МИНОБРНАУКИ РОССИИ

Федеральное государственное бюджетное образовательное учреждение

высшего образования

«МИРЭА — Российский технологический университет»

РТУ МИРЭА

Институт информационных технологий (ИТ)

Кафедра инструментального и прикладного программного обеспечения (ИиППО)

ОТЧЁТ ПО ПРАКТИЧЕСКИМ ЗАНЯТИЯМ

по дисциплине

«АРХИТЕКТУРА ПРОГРАММНЫХ ПРОДУКТОВ И СИСТЕМ»

Выполнили студенты группы ИКБО-03-18 Маковецкий И. А.

Ненашев И. А.

Принял ассистент Ермаков С. Р.

Практические занятия выполнены «____» _____ 2021 г.

Практические занятия зачтены «____» _____ 2021 г.

Москва 2021

СОДЕРЖАНИЕ

1 ПРАКТИЧЕСКАЯ РАБОТА № 1 «ВЫЯВЛЕНИЕ ЗАИНТЕРЕСОВАННЫХ ЛИЦ В ИТ-ПРОЕКТАХ»

1.1 Цель работы

Сформировать навыки работы с реальными заказчиками программных систем; идентификации заинтересованных лиц и интервью с ними; анализа полученного материала; формулирования проблемы, ее актуальности и потребностей заинтересованных лиц.

1.2 Краткие теоретические сведения

На этапе анализа проблемы проводится анализ предметной области, для которой разрабатывается ПО.

Цели этапа:

1. Определение границ или контура системы.
2. Описание объектов автоматизации и / или формализации данных об этих объектах.
3. Выявление или определение потребностей заказчика ПО.

Анализ предметной области можно проводить, например, основываясь на теории системного анализа и использовать предложенные в ней методы.

Исходными данными для этапа системного анализа являются:

1. Регламенты работы отделов и должностные инструкции сотрудников этих отделов.
2. Анкеты опроса заинтересованных лиц.

3. Записи интервью с заинтересованными лицами.
4. Другие документы, имеющие отношение к исследуемому объекту.

Выходными данными, или результатом, этапа системного анализа являются:

1. Перечень заинтересованных лиц.
2. Список потребностей заинтересованных лиц в разрабатываемом ПО.
3. Описание объектов автоматизации.
4. Модель объектов автоматизации или предметной области.

1.3 Задание

1. Составить перечень заинтересованных лиц.
2. Провести интервью и/или анкетирование с каждым заинтересованным лицом.
3. Проанализировать полученную информацию и сформулировать актуальность проблемы и потребности заинтересованных лиц.

1.4 Выполнение практической работы

В качестве объекта автоматизации выбран ресторан армянской кухни имени Мартина Хайдеггера «Этр лю’ля».

1.4.1 Перечень заинтересованных лиц

1. Главный кондитер.
2. Менеджер по персоналу.

1.4.2 Интервью с заинтересованными лицами

1.4.2.1 Интервью с главным кондитером

1. Имя. Хуциевян, Марлен Мартынович.

2. Наименование организации. ООО «Ресторан армянской кухни имени Мартина Хайдеггера l'Êtrelà.»

3. Наименование структурного подразделения. Кухня, кондитерский цех.

4. Должность. Главный кондитер.

5. Кому Вы непосредственно подчиняетесь? Исполняющему директору ресторана.

6. Каковы Ваши основные обязанности? Контроль десертов на соответствие внутренним регламентам ресторана, административное управление подчиненными, производство десертов.

7. Что Вы в основном производите? Гата, нузук, суджук, пахлаву по-ереванski, торт «Микадо».

8. Для кого? Посетителей, проверяющих и выручки директора ресторана.

9. Какие документы или какую информацию можно считать входящими, или необходимыми, для Вашей деятельности? Рецепты и входящее сообщение от официанта с заказом посетителя, а также распоряжения руководства.

10. Какие документы или какую информацию можно считать исходящими, или результатом Вашей деятельности? Сообщение официанту о статусе изготовления продукта, отчет о необходимых ингредиентах управляющему.

11. Как измеряется успех Вашей деятельности? Качественным образом — в отзывах клиентов. Количественных метрик для описания вкуса торта «Микадо» найти не удалось.

12. Какие проблемы влияют на успешность Вашей деятельности? Несве-

жие ингредиенты, плохое положение республики Армения на политической арене, неопытные кондитеры в моем подчинении, etc, etc.

13. Какие тенденции, если такие существуют, делают Вашу работу проще или сложнее? Всеобщая механизация ручного труда и развитие дигитального ресторанного бизнеса с одной стороны и повальное ухудшение качества образования кондитеров с другой.

14. Какой интерес или какие потребности у Вас есть относительно бу дущего решения (разрабатываемого ПО)? Я надеюсь, что при разработке ПО найдутся количественные метрики для описания вкуса торта «Микадо».

1.4.2.2 Интервью с менеджером по персоналу

1. Имя. Смотрящих Артем Артемович.

2. Наименование организации. ООО «Ресторан армянской кухни имени Мартина Хайдеггера l'Être là».

3. Наименование структурного подразделения. Административный персонал.

4. Должность. Менеджер по персоналу.

5. Кому Вы непосредственно подчиняетесь? Исполняющему директору ресторана.

6. Каковы Ваши основные обязанности? Распределение обязанностей между сотрудниками, обучение персонала, подготовка заведения к открытию и закрытию, приём гостей — если в заведении нет хостес.

7. Что Вы в основном производите? Контроль и регуляция правильной работы и функционирования всех работников кухни и зала.

8. Для кого? Посетителей, контролирующих органов и директора организации.

9. Какие документы или какую информацию можно считать входящими, или необходимыми, для Вашей деятельности? Зарезервированные столики,

время резерва, ФИО посетителя, состав персонала, продукты в наличии.

