**УЧРЕЖДЕНИЕ ОБРАЗОВАНИЯ**

**БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ**

**ИНФОРМАТИКИ И РАДИОЭЛЕКТРОНИКИ**

**Факультет непрерывного и дистанционного обучения**

# Специальность: Автоматизированные системы обработки информации

**КУРСОВОЙ ПРОЕКТ**

**ПО ОБЪЕКТНО-ОРИЕНТИРОВАННОЕ ПРОГРАММИРОВАНИЕ И ПРОЕКТИРОВАНИЕ**

**Вариант № 4**

**Тема: Объектно-ориентированный анализ и проектирование программного обеспечения. Программное обеспечение онлайновой театральной кассы**

***Соболевского Дмитрия Александровича***

***Группа: 590651***

***Зачетная книжка: ‎000623-28***

***Электронный адрес:*** [***sobolevskidmitry@gmail.com***](mailto:sobolevskidmitry@gmail.com) ***/ BSUIR\sda***

Оглавление

[Глава 1. Постановка задачи 3](#_Toc280709553)

[Глава 2. Анализ требований 4](#_Toc280709554)

[Глоссарий 4](#_Toc280709555)

[Описание действующих лиц 4](#_Toc280709556)

[Диаграмма вариантов использования 4](#_Toc280709557)

[Описание вариантов использования 5](#_Toc280709558)

# Введение

В предложенном варианте требуется при помощи CASE-средства Rational Rose построить модель программного обеспечения. Процесс создания модели состоит из нескольких этапов:

1. Составление глоссария проекта.
2. Создание модели вариантов использования.
3. Анализ вариантов использования (по окончании производится промежуточная сдача задания).
4. Проектирование системы.
5. Реализация системы.

Процесс создания модели должен проходить так, как это описано в методическом пособии [[Вендров 2004].](file:///D:\Learning\%D0%9E%D0%B1%D1%8A%D0%B5%D0%BA%D1%82%D0%BD%D0%BE-%D0%BE%D1%80%D0%B8%D0%B5%D0%BD%D1%82%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%BD%D0%BE%D0%B5%20%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5%20%D0%B8%20%D0%BF%D1%80%D0%BE%D0%B5%D0%BA%D1%82%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5.%20%D0%A7%D0%B0%D1%81%D1%82%D1%8C%202\files\umlpracticum2004.pdf) Структура модели в браузере Rose должна соответствовать структуре, предусмотренной Rational Unified Process.

После выполнения третьего этапа модель должна удовлетворять перечисленным ниже требованиям. Глоссарий проекта должен иметь вид таблицы и храниться в отдельном файле. На диаграммах вариантов использования каждое действующее лицо (actor) и вариант использования должны сопровождаться описанием. Эти описания должны быть составлены на русском языке. Описание действующего лица должно коротко (в одну-две строки) сообщать о роли данного лица. Описание варианта использования должно включать в себя пояснение, предусловие, потоки событий (основной и альтернативные, если таковые есть) и постусловие. Описания представляют собой либо присоединенные текстовые файлы, либо текст, введенный в поле Documentation спецификации соответствующего элемента диаграммы. Диаграммы взаимодействия, соответствующие потокам событий вариантов использования, должны содержать необходимые пояснения. Сложные потоки событий (с ветвлениями, циклами) должны быть смоделированы с помощью диаграмм деятельности.

\* \* \*

При проектировании системы требуется:

* создать иерархию классов системы;
* для классов указать стереотипы;
* разместить классы по пакетам в Design model, как это описано в методичке (придерживаться принципа: количество связей внутри пакета больше количества связей, проходящих за границы пакета);
* связать объекты на диаграммах взаимодействия с классами, а сообщения – с операциями;
* каждый класс снабдить описанием, которое должно включать в себя краткое описание (ответственность класса), описание атрибутов в виде таблицы (имя, описание, тип), таблицу с описанием операций (имя, описание, сигнатура);
* построить диаграммы классов системы, отображающие связи между классами;
* для описания поведения экземпляров отдельных классов построить диаграммы состояний;
* построить диаграммы деятельности для моделирования сложных операций (с альтернативами, циклами);
* разработать (если это требуется вариантом задания) схему базы данных и отобразить ее на диаграмме «сущность – связь».

При реализации системы необходимо построить диаграммы компонентов для каждого пакета и для системы в целом. Также следует разработать диаграмму размещения. В зависимости от варианта задания диаграмма размещения должна показывать расположение компонентов в распределенном приложении или связи между встроенным процессором и устройствами. Должна быть произведена проверка корректности модели средствами Rational Rose.

# Глава 1. Анализ требований

**Вариант 4. Онлайновая театральная касса**

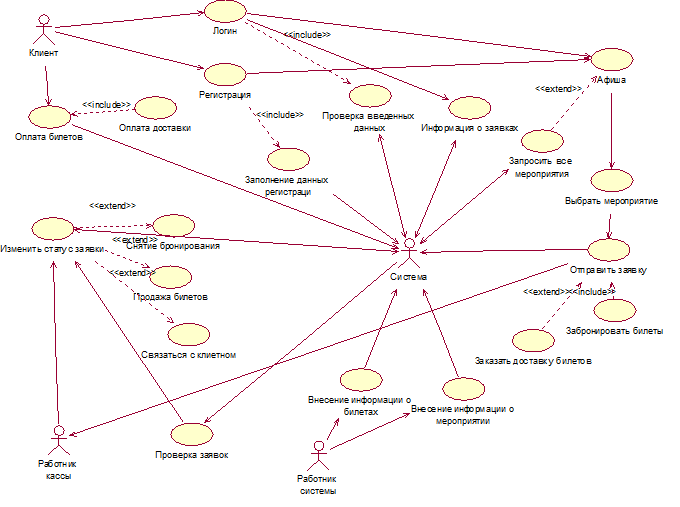
Онлайновая театральная касса "Билетов.Нет" представляет собой web-сайт службы бронирования и доставки билетов на спектакли и концерты.  
Перед тем как впервые воспользоваться услугами кассы клиент должен зарегистрироваться. В ходе регистрации он указывает данные о себе (ф. и. о., телефон, адрес электронной почты) и получает логин и пароль (логины и пароли разных клиентов не должны совпадать).  
Войдя в систему, клиент может ознакомиться с афишей, выбрать интересующее его мероприятие, указав название, дату и место проведения. Получив от системы сведения о билетах имеющихся в наличии, пользователь может забронировать нужное ему количество билетов. Билеты бывают разных типов: партер, балкон, ложа, бельэтаж, 1-2-3 ярус, vip-места и т. п. Цена билета зависит от его типа и расположения зрительского места. Билеты могут быть выкуплены в течение трех суток с момента бронирования, но не позднее пяти суток до начала спектакля. Клиент может самостоятельно выкупить забронированные билеты, приехав в офис, или заказать доставку билетов курьером, сделав пометку в заявке и указав адрес доставки. Стоимость доставки зависит от дальности: центр / спальный район / дальний пригород. Клиент может получить информацию обо всех своих заявках с web-страницы онлайновой кассы.  
Заявки клиентов хранятся в системе. В каждой указаны: сведения о клиенте, название спектакля, место и время проведения, количество и тип забронированных билетов, стоимость билетов, время создания заявки, время оплаты, вид доставки (самовывоз / курьер), адрес доставки, стоимость доставки, статус заявки (новая / рабочая / оплаченная / аннулированная). По истечении 12 месяцев с момента создания заявки данные автоматически удаляются из системы.  
В обязанности работников онлайновой кассы входит внесение в систему сведений о мероприятиях и об имеющихся в продаже билетах. Данные о мероприятии – вид: концерт / шоу / спектакль; название; описание; место проведения; дата; – хранятся в системе. Один и тот же спектакль может идти в разные дни и в разных местах, но разные спектакли не могут пересекаться по времени и месту проведения. Запись о билете содержит название спектакля, дата, время, место проведения, тип билета, зрительский ряд, зрительское место, цену билета, статус билета (есть в наличии / забронирован / продан / передан для реализации). По истечении 12 месяцев с даты, указанной в билете, данные автоматически удаляются из системы.  
Работник кассы, получив новую заявку клиента, связывается с ним для подтверждения и уточнения мест. Согласовав с клиентом зрительские места, работник делает пометку о бронировании билетов в системе (тем самым уменьшается количество билетов, имеющихся в наличии) и меняет статус заявки на "рабочая". После оплаты и/или доставки "рабочей" заявки билеты из заявки помечаются как проданные, а заявка – как оплаченная. За 5 суток до начала спектакля все не проданные билеты передаются для реализации в обычные кассы, в системе они автоматически помечаются как "передан для реализации", заявки на них аннулируются, клиенты, не успевшие оплатить заказанные билеты, информируются о снятии брони. Через 4 суток после создания "рабочие" заявки автоматически аннулируются, бронирование с билетов снимается, клиентам посылается соответствующее сообщение. Также должна быть возможность аннулирования заявок вручную работниками онлайновой кассы. При аннулировании заявки вручную работник должен уведомить клиента, изменить статус заявки, снять бронирование билетов (количество билетов в наличии возрастает).

# Глава 2. Глоссарий проекта

|  |  |
| --- | --- |
| Клиент | Человек решивший воспользоваться системой Онлайновая театральная касса "Билетов.Нет" |
| Работник кассы | Наемный служащий работающий в театральной кассе «Билетов.Нет» |
| Работник системы | Наемный служащий работающий в Системе Онлайновой театральной кассы «Билетов.Нет». |
| Система | Web-сайт службы бронирования и доставки билетов на спектакли и концерты |
| Заявка | Заявка содержит информацию: о клиенте, количестве билетов, и доставке |
| Афиша | Список всех мероприятий системы с датами и информацией о них |

# Глава 3. Проектирование системы

## Диаграмма вариантов использования



## **Описание вариантов использования**

### Вариант использования «Логин (Login)»

*Краткое описание:*

Данный вариант использования описывает вход пользователя в систему.

*Основной поток событий:*

Данный вариант использования начинает выполняться, когда пользователь хочет войти в систему.

1. Система запрашивает логин и пароль.

2. Пользователь вводит логин и пароль.

3. Система подтверждает логин и пароль, после чего открывается доступ в систему.

*Альтернативные потоки:*

Неправильное имя/пароль:

Если во время выполнения основного потока обнаружится, что пользователь ввел неправильное логин и/или пароль, система выводит сообщение об ошибке. Пользователь может вернуться к началу основного потока или отказаться от входа в систему, при этом выполнение варианта использования завершается.

*Предусловия:*

Отсутствуют.

*Постусловия*:

Если вариант использования выполнен успешно, пользователь входит в систему. В противном случае состояние системы не изменяется.

### Вариант использования «Регистрация (Register)»

*Краткое описание:*

Данный вариант использования описывает регистрацию пользователя в системе.

*Основной поток событий:*

Данный вариант использования начинает выполняться, когда пользователь хочет зарегистрироваться в

системе .

1. Система запрашивает ф.и.о, телефон, адрес электронной почты.

2. Пользователь вводит ф.и.о, телефон, адрес электронной почты.

3. Система генерирует уникальный логин и пароль для пользователя.

4. Пользователь получает логин и пароль

*Альтернативные потоки:*

Неправильное имя/пароль:

Если во время выполнения основного потока обнаружится, что пользователь ввелуже зарегестрированный адрес электронной почты или телефон система выводит сообщение об ошибке. Пользователь можетвернуться к началу основного потока или отказаться от регистрации, при этом выполнениеварианта использования завершается.

*Предусловия:*

Отсутствуют.

*Постусловия:*

Если вариант использования выполнен успешно, пользователь входит в систему и регистрация завершается. Впротивном случае состояние системы не изменяется.

### Вариант использования «Афиша (Аfisha)»

*Краткое описание:*

Данный вариант использования описывает ознакомление клиента с афишей.

Клиент может ознакомиться с афишей, выбрать интересующее его мероприятие, указав название, дату и место проведения.

*Основной поток событий:*

Данный вариант использования начинает выполняться, когда пользователь хочет ознакомиться с афишей

1. Пользователь указывает название мероприятие, дату и место проведения

2. Система предоставляет мероприятия подходящие под критерии пользователя

*Альтернативные потоки:*

Неправильная информация:

Если во время выполнения основного потока обнаружится, что пользователь ввел не правильную дату или место проведения система выводит сообщение об ошибке. Пользователь может вернуться к началу основного потока или ввести данные заново.

*Предусловия:*

Выполненная авторизация или регистрация для нового пользователя

*Постусловия:*

Если вариант использования выполнен успешно, пользователей получает список мероприятий. В противном случае состояние системы не изменяется.

### Вариант использования «Выбрать мероприятие (AfishaSelectEvent)»

*Краткое описание:*

Данный вариант использования описывает получение информации о выбранном мероприятии.

*Основной поток событий:*

Данный вариант использования начинает выполняться, когда пользователь хочет получить информацию о выбранном мероприятии.

1. У системы запрашиваются информация о запрошенном мероприятии (название, дату и место проведения).

2. Пользователь получает данные о интересующем его мероприятии, и данные по наличию билетов, их типе и цене.

*Альтернативные потоки:*

Отсутствие билетов :

При отсутствии билетов на мероприятии, пользователь получает уведомление. И возможность выбрать другое мероприятие.

*Предусловия:*

Пользователь авторизован, и получил список мероприятий

*Постусловия:*

Если вариант использования выполнен успешно, пользователь получает информации по мероприятию.

### Вариант использования «Отправить заявку (TicketsSelect)»

*Краткое описание:*

Данный вариант использования описывает отправление заявки по выбранным билетам.

*Основной поток событий:*

Данный вариант использования начинает выполняться, когда пользователь бронирует билеты на данном мероприятие.

1. Создается заявка с статусом "Новая", хранящая в себе: сведения о клиенте, название спектакля, место и время проведения, количество и тип забронированных билетов, стоимость билетов, время создания заявки, время оплаты, вид доставки, адрес доставки, стоимость доставки, статус заявки.

2. Заявка передается в систему.

*Предусловия:*

Пользователь авторизован, получен список мероприятий, выбрано мероприятие

*Постусловия:*

Если вариант использования выполнен успешно, пользователь получает оповещении о поданной заявке.

### Вариант использования «Внесение информации о мероприятии(AddingEventInformation)»

*Краткое описание:*

Данный вариант использования описывает добавление информации о мероприятие работником системы.

*Основной поток событий:*

Данный вариант использования начинает выполняться, когда работник системы добавляет информации по мероприятию.

1. Создается новое мероприятие указывается:

- вид: концерт / шоу / спектакль;

- название;

- описание;

- место проведения;

- дата;

2. Данные отправляются в систему и там сохраняются

*Предусловия:*

Работник системы авторизован

*Постусловия:*

Если вариант использования выполнен успешно, в систему добавляется новое мероприятие.

### Вариант использования «Внесение информации о билетах (AddingTickets)»

*Краткое описание:*

Данный вариант использования описывает добавление информации о билетах работником системы.

*Основной поток событий:*

Данный вариант использования начинает выполняться, когда работник системы добавляет билеты в систему.

1. Создается новое мероприятие указывается :

- название спектакля;

- дата;

- время;

- место проведения;

- тип билета;

- зрительский ряд;

- зрительское место;

- цена билета;

- статус билета;

2. Данные отправляются в систему и там сохраняются

*Альтернативные потоки:*

Некорректная информация по мероприятию:

при указании некорректной информации по мероприятие при добавлении информации по билетам система выводит сообщение об ошибке, пользователь может ввести данные заново.

*Предусловия:*

Работник системы авторизован

*Постусловия:*

Если вариант использования выполнен успешно, в систему добавляется информация по билетам.

### Вариант использования «Проверка заявок (SystemCheckRequest)»

*Краткое описание:*

Данный вариант использования описывает работу системы при автоматической проверке заявок и билетов.

*Основной поток событий:*

Данный вариант использования начинает выполняется каждый день

1. У системы запрашиваются информация о всех заявках.

2. И проверяется их возраст, если он более 12 месяцев, то они удаляются из системы.

3. Если возраст рабочей заявки больше либо равен 4 суткам, заявки аннулируются с билетов снимается бронь.

4. Если в заявке (статус: "Рабочая") есть билеты на мероприятия которые начнутся через 5 суток, заявка передается на аннулирование, бронирование билетов снимается, клиентам посылается сообщение

5. У системы запрашиваются все не проданные билеты на мероприятия которые начнутся через 5 суток.

6. Билеты передаются для реализации в обычные кассы

*Предусловия:*

Наличие заявок и билетов в системе

*Постусловия:*

Если вариант использования завершится успешно, заявки будут переданы на аннулирование, или удалены. В противном случае состояние системы не изменится.

### Вариант использования «Изменить статус заявки (SystemCheckRequest)»

*Краткое описание:*

Данный вариант использования описывает изменение заявки о заказе билетов.

*Основной поток событий:*

Данный вариант использования начинает выполняться, когда работник кассы получает уведомление от системы о новой заявке от пользователя или изменение статуса заявки от системы.

Приходит заявка с статусом "Новая":

1. Работнику кассы приходит новая в заявка в статусе "Новая".

2. Работник кассы связывается с клиентом для подтверждения и уточнения мест

3. Согласовав с клиентом заявку, работник бронирует билеты и присваивает заявке статус "Рабочая"

Приходит заявка с статусом "Рабочая":

1. Работнику кассы приходит новая в заявка в статусе "Рабочая".

2. Если заявка подходит под условия аннулирования, система присваивает заявке статус "Аннулированная", оповещает клиента

Приходит информация от системы об оплате заявки:

1. В систему приходит информация об оплате заявки.

2. Система меняет статус заявки "Оплаченная"

Ручное аннулирование заявки:

1. Работник кассы запрашивает у системы нужную заявку.

2. Работник меняет статус заявки на "Аннулированная" и оповещает клиента.

3. Снимает бронированные билеты

4. В систему передается сообщение о освободившихся билетах

*Предусловия:*

Работник кассы авторизован или пришла заявка с статусом "Рабочая"

*Постусловия:*

Если вариант использования выполнен успешно, статус заявки изменяется. В противном случае состояние системы не изменяется.

### Вариант использования «Информация о завках (UserRequests)»

*Краткое описание:*

Данный вариант использования описывает получение информации о текущих заявках пользователя.

*Основной поток событий:*

Данный вариант использования начинает выполняться, когда пользователь нажимает на кнопку получить информацию о текущих заявках.

1. У системы запрашиваются все заявки пользователя.

2. Пользователь получает все свои заявки.

*Альтернативные потоки:*

Отсутствие заявок:

Если во время выполнения основного потока обнаружится, что пользователь не имеет заявок то система выводит сообщение.

*Предусловия:*

Пользователь авторизован

*Постусловия:*

Если вариант использования выполнен успешно, пользователь получает список своих заявок от системы. В противном случае состояние системы не изменяется.