицы.
6. Система должна представлять из себя веб-портал, поддерживающий работу во всех популярных браузерах (Chromeи Safari).

# Use-Case View

В данном разделе будут представлены диаграммы прецедентов, диаграммы деятельности, диаграммы последовательностей, диаграммы коопераций и взаимодействий, предварительная диаграмма классов и диаграмма состояний ключевой сущности системы – «Заказ», которые предназначены для описания основных сценариев использования системы различными пользователями и базового описания сущностей системы.

## Use Case Diagrams

Этот тип диаграмм описывает возможные варианты работы с системой различными группами пользователей.

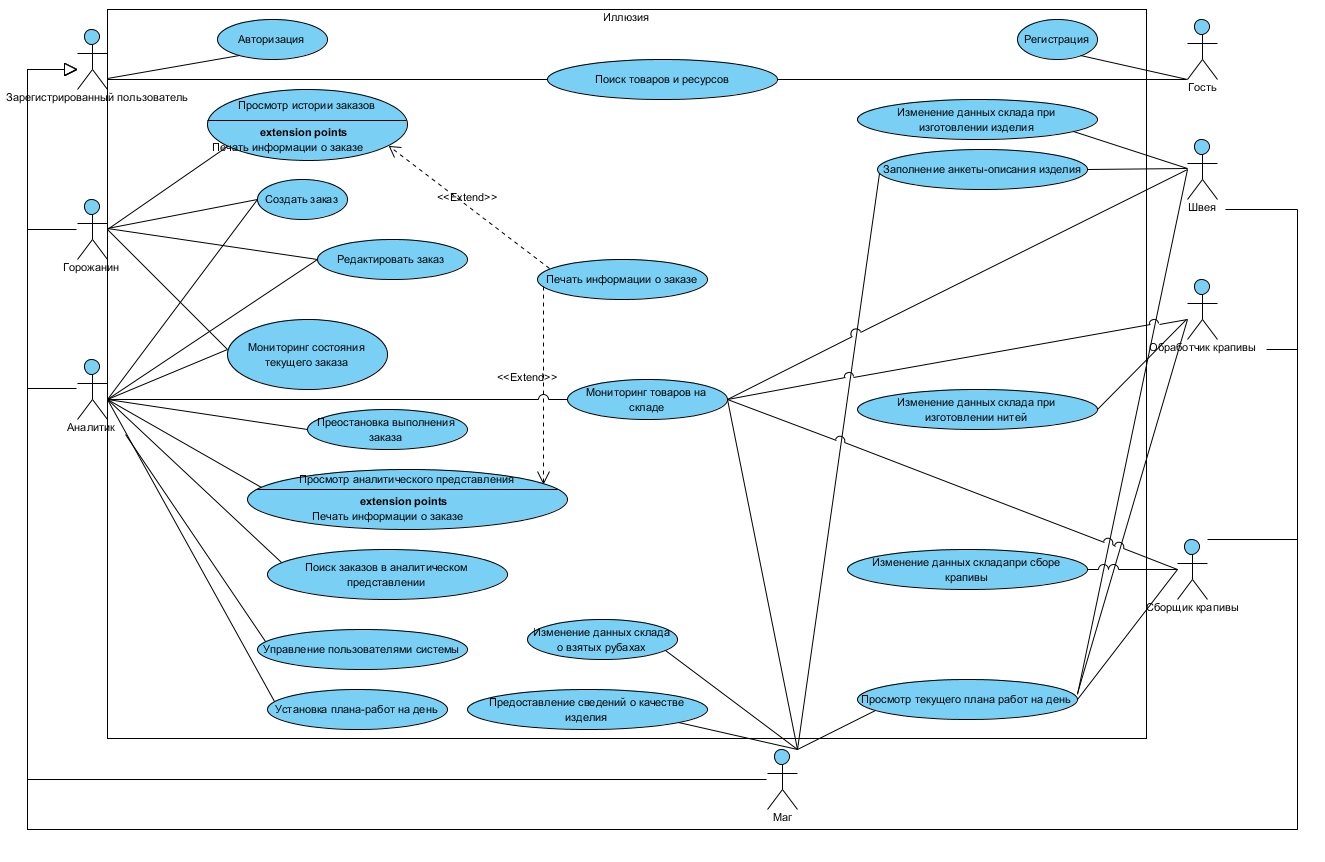


Рисунок2 - Use-Case Diagram

## Activity Diagram for creating order by townsman

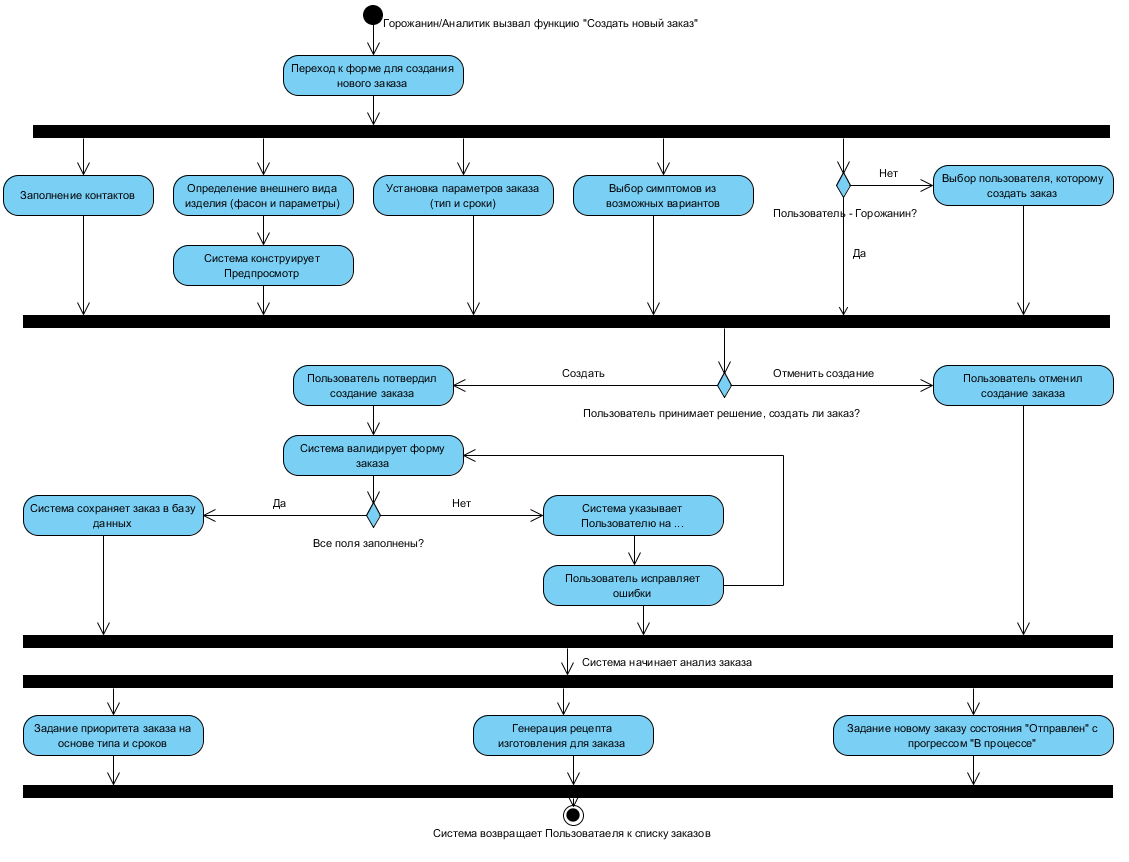


Рисунок 3 - Activity Diagram for creating order by townsman

Эта диаграмма описывает последовательность действий Горожанина и Системы для создания заказа. В описание входит также отмена создания и валидация формы.

## Sequence Diagrams for changing warehouse info about collected nettles

Данная диаграмма описывает вариант использования системы сборщиком крапивы – «Изменение информации склада о собранной крапиве».

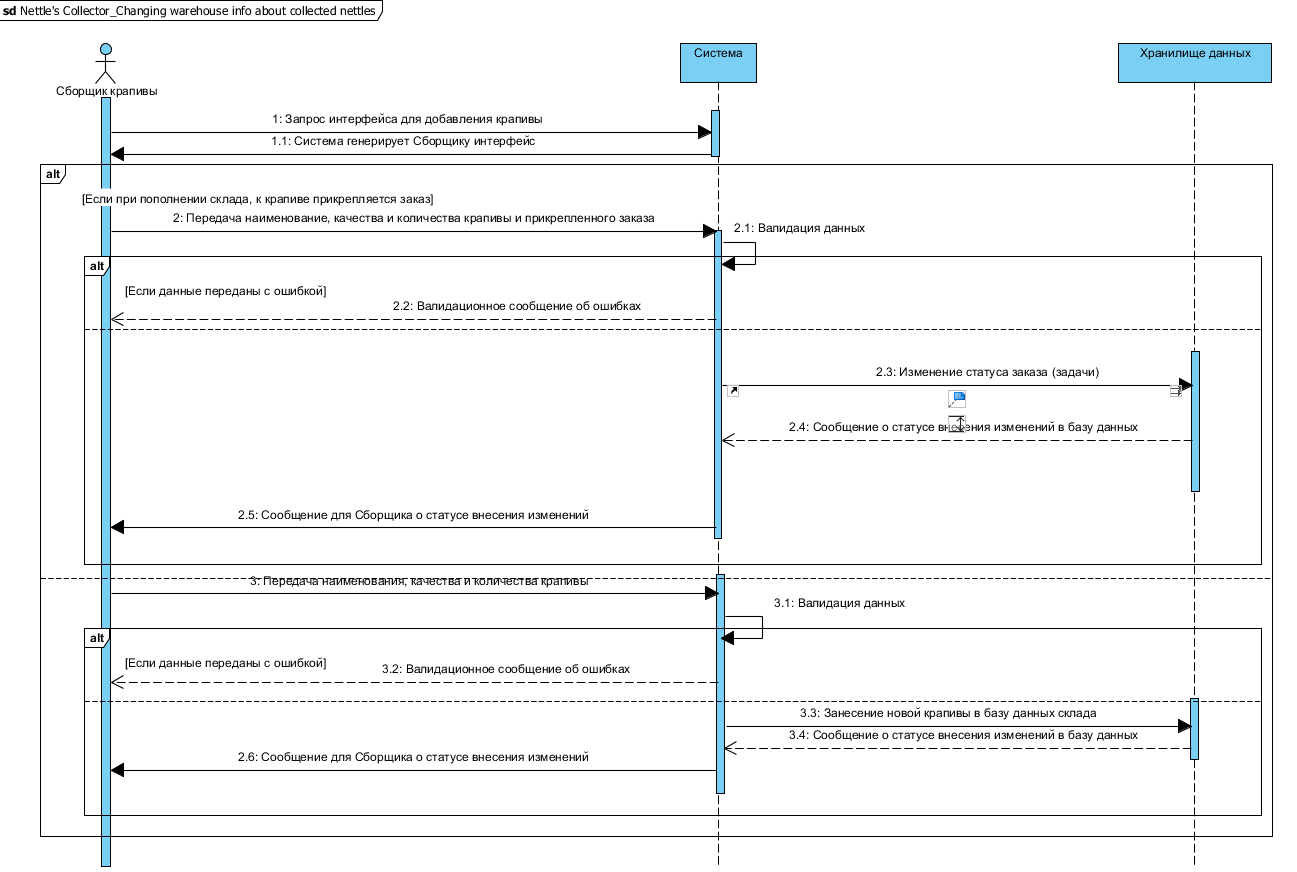


Рисунок 4 - Sequence Diagram for changing warehouse info about collected nettles

## Cooperation Diagram for writing instruction (description) for good by seamstress

Данная диаграмма взаимодействие основных элементов системы (актора, самой системы и хранилища данных) при написании инструкции (описания) изделия швеей (главный актор).

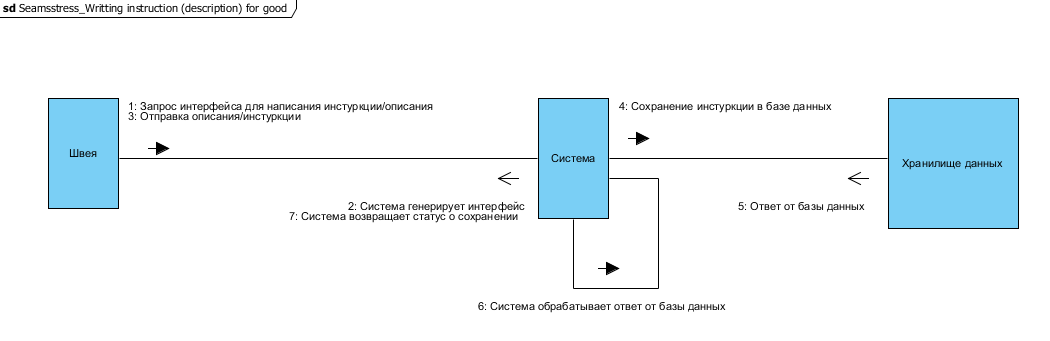


Рисунок 5 - Cooperation Diagram for writing instruction (description) for good by seamstress

## State Machine diagram for key entity “Order”

Ключевой сущностью системы «Иллюзия» является «Заказ» («Order»), так как вся система направлена на работу с ними, обеспечение их выполнения для конечных пользователей путем поочередной работы с этой сущностью всех участников процесса. В ходе этой работы состояния сущности «Заказ» изменяется, как показано на диаграмме ниже.

Остальные сущности в системе являются подчиненными этой сущности и, соответственно, каждое из состояний сущности «Заказ» определяется агрегацией нескольких других сущностей.

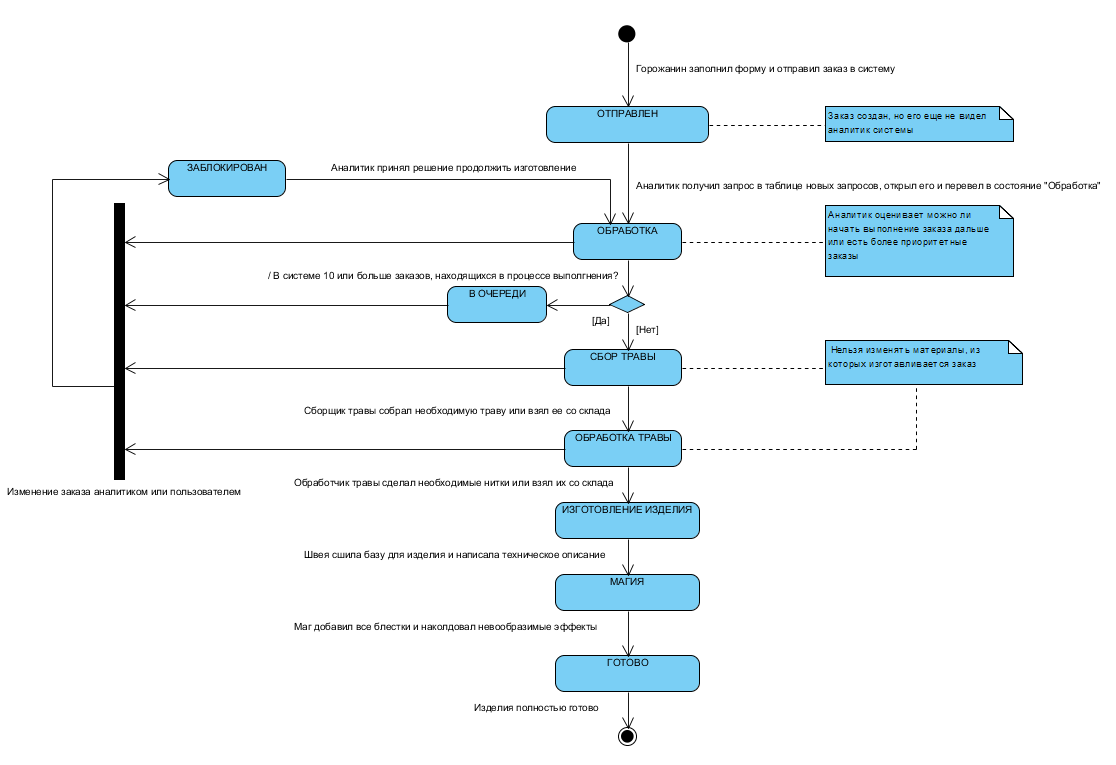


Рисунок6 - State Machine Diagram for states of the entity "Order"

Кроме состояний у заказов также есть статус, который может быть изменен пользователем или аналитиком. По умолчанию статус – «Обычный». Пользователь при создании заказа или во время стадии «Обработка» может изменить его на «Срочный». На дальнейших стадиях изменения запрещены для пользователя, а аналитик может только заблокировать «Заказ». Ниже представлена диаграмма изменений состояний статуса заказа.

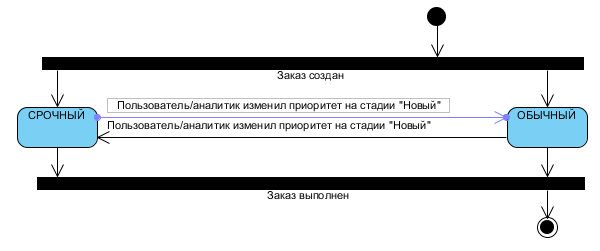


Рисунок7 - State Machine Diagram for statuses of the entity "Order"

Соответственно, статусы и состояния связаны следующим образом:

Таблица 2 - Соотношение крупноблочных состояний и статусов заказа

|  |  |  |
| --- | --- | --- |
| Состояния \ Статусы заказа | Обычный | Срочный |
| Новый | + | + |
| В процессе | + | + |
| Заблокированный | + | + |
| Сделанный | + | + |

В этой таблице состояния описаны крупными блоками:

* «Новый» - включает состояния «Отправлен», «Обработка»;
* «В процессе» - включает состояния «В очереди», «Сбор травы», «Обработка травы», «Изготовление изделия», «Магия»;
* «Заблокированный» - включает состояние «Заблокирован»;
* «Сделанный» - включает состояние «Готово».

## Preliminary Class Diagram

Данный тип диаграмм отображает предварительную структуру классов приложения и их взаимодействие друг с другом.

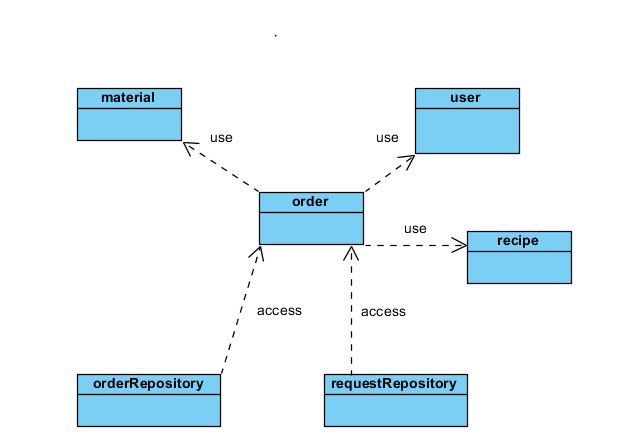


Рисунок 8 - Preliminary Class Diagram (key-value entities' relations)

## Data Base Diagram

Данный тип диаграммы отображает предварительную структуру базы данных. Дальнейшее ее уточнение будет производиться во время реализации системы.

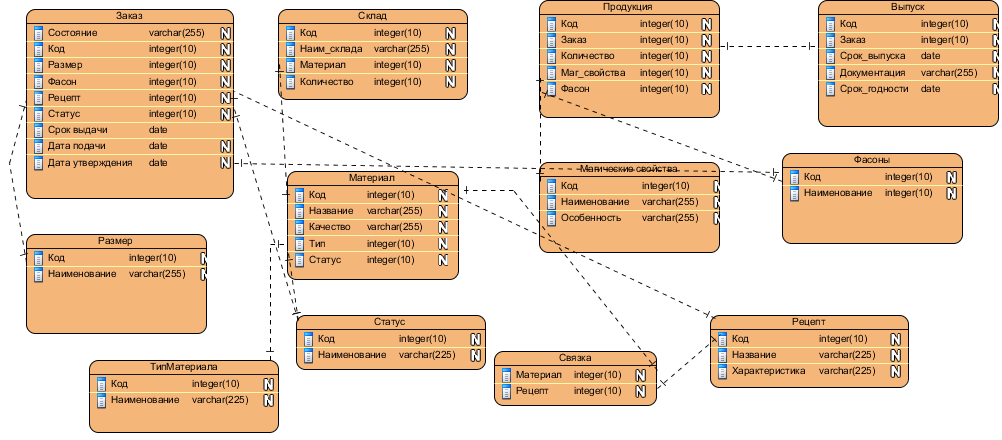


Рисунок9 - Data Base Diagram

## Use-Case Realizations

Описание реализации прецедентов описывается в [USECASES].

# Logical View

Этот раздел содержит архитектурно важные части конструкции модели такие, как ее декомпозиция в подсистемы и пакеты. И для каждого значимого пакета представлена его декомпозиция в классы и утилиты класса. Также раздел содержит подробное описание всех значимых классов системы с точки зрения их функциональных обязанностей. Показаны ключевые отношения между классами, операции класса и атрибуты.

## Overview



Рисунок 10 - Layers (base)

Модель иерархического представления приложения “Иллюзия” основывается на стратегии иерархического представления ответственности, которая связывает каждый уровень с определенной ответственностью.

Эта стратегия была выбрана, потому что она изолирует различные обязанности системы друг от друга так, чтобы были улучшены и дальнейшая системная разработка, и обслуживание.

## Architecturally Significant Design Packages

Данный тип диаграммы отображает взаимодействие логических уровней системы. В левой части диаграммы отображены названия уровней и то, как они взаимодействуют между собой. В правой части – уточнение и указание уровней и варианты реализации системы.

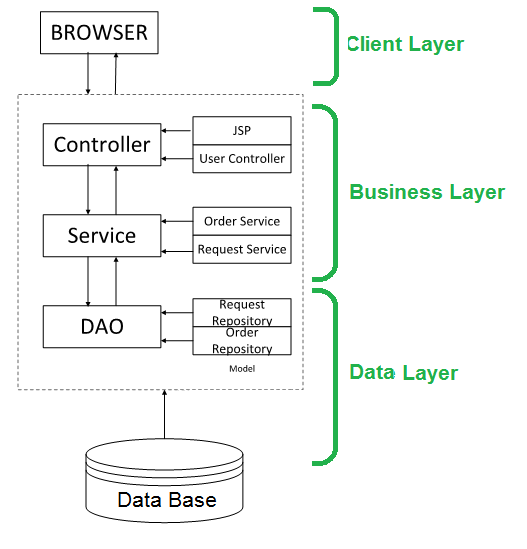


Рисунок 11 – Layers (extended)

## Detailed Class Diagram for key-value entities

Эта диаграмма является дополненной версией диаграммы классов из Use Case View. Она также показывает все ключевые сущности системы и взаимосвязи между ними, но в отличие от предыдущей диаграммы, эта диаграмма классов описывает все поля и методы классов (с их атрибутами доступа). Таким образом, из данной диаграммы можно судить о структуре каждого из ключевых классов и его функциях в системе.

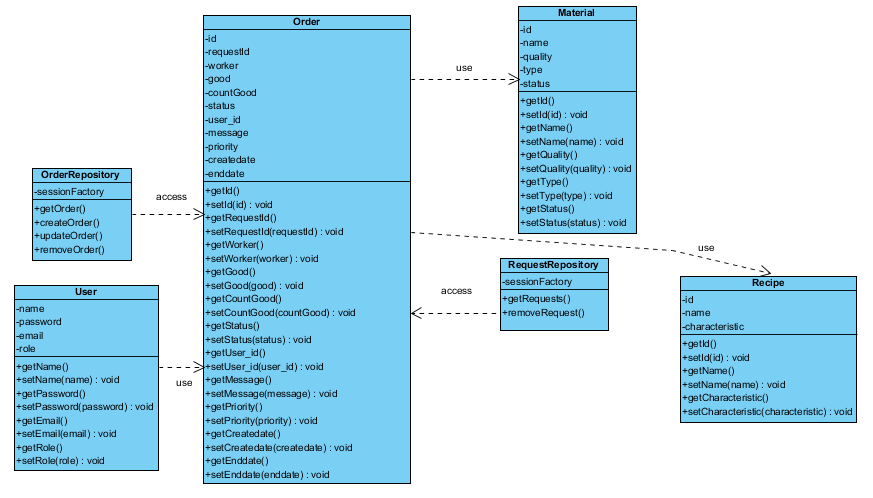


Рисунок 12 - Detailed Class Diagram for key-value entities

## Package Diagram

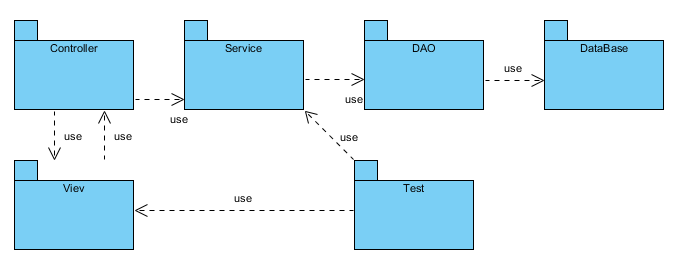


Рисунок 13 - Package Diagram for Logical View

Данная диаграмма описывает разделение всех классов приложения на пакеты.

## Activity Diagram for suspension order running by analyst

Данная диаграмма деятельности описывает вариант использования «Приостановка выполнения заказа», доступный для заказа, который находится в состоянии «в процессе».

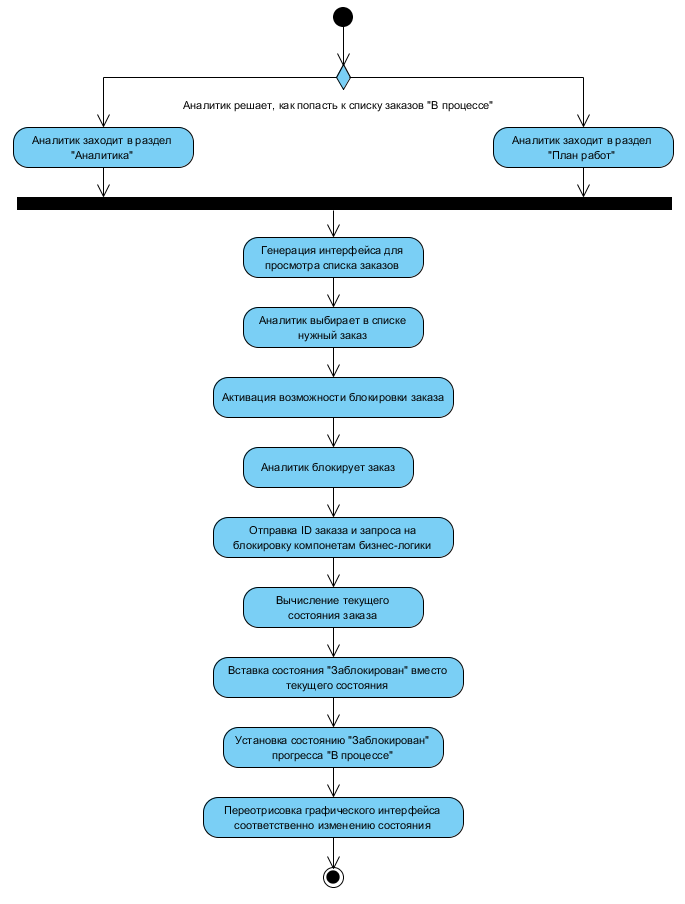


Рисунок 14 - Activity Diagram for suspension order running by analyst

## State Machine Diagram for changing progress of the each order state of “In Progress” type

Диаграмма состояний описывает запрос на изменение прогресса работы над заказом для любого заказа, находящегося в группе состояний «В процессе», описанных в диаграмме состояний из Use Case View.

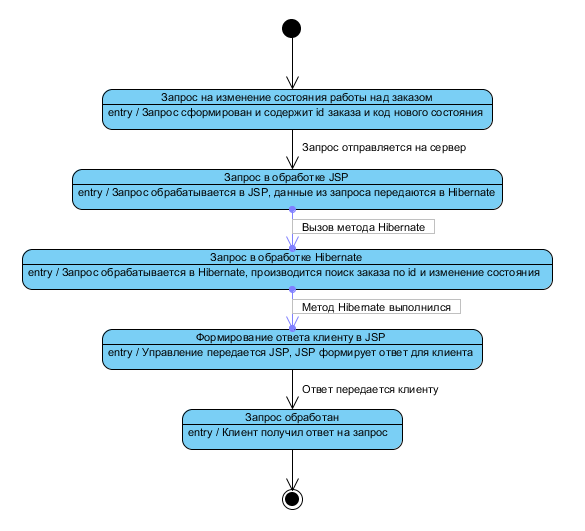


Рисунок 15 - State Machine Diagram for changing progress

## Sequence Diagram for searching order in the analytic view

Данная диаграмма последовательности отображает последовательность действий при выполнении варианта использования «Поиск в аналитическом представлении», который позволяет Аналитику искать заказы в аналитическом представлении либо по их номеру, либо по информации о них.

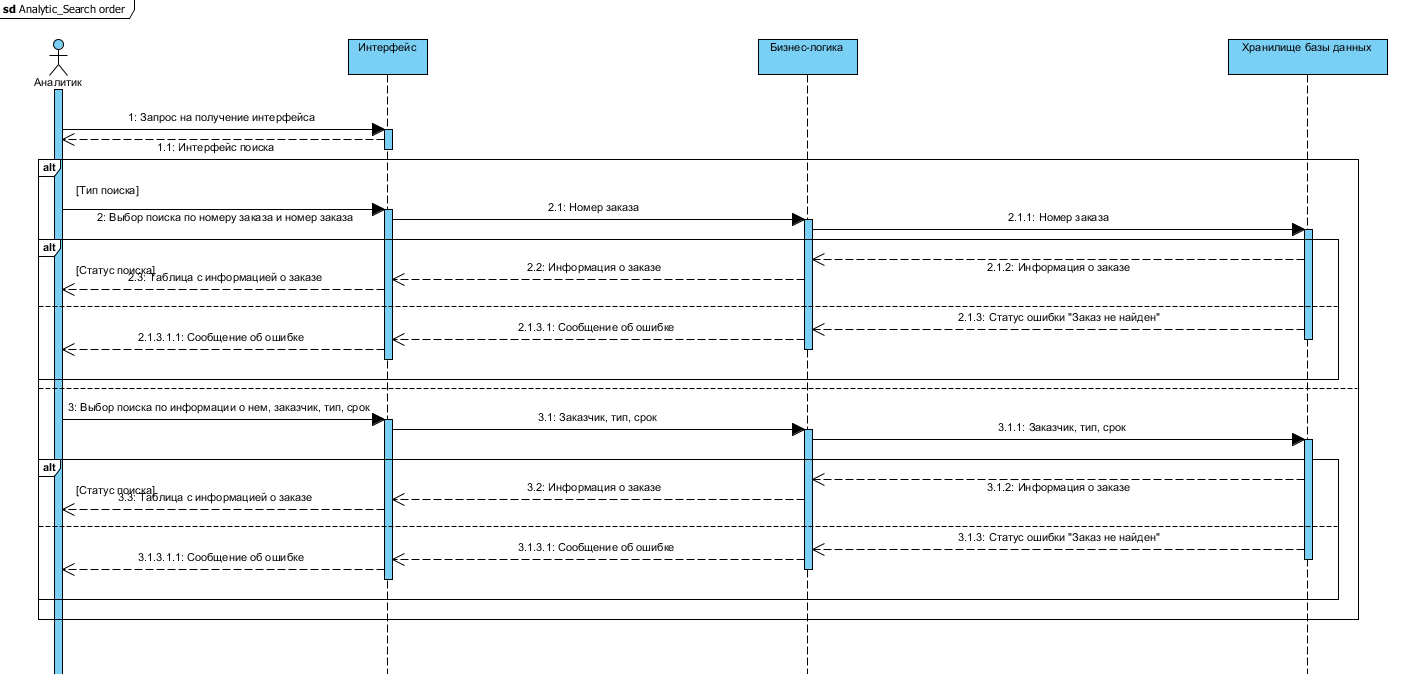


Рисунок 16 - Sequence Diagram for searching order in the analytic view

## Cooperation Diagram

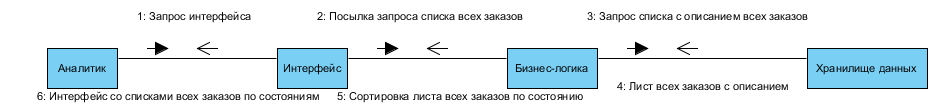


Рисунок 17 - Cooperation Diagram for viewing analytical view

Данная диаграмма взаимодействия отражает прецедент «Просмотр аналитического представления» аналитиком.

# Deployment View

Данное представление описывает конфигурацию, на которой будет развернута и запущена. Конфигурация описывается набором конфигурационных файлов, их описанием, расположением в системе и взаимодействием друг с другом. Это представление является Моделью Развертывания.

## Deployment Diagram

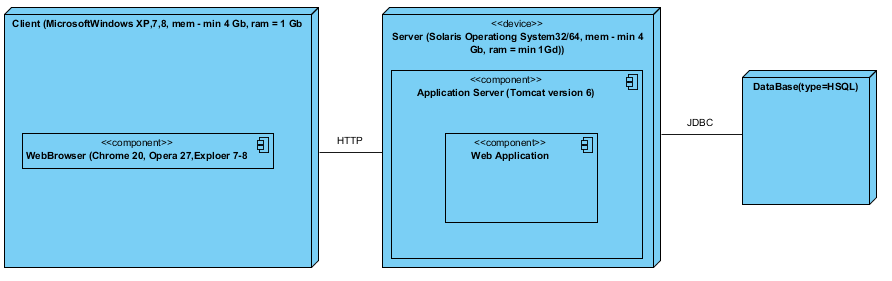


Рисунок 18 - Full Deployment Diagram

# Implementation View

Этот раздел описывает общую структуру системы уже в реализованном виде, разделение этой системы на слои и подсистемы.

## Overview

В данной системе выделяется три слоя, а именно Data layer, Business layer и Client layer.

На уровне Data layer взаимодействие базы данных проходит через hibernate, что позволяет использовать средства для автоматического построения запросов и извлечения данных. Выделяется несколько сущностей для абстрагирования от таблиц в базе данных: fashion, material, order, recipe, size, status, typeofmaterials, user. Классы OrderRepository и RequestRepository позволяют производить различные операции при выборке данных из базы, такие как создание заказа, удаление заказа, получение необходимых данных с заданными параметрами.

На уровне Business layer происходит обработка заказа, заполнение его данными, обновление уже существующего заказа, а так же его удаление. Для этого используется два класса OrderService и RequestService. Класс UserController обеспечивает связь между пользователем и системой, контролирует ввод данных пользователем и использует модели, описанные выше для реализации определенных действий.

Непосредственное взаимодействие с системой происходит через браузер, где пользователь вводит необходимые данные для создания заказа, они проходят клиентскую валидацию, и далее идут по нижележащим уровням. Это уровень Client layer.

## Layers

1. Data layer

В данный слой входит сама база данных, сущности описанные в виде VO, и взаимодействие по получению и отправке данных по заказу и/ или изменению через классы Request repository, Order repository.

1. Business layer

Данный слой описывает взаимодействие с получением, отправкой заказа, его изменением и/или обновлением. Service осуществляется через классы OrderService и RequetsService.Так же на этом уровне осуществляется обработка данных с формы с помощью класса UserController.

1. Client layer

В данном слое пользователь непосредственно может вносить данные для оформления заказа и выбирать из списка существующие характеристики изделия. После того, как заказ будет составлен, система добавляет заказ в базу, для дальнейшего выполнения.

## Activity Diagram for getting recipe of the good

Данная диаграмма деятельности описывает выборку рецептов из базы данных для заказа.

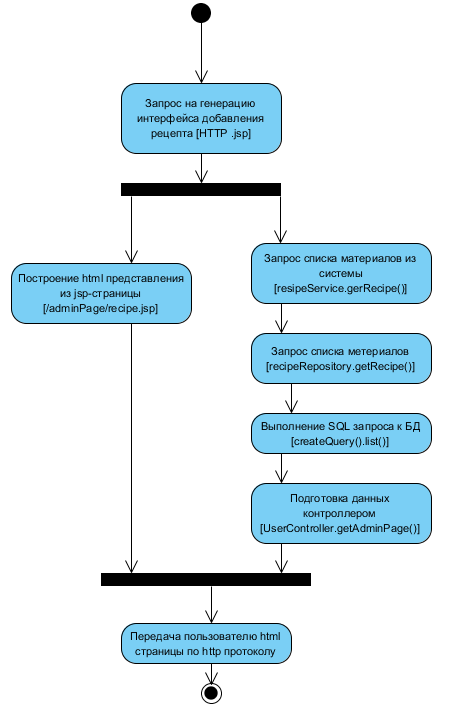


Рисунок 19 - Activity Diagram for getting recipe of the good

## State Machine Diagram for changing quality description by wizard

Данная диаграмма отображает изменение состояний системы при отправке отчета о качестве продукта магом.

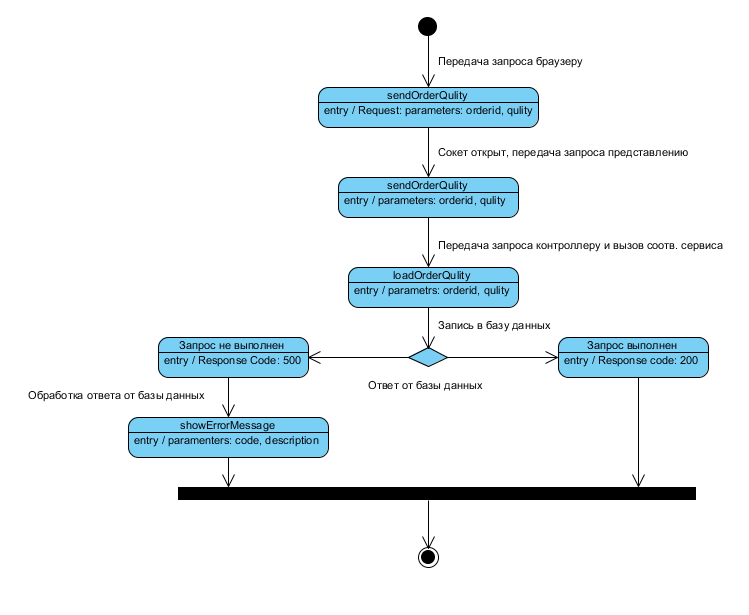


Рисунок 20 - State Machine Diagram for changing quality description by wizard

## Sequence Diagram for uploading predefined values for goods in database

Данная диаграмма последовательности отображает последовательность действий при загрузке типов изделий в базу данных при первичной загрузке системы.

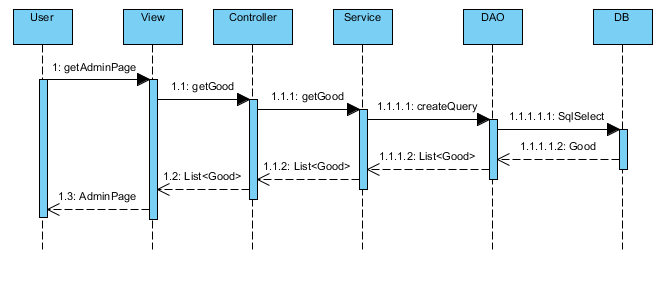


Рисунок 21 - Sequence Diagram for uploading predefinde values for goods in database

## Cooperation Diagram for setting size for the good in the system

Данная диаграмма взаимодействия описывает взаимодействие различных уровней (компонентов) системы при установке размера изделия.

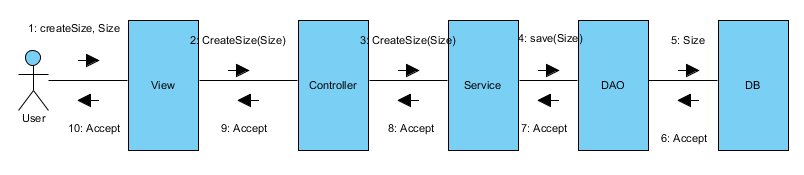


Рисунок 22 - Cooperation Diagram for setting size for the good in the system

## Class Diagram (full version)

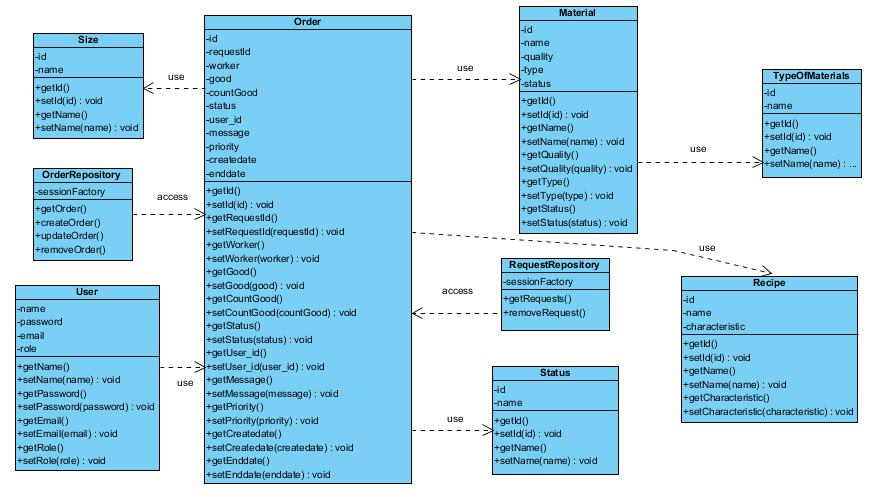


Рисунок 23 - Class Diagram (full)

## Data Base Diagram

Данная диаграмма описывает архитектуру базы данных более детально по сравнению с Use Case View и является реальной структурой базы данных системы.

****

Рисунок7 – Data Base Diagram (detailed)

Таблицы Размер, Тип материалов, Статус, Фасон являются словарями, в которых хранятся уникальные значения.

Таблица Связка представляет реализацию many-to-many связи Рецептов и Материалов.

Поясним некоторые специфичные поля таблиц:

**ORDER –** таблица-заказ. В ней хранится заказ на изготовление текстильного изделия.

ORDER.ORDER\_STATUS – состояние заявки.

ORDER.ORDER\_SIZE – размер текстильного изделия

ORDER.ORDER\_RECIPE – рецепт, по которому будет выполняться изделие.

ORDER.ORDER\_FASON – тип изделия

**RECIPE\_GOOD** – таблица связка many to many таблиц RECIPE и GOOD

**SIZE** – таблица размеров текстильного изделия.

SIZE.SIZE\_NAME – размеры

STATE – таблица состояния заказа

STATE.STATE\_NAME - состояние

**FASHION** – таблица типов текстильных изделий

FASHION .FASHION\_NAME – типы изделий

**GOOD** – таблица материалов, которые используются в рецепте

GOOD.GOOD\_NAME – название материала

GOOD.GOOD\_QUALITY – качество материала

GOOD.GOOD\_GOODTYPE – тип материала.

**GOODTYPE** – таблица типов материала

GOODTYPE.GOODTYPE\_NAME– наименование типа материала

**WAREHOUSE**- таблица, описывающая наличие материалов и их кол-во на складе.

WAREHOUSE.WAREHOUSE\_NAME – имя склада

WAREHOUSE.WAREHOUSE\_GOOD – Материал

WAREHOUSE.WAREHOUSE\_COUNT – кол-во материала

**RECIPE** – таблица рецептов, по которым производится изготовление текстильного изделия.

RECIPE.RECIPE\_NAME– Название рецепта

RECIPE.RECIPE\_SPECIFICATION – Характеристики рецепта, т.е. его описание

**MAGIC-**  в данной таблице хранится вся информация о магических свойствах или особенностях.

MAGIC.MAGIC\_NAME – название свойства

MAGIC\_MAGIC\_SPECIFICATION - описание

**PRODUCT –** таблица хранит созданные товары, являющиеся результатом выполнения заказа.

PRODUCT.PRODUCT\_FINISHDATE– срок выпуска

PRODUCT.PRODUCT\_PERIODUSE– срок годности

PRODUCT.PRODUCT\_SPECIFICATION – документация

PRODUCT.MAGIC\_ID – магические свойства

PRODUCT.PRODUCT\_FASHION – фасон изделия

При заполнении нового Заказа необходимо выбрать (привязать) один из рецептов.

# Data View

Представление данных представляет значительную часть системы «Иллюзия». Модуляризация (нормализация) выбрана как подход к проектированию физической модели данных. Непротиворечивость данных и качество осуществлены через первичные и внешние ключи.

Возможность просмотра данных, их изменения и внесения новых в представление данных реализовано через пользовательский интерфейс системы. Однако, в системе есть словари данных, такие как “Magic”, “State” и другие, которые не доступны для изменения через пользовательский интерфейс. Добавление новых значений или изменение имеющихся возможно напрямую через базу данных, и возможности для проведения этих операций через пользовательский интерфейс не предусматривается в системе, как новый функционал.

Представление данных описывается диаграммой базы данных, которая представлена в описании Implementation View.

# Size and Performance

Производительность:

* С максимальной нагрузкой все транзакции выполняются в соответствии со стандартным сценарием соединения сервера с базой данных с таймаутом – 60 секунд. Механизм таймаута обеспечивается с помощью Spring Security, который предоставляет возможность работы через сущность Session Factory. Session Factory настроена на выполнение данного нефункционального требования.

# Quality

Данный раздел описывает, насколько онлайн-приложение обеспечивает качество по следующим критериям:

Масштабируемость:

* + Описание: реакция системы при увеличении числа пользователей
  + Решение: Данная версия системы не нацелена на поддержание большого количества пользователей. Любые проблемы с производительностью будут решатся в следующей версии системы.

Надежность, доступность:

* + Описание: прозрачный механизм отказоустойчивости, включая время между отказом
  + Решение: Нагрузка не выравнивается. Если в системе возникает перегрузка, запросы перестают выполнятся, так как на данный момент система не рассчитана на активное использование. Вопрос надежности будет пересмотрен в следующей версии продукта.

Мобильность:

* + Описание: возможность развертывания системы в любой среде
  + Решение: система может быть развернута только в той среде, которая описана в данном документе, так как она является частной системой и предполагается развертывание только в одной единственной среде.

Безопасность:

* + Описание: защита данных пользователей и защита целостности системы
  + Решение: механизмы аутентификации и авторизации, Session Factory, Spring Security