10. Какие документы или какую информацию можно считать исходящими, или результатом Вашей деятельности? Прошедшие посетители, сумма заказа, официант, обслуживший гостя, результат и возможный отзыв.

11. Как измеряется успех Вашей деятельности? Лучшее отражение результата работы — отзывы клиентов и отсутствие потерянной выручки.

12. Какие проблемы влияют на успешность Вашей деятельности? Невыход персонала на работу, посетитель не явился/опоздал на бронированный столик.

13. Какие тенденции, если такие существуют, делают Вашу работу проще или сложнее? Соблюдение графика со стороны как работников, так и посетителей, отсутствие форсмажоров.

14. Какой интерес или какие потребности у Вас есть относительно будущего решения (разрабатываемого ПО)? Быстрые, насколько возможно, уведомления о заказах и бронировании, а также обновления статуса занятых столов и блюд, которые есть в наличии.

1.5 Анализ полученной информации

В результате анкетирования и интервьюирования всех заинтересованных лиц были сформулированы потребности заказчика относительно разрабатыва емого ПО. Среди них основными можно назвать:

1. Реализовать возможность оценивания готовых блюд.
2. Реализовать возможность оценивания обслуживания и отзывов.
3. Автоматизировать систему учёта свободных столов и работу уведомлений о бронировании.
4. Наладить меню работы ресторана на работу в реальном времени.

1.6 Вывод

В результате выполнения работы были сформированы навыки работы с реальными заказчиками программных систем; идентификации заинтересованных лиц и интервью с ними; анализа полученного материала; формулирования проблемы, ее актуальности и потребностей заинтересованных лиц.

2 ПРАКТИЧЕСКАЯ РАБОТА № 2 «СТРУКТУРНЫЙ АНАЛИЗ И ПРОЕКТИРОВАНИЕ ПРОГРАММНЫХ СИСТЕМ»)

2.1 Цель работы

Провести структурный анализ и построить DFD-диаграммы для заданной предметной области. Одна диаграмма должна представлять обобщенную работу системы, одна — детализацию одного из процессов (обе диаграммы должны включать 5-7 процессов). Построить STD-диаграмму системы.

2.2 Краткие теоретические сведения

Понятие жизненного цикла (ЖЦ) программного обеспечения (ПО) регламентирует основные процессы его разработки и эксплуатации. ЖЦ формулируется как модель создания и использования ПО, отражающая его различные состояния, начиная с момента возникновения необходимости в данном программном изделии и заканчивая моментом его полного выхода из употребления у всех пользователей.

В ЖЦ ПО выделяются следующие основные этапы:

1. Анализ требований и постановка задачи.
2. Анализ и исследование задачи, модели, проектирование.
3. Кодирование (программная реализация).
4. Тестирование и отладка.
5. эксплуатация и сопровождение.

ЖЦ образуется в соответствии с принципом нисходящего проектирова-

ния и, как правило, носит итерационный характер: реализованные этапы, начиная с самых ранних, циклически повторяются в соответствии с изменениями требований и внешних условий, введением ограничений и т.п. На каждом этапе ЖЦ порождается определенный набор документов и технических решений, при этом для каждого этапа исходными являются документы и решения, полученные на предыдущем этапе. Каждый этап завершается верификацией порожденных документов и решений с целью проверки их соответствия исходным.

Модель ЖЦ формулирует порядок исполнения этапов в ходе разработки, а также правила перехода от одного этапа к другому. На литературе по программной инженерии выделяют следующие модели как основные вехи в истории развития принципов анализа:

2.2.1 Каскадная модель

Каскадная модель была предложена Уинтсоном Ройсом в 1970 году. Эту модель принято называть классической и она предполагает, что переход на следующий этап после полного окончания работ по предыдущему этапу.

К преимуществам каскадной модели относят:

1. Четкую регламентацию выполнения этапов проекта.
2. Возможность оценки качества продукта на каждом из этапов.

В то же время у этой модели имеются существенные недостатки:

1. Модель не предусматривает обратных связей между этапами.
2. Она не соответствует реальным условиям разработки программного продукта.

2.2.2 Поэтапная модель с промежуточным контролем

Поэтапная модель с промежуточным контролем — итерационная модель разработки ПО с циклами обратной связи между этапами. Преимущество та-

кой модели заключается в том, что межэтапные корректировки обеспечивают меньшую трудоемкость по сравнению с каскадной моделью; с другой стороны, время жизни каждого из этапов растягивается на весь период разработки.

В этой модели предусмотрен промежуточный контроль за счет обратных связей. По итогам каждого из этапов возможен откат на предыдущий (-ие) шаги в случае, если выход этапа плохо верифицируем или в процессе разработки изменились исходные требования к системе. При работе с реальным проектом в классической каскадной модели обычно возникают проблемы при обнаружении недоработок и ошибок, допущенных на ранних этапах. Стройность процесса разработки нарушают также изменениями окружения, в котором разрабатывается ПО, такие как изменения требований заказчика, изменения политик разрабатывающей или эксплуатирующей организации, изменения отраслевых стандартов, появление конкурирующих продуктов и пр. Подобные нелинейности в процессе разработки учитываются в модели с промежуточным контролем. Недостатком поэтапной модели в сравнении с классической можно назвать существенное (до 10 раз) увеличение затрат на реализацию проекта.

2.2.3 Спиральная модель

Спиральная модель — делает упор на начальные этапы ЖЦ: анализ требований, проектирование спецификаций, предварительное и детальное проектирование. На этих этапах проверяется и обосновывается реализуемость технических решений путем создания прототипов.

Модель строится на четырех основных операциях, соответствующих квадрантам спирали:

1. Планирование — определяет цели разработки, формулирует варианты решения и накладывает ограничения на работу системы;
2. Анализ рисков — заключается в оценке рисков для различных вариантов решения;

3. Конструирование — непосредственная разработка продукта на новом уровне;

4. Оценивание — предполагает оценку результатов, достигнутых на этапе конструирования.

Каждый виток спирали соответствует поэтапной модели создания фрагмента или версии программного изделия, на нем уточняются цели и характеристики проекта, определяется его качество, планируются работы следующего витка спирали. Таким образом, углубляются и последовательно конкретизируются детали проекта и в результате выбирается обоснованный вариант, который доводится до реализации. При этом на этапе конструирование на каждом витке спирали может включать полный каскад классического жизненного цикла.

Специалистами отмечаются следующие преимущества спиральной модели:

1. Накопление и повторное использование программных средств, моделей и прототипов;

2. Ориентация на эволюционное развитие и модификацию ПО в процессе его проектирования;

3. Позволяет оценивать риски и издержки в процессе проектирования на каждом витке эволюции разработки.

Недостатками спиральной модели можно считать сложность оценки времени процесса разработки и необходимость более тесного вовлечения заказчика в процесс разработки.

Главная особенность индустрии ПО состоит в концентрации сложности на начальных этапах ЖЦ (анализ, проектирование) при относительно невысокой сложности и трудоемкости последующих этапов. Более того, нерешенные вопросы и ошибки, допущенные на этапах анализа и проектирования, порож-

дают на последующих этапах трудные, часто неразрешимые проблемы и, в конечном счете, приводят к неуспеху всего проекта.

2.3 Заданная предметная область

Антивирус-ревизор. Система фиксирует состояние заданных пользователем папок и в случае изменения их состояния (количество файлов, размер, дата создания или модификации, атрибуты) отмечает измененные элементы файловой системы как подозрительные. Проверка состояния папок осуществляется либо по расписанию, либо команде пользователя.

2.4 Выполнение практической работы

2.4.1 Диаграмма обобщенной работы системы

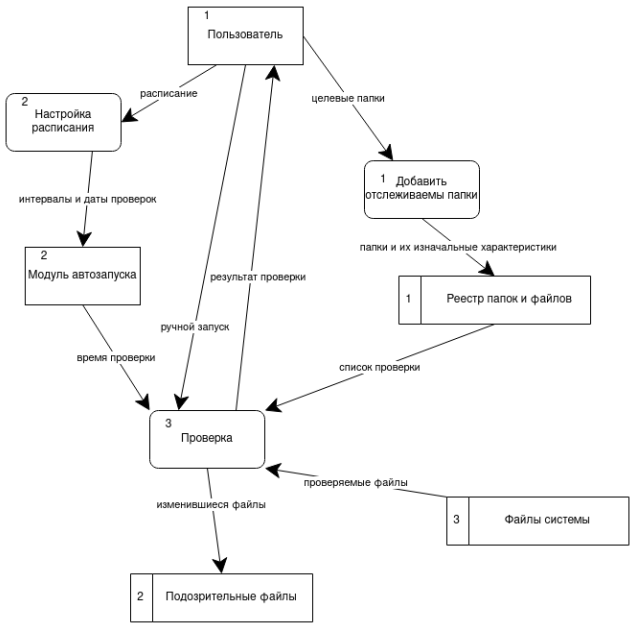

Рис. 2.1: Диаграмма обобщенной работы системы

2.4.2 Детализация процесса системы

Рис. 2.2: Детализация процесса системы

2.4.3 STD-диаграмма системы

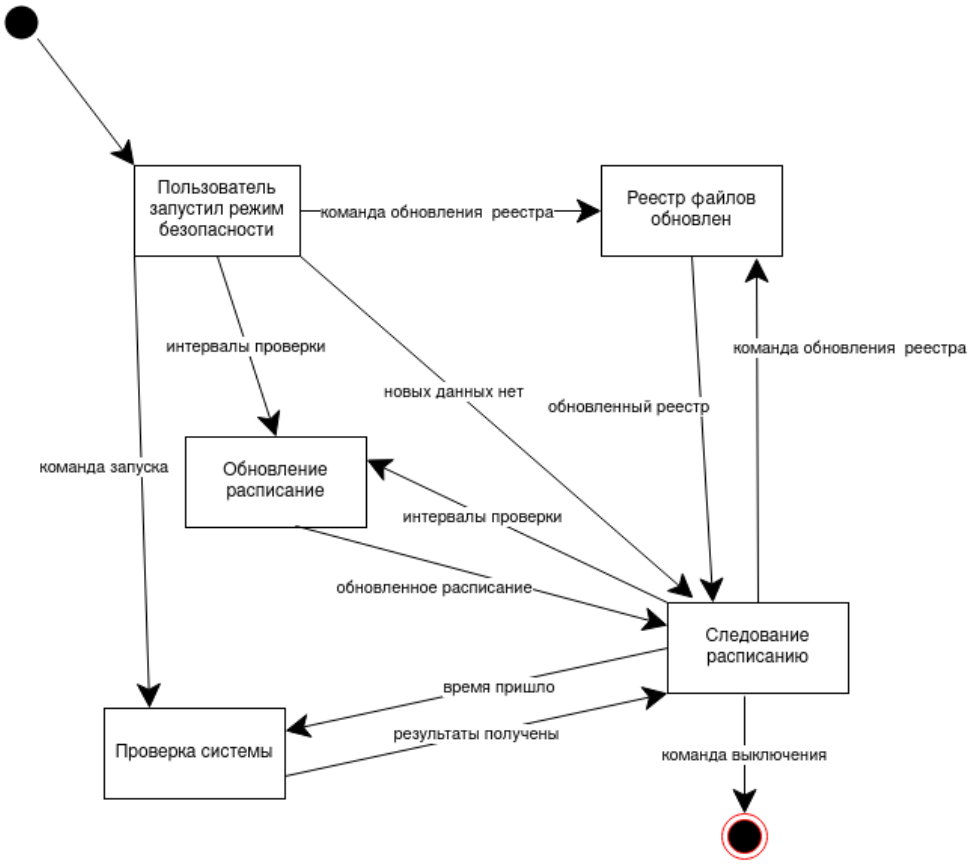

Рис. 2.3: STD-диаграмма системы

2.5 Вывод

Был проведен структурный анализ и постреены DFD-диаграммы для заданной предметной области.

3 ПРАКТИЧЕСКАЯ РАБОТА № 3 «МЕТОДОЛОГИЯ ФУНКЦИОНАЛЬНОГО МОДЕЛИРОВАНИЯ АРХИТЕКТУРЫ ИС»

3.1 Цель работы

Работа направлена на ознакомление с методологиями функционального моделирования IDEF0 и IDEF3, получение навыков по применению данных методологий для построения функциональных моделей на основании требований к информационной системе. В ходе выполнения работы необходимо выполнить обследование объекта автоматизации и составить обоснование необходимости создания ИС, согласно ГОСТ 34.601-90, выполнить анализ рисков, ознакомиться с основными методами и средствами для реализации и документирования аналитического отчета по проектированию ИС.

3.2 Постановка задачи

Выполнить проектировании следующей системы.

Предприятие занимается производством строительных материалов различных видов (цемент, кирпич, шифер, бетонные блоки). После выпуска партии готовой продукции, она передается на склад. Со склада производится отгрузка готовой продукции покупателю. При возникновении производственного брака, оформляется списание готовой продукции. Если продукция используется для производства нового изделия, ее возвращают со склада на переработку.

3.3 Видение проекта

Основными целями создания программного продукта для предприятия, занимающегося производством строительных материалов, являются:

1. Увеличение оборота предприятия за счет повышения продаж в дигитальной сфере.
2. Снижение объемов потери за счет введения электронной системы учета строительных материалов различных видов.

К проектным ограничениям можно отнести:

1. Мощности ЭВМ предприятия ограничены, на большинстве ЭВМ сотрудников предприятия установлено проприетарное ПО — оплата лицензий ПО превышает 5 % дохода.
2. Временные затраты на разработку зависят от пика продаж строительства на рынке — программная система дожна быть реализована в период между II и IV кварталом текущего года.
3. Помимо этого, бюджет на разработку ограничен и не может превышать 25000 USD, также средства необходимо направить на обучение работников цифровым навыкам.

3.4 Отчёт об обследовании предприятия

3.4.1 Организационная структура объекта

3.4.1.1 Итоговое словесное описание

Организационную структуру можно представить следующим образом:

Главную должность в структуре объекта занимает генеральный директор, ему непосредственно подчиняются производственный отдел, складской отдел и бухгалтерия.

У начальника производственного отдела в подчинении отдел переработки и отдел готовых продуктов.

У начальника складского отдела в подчинении отдел логистики и рабочий состав склада.

3.4.1.2 Схема

На рисунке 3.1 представлена оргструктура объекта.

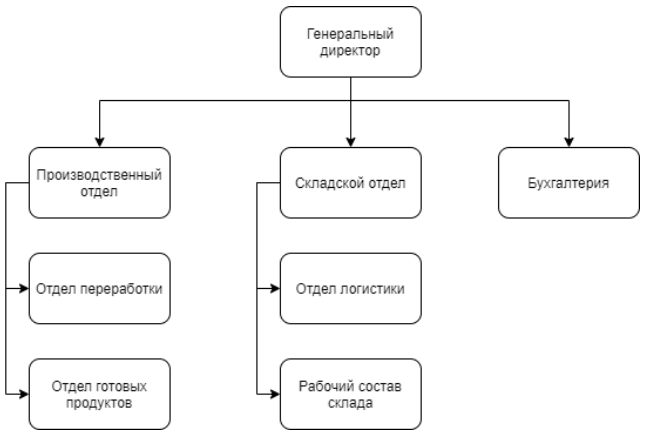

Рис. 3.1: Организационная структура объекта

3.4.2 Кадровая структура объекта

3.4.2.1 Итоговое словесное описание

Вертикальные связи.

Генеральному директору подчиняются начальник завода, начальник складского отдела, главный бухгалтер.

Начальнику завода подчиняется начальник смены.

Начальнику смены подчинются рабочие.

Начальнику складского отдела подчиняются главный логист и начальник склада.

Главному логисту подчиняется водитель фуры.

Начальнику склада подчиняется работник склада.

Главному бухгалтеру подчиняются бухалтеры.

Горизонтальные связи.

Уровень 1.

Между собой взаимодействуют начальник завода, начальник складского отдела и главный бухгалтер.

Уровень 2.

Между собой взаимодействуют начальник смены, главный логист, начальник склада и бухгалтеры.

3.4.2.2 Схема

На схеме ниже представлена кадровая структура объекта.

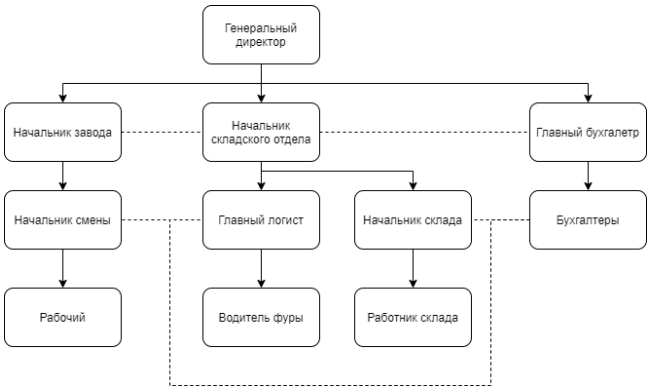

Рис. 3.2: Кадровая структура объекта

3.4.3 Основные бизнес-процессы объекта

3.4.3.1 Итоговое словесное описание

Основные бизнес-процессы объекта:

1. Отгрузка материалов со склада.
2. Создание продуктов.
3. Отгрузка на склад.
4. Отгрузка покупателю.
5. Оформление договора купли-продажи.
6. Переработка брака.

Отгрузка материалов со склада.

Нормативные документы — техника безопасности. Вход — заявка на производство. Выход — материалы. Исполнитель — работник склада.

Создание продуктов.

Нормативные документы — техника безопасности. Вход — материалы. Выход — готовая партия. Исполнитель — рабочие завода.

Отгрузка на склад.

Нормативные документы — техника безопасности. Вход — готовая партия. Выход — накладная. Исполнитель — работник склада.

Отгрузка покупателю.

Нормативные документы — техника безопасности и законы России. Вход — заявка на производство. Выход — список отгруженных товаров. Исполнитель — работник склада.

Оформление договора купли-продажи.

Нормативные документы — законы России. Вход — список отгруженных товаров. Выход — чек сделки. Исполнитель — бухгалтерв.

Переработка брака.

Нормативные документы — техника безопасности. Вход — партия для переработки. Исполнитель — рабочие завода.

3.4.3.2 Схема

На схеме ниже представлены основные бизнес-процессы объекта.

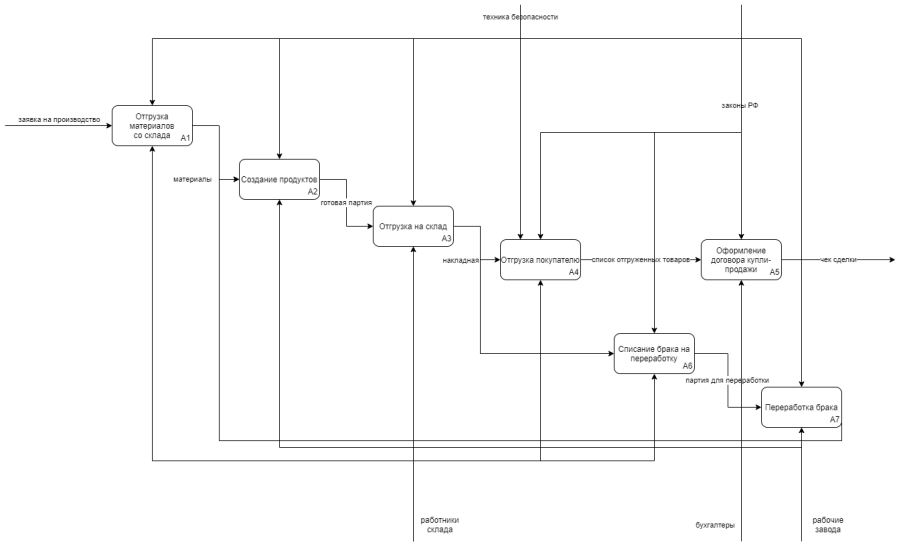

Рис. 3.3: Основные бизнес-процессы объекта

3.4.4 Декомпозиция основных бизнес-процессов объекта

Декомпозиция процесса отгрузки на склад.

Расфасовка в доставочные блоки.

Нормативные документы — техника безопасности. Вход — готовая партия. Выход — упакованный товар. Исполнитель — работники склада.

Распределение по складу.

Нормативные документы — техника безопасности. Вход — упакованный товар. Выход — новые поступления на склад. Исполнитель — работники склада.

Оформление новых товаров на складе.

Нормативные документы — законы РФ. Вход — поступление на склад. Выход — накладная. Исполнитель — работники склада.

3.4.4.1 Схема

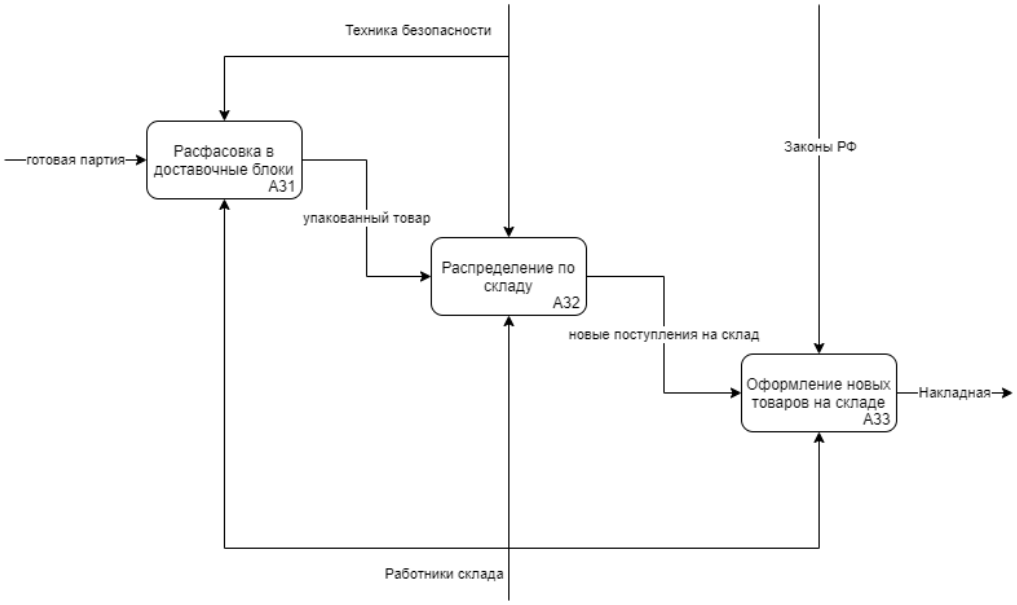

Рис. 3.4: Декомпозиция основных бизнес-процессов объекта

3.4.5 Основные проблемы в организации

Таблица 3.1 — основные проблемы организации.

Бизнес-процесс	Участники	Проблема
Оформление заявки	Бухгалтерия	Отсутствие автоматической обработки заявок
Отгрузка на склад	Начальник склада	Отсутствие автоматизированной системы учета товара
Списание брака на переработку	Начальник склада	Отсутствие системы учета товара
Оформление договора купли\продажи	Главный бухгалтер	Отсутствие системы формирования договора
Отгрузка покупателю	Главный логист	Отсутствие системы контроля доставки

3.5 Анализ рынка

3.5.1 Перечень похожих программных продуктов

Было найдено два похожих программных продукта:

1. МойСклад.
2. 1С: Розница.

3.5.2 Сравнительный анализ выбранного ПО

Таблица 3.2 — Сравнительный анализ выбранного ПО

Функция	МойСклад	1С: Розница
Автогенерация заказов	Есть	Есть
Поддержка маркировки товара	Есть	Нет
Складской учет	Есть	Есть
Банк и касса	Есть	Частично
Визуализация статистики	Нет	Есть

Были найдены следующие достоинства и недостатки систем:

Таблица 3.3 — МойСклад

МойСклад	
Достоинства	Недостатки
Удобный интерфейс	Отсутствие 3D-визуализации статистики
Быстрая поддержка развивающегося законодательства	Медленная работа службы поддержки
Есть мобильная версия	
Средняя стоимость	

Таблица 3.4 — 1С:Розница

1С: Розница	
Достоинства	Недостатки
Невысокая стоимость	Сложный интерфейс
Наличие планирования контактов с клиентами	Необходимость создания отчетов вручную
Удобное разграничение доступа	

3.5.3 Задачи, решаемые системой

Были выделены следующие задачи:

1. Контроль прогресса доставки товара.
2. Формирование договора.
3. Систематизирование складского учета.
4. Оформление поступлений и отгрузок со склада.
5. Формирование рабочей смены с учетом нагрузуки на работника и количества заказов.

На рисунке 3.5 представлена юзкейс-диаграмма системы.

Рис. 3.5: Use-case диаграмма системы

3.5.4 Круг заинтересованных лиц

Таблица 3.5 — Круг заинтересованных лиц

Эктор	Уровень доступа
Генеральный директор	Администратор
Начальник завода	Администратор
Начальник складского отдела	Администратор
Главный бухгалтер	Администратор
Начальник смены	Пользователь с расширенными возможностями
Главный логист	Пользователь с расширенными возможностями
Бухгалтеры	Пользователь
Водитель фуры	Пользователь
Рабочие	Пользователь
Складской работник	Пользователь

3.5.5 Границы использования системы

1. Мощности ЭВМ предприятия ограничены, на большинстве ЭВМ сотрудников предприятия установлено проприетарное ПО — оплата лицензий ПО превышает 5 % дохода.

2. Временные затраты на разработку зависят от пика продаж строительства на рынке — программная система дожна быть реализована в период между II и IV кварталом текущего года.

3. Помимо этого, бюджет на разработку ограничен и не может превышать 25000 USD, также средства необходимо направить на обучение работников цифровым навыкам.

3.5.6 Основные свойства системы

Таблица 3.6 — Основные свойства системы

Бизнес-процесс	Участники	Выгода для предприятия
Оформление договора	Бухгалтеры	Уменьшение временных затрат на обработку заявок
Отгрузка покупателю	Главный логист	Уменьшение времени согласования и сложности контроля процесса доставки
Отгрузка покупателю	Водитель фуры	Удобное формирование графика через ИС
Отгрузка на склад	Начальник склада	Удобство хранения и поиска документов продукта

3.6 Диаграммы IDEF0 для предметной области

На рисунках 3.6 и 3.7 представлены диаграммы IDEF0 для предметной области.

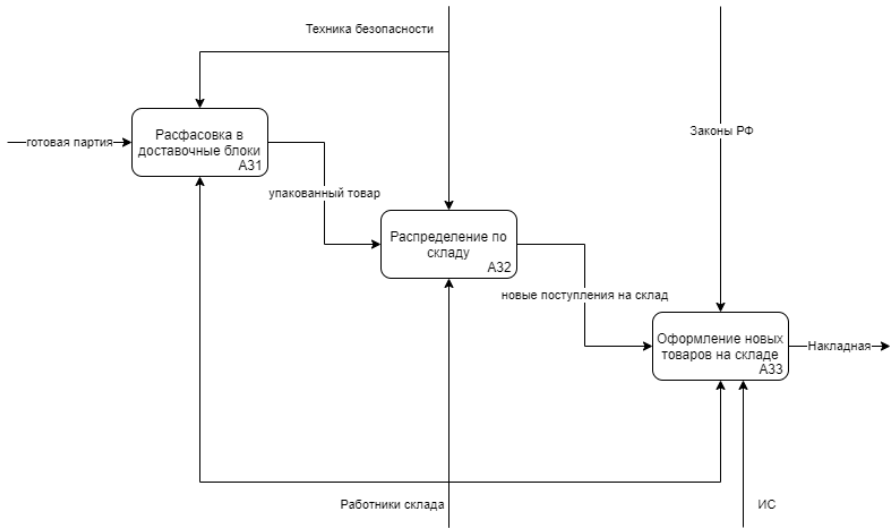

Рис. 3.6: Автоматизированная схема объекта

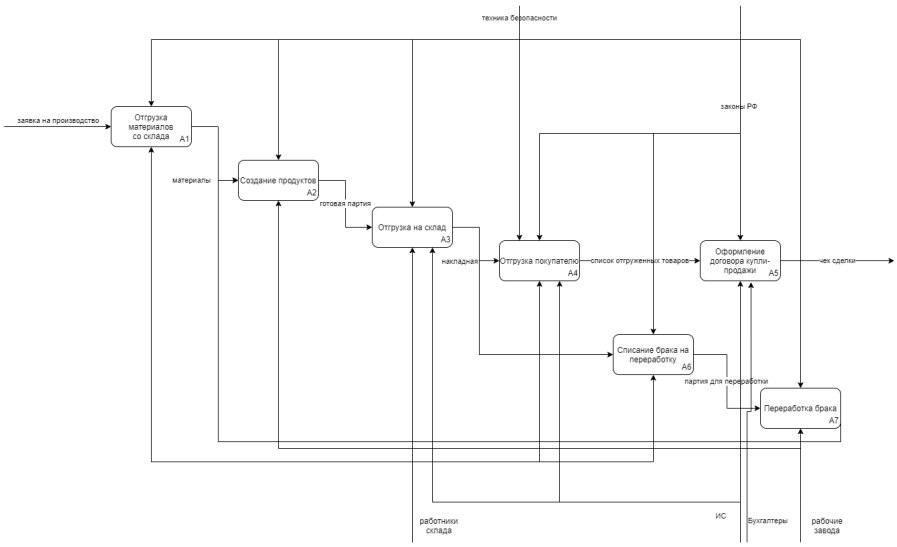

Рис. 3.7: Автоматизированная схема объекта

3.7 Заключение

В результате выполнения практической работы были получены навыки использования методологии функционального моделирования IDEF0, построены функциональные модели на основании требований к информационной системе агентства недвижимости. В процессе работы было выполнено обследование объекта автоматизации, составление обоснования необходимости информационной системы, выполнен анализ рисков и составлен аналитический отчет по проектированию информационной системы.

4 ПРАКТИЧЕСКАЯ РАБОТА № 4 «ВВЕДЕНИЕ В UML»

4.1 Постановка задачи

Используя открытые источники, необходимо найти и кратко описать методику использования модели «4+1» в методологии RUP, а именно указать, какой инструментарий и конкретные модели возможно использовать для отображения каждого из представления модели «4+1».

4.2 Логическое или концептуальное представление

Является объектной моделью проектирования (в том случае, если используется объектно-ориентированная модель проектирования).

Основной целью логического представления в данной методике является описание функциональных требований: что система должна выполнять в терминах конечных пользователей. Для этого представления используются различные абстрактные конструкции, такие как объекты и классы объектов. Для их иллюстрирования могут применяться диаграммы классов (в нотации языка UML) либо, например, диаграммы ”сущность-связь если в разработке приложения доминируют данные.

Для этого представления подойдет диаграмма Прецедентов (RUP).

Унифицированный процесс — это процесс, управляемый прецедентами, которые отражают сценарии взаимодействия пользователей. Фактически, это взгляд пользователей на программную систему снаружи. Таким образом, одним из важнейших этапов разработки, согласно RUP, будет этап определения требований, который заключается в сборе всех возможных пожеланий к работе

системы, которые только могут прийти в голову пользователям и аналитикам. Позднее эти данные должны будут систематизированы и структурированы, но на данном этапе в ходе интервью с пользователями и изучения документов, аналитики должны собрать как можно больше требований к будущей системе, что не так просто, как кажется на первый взгляд. Пользователи часто сами не представляют, что они должны получить в конечном итоге. Для облегчения этого процесса аналитики используют диаграммы прецедентов

Диаграмма представляет собой отражение действующих лиц (актантов), которые взаимодействуют с системой, и реакцию программных объектов на их действия. Актантами могут быть как пользователи, так и внешние агенты, которым необходимо передать или получить информацию. Значок варианта использования отражает реакцию системы на внешнее воздействие и показывает, что должно быть сделано для актанта.

Простота диаграммы прецедентов позволяет аналитикам легко общаться с заказчиками в процессе определения требований, выявлять ограничения, налагаемые на систему и на выполнение отдельных требований, такие, например, как время реакции системы, которые в дальнейшем попадают в раздел нефункциональных требований

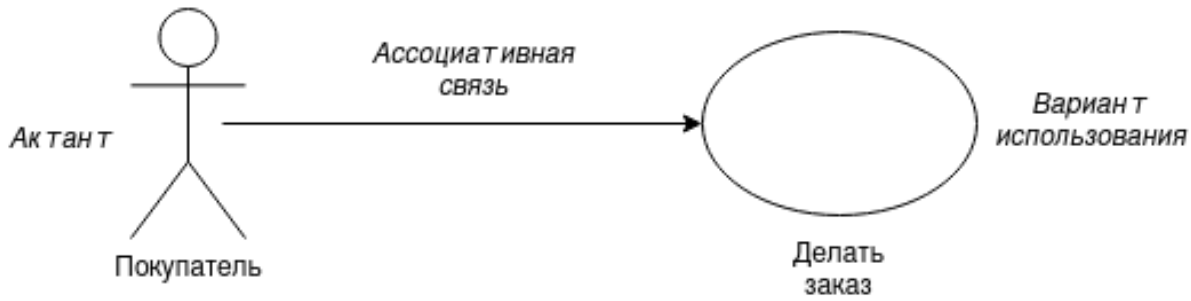

Рис. 4.1: Диаграмма прецедентов

4.3 Представление процесса

Описывает вопросы параллельного исполнения и синхронизации процессов.

Процессное представление учитывает некоторые нефункциональные требования к системе, включая производительность и доступность. С помощью этого представления рассматриваются такие аспекты, как одновременное выполнение и распределение процессов, интеграция системы, устойчивость к сбоям, а также то, как основные объекты абстракции, рассмотренные на уровне логического представления, соответствуют архитектуре процессов. Архитектура процессов может быть представлена на различных уровнях абстракции. На самом высоком уровне система рассматривается как набор независимо выполняемых сетей взаимодействующих между собой программ. На более низких уровнях рассматриваются процессы и задачи.

Для этого представления подойдет диаграмма Активности (RUP).

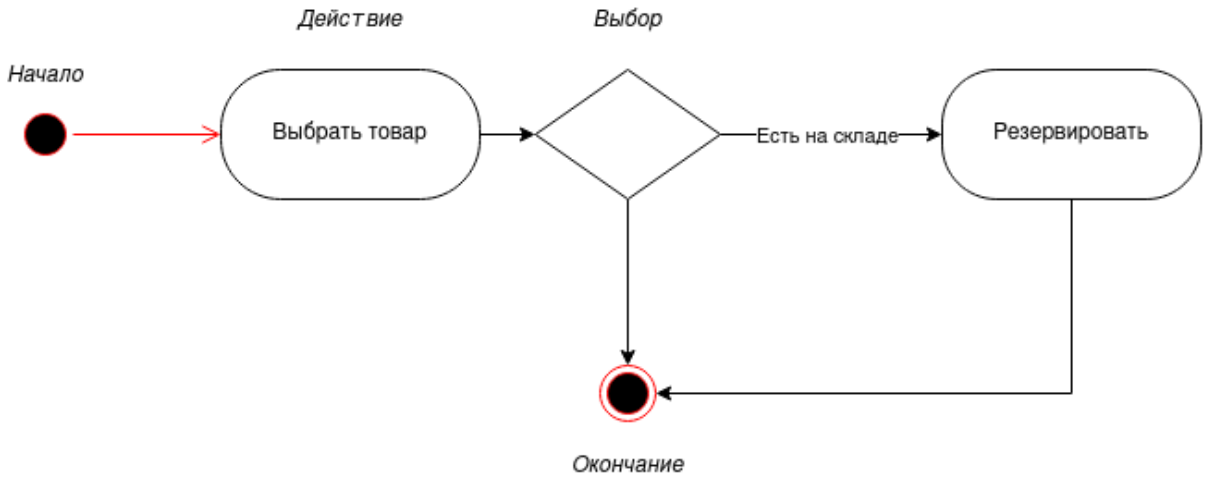

Рис. 4.2: Диаграмма Активности

4.4 Физическое представление

Описывает размещение программных компонент системы на аппаратных платформах и аспекты, связанные с физическим расположением системы.

Физическое представление, в основном, рассматривает нефункциональные требования, такие как доступность, надежность, устойчивость, производительность, масштабируемость. Этот уровень описывает распределение различных элементов – сетей, процессов, задач и объектов – по различным узлам (элементам аппаратного обеспечения, объединенным в сеть).

Для этого представления подходит реализация(RUP).

Основная задача процесса реализации – создание системы в виде компонентов – исходных текстов программ, сценариев, двоичных файлов, исполняемых модулей и т.д. На этом этапе создается модель реализации, которая описывает то, как реализуются элементы модели проектирования, какие классы будут включены в конкретные компоненты. Данная модель описывает способ организации этих компонентов в соответствии с механизмами структурирования и разбиения на модули, принятыми в выбранной среде программирования и представляется диаграммой компонентов .

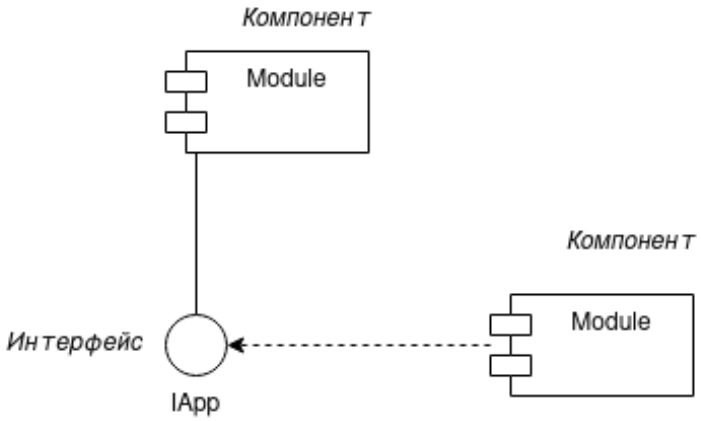

Рис. 4.3: Диаграмма реализации

4.5 Представление уровня разработки

Описывает статическую организацию программной системы в среде разработки.

Представление уровня разработки описывает фактическую организацию модулей системы, разделение ее на подсистемы, которые могут разрабатываться независимо.

Для этого представления подходит тестирование (RUP).

В процессе тестирования проверяются результаты реализации. Для данного процесса создается модель тестирования, которая состоит из тестовых примеров, процедур тестирования, тестовых компонентов, однако не имеет отображения на UML диаграммы.

4.6 Вывод

Используя открытые источники, нашли и кратко описали методику использования модели «4+1» в методологии RUP, указали, какой инструментарий и конкретные модели возможно использовать для отображения каждого из представления модели «4+1».

5 ПРАКТИЧЕСКАЯ РАБОТА № 5 «ДИАГРАММА СИСТЕМНОГО КОНТЕКСТА И ДИАГРАММА ВАРИАНТОВ ИСПОЛЬЗОВАНИЯ»

5.1 Цель работы

Цель работы: целью данной практической работы является ознакомление с формальной семантикой UML-диаграммы вариантов использования и применением диаграмм при проектировании архитектуры программного обеспечения, а также выявление особенностей их использования на практике.

5.2 Теоретическое введение

Использование UML, при правильной организации процесса моделирования, позволяет раскрыть особенности архитектуры разрабатываемого ПО с требуемым уровнем детализации для каждого из участников проекта. При разработке архитектур на ранних этапах полезно определить только те характеристики, которые на этом этапе будут с требуемой точностью определять основные концептуальные особенности разрабатываемой программной системы, тем самым, четко определив концептуальные границы разрабатываемого ПО, что позволит определить ту отправную точку, которая будет показывать, как программная система по своему объему вписывается в окружающий мир, а также представит общий вид программной системы. Для этих целей можно использовать, так называемую, системную контекстную диаграмму.

Диаграмма вариантов использования представляет проектируемую систему в виде множества сущностей или актёров, которые взаимодействуют с этой

системой посредством вариантов использования. Варианты использования описывают поведение моделируемой системы при различных условиях и воздействиях на неё со стороны заинтересованных лиц, которых называют актёрами. Как было отмечено выше, основной целью диаграмм вариантов использования является охват всех функций системного уровня, которые будут использовать эту разрабатываемую программную систему.

Каждый вариант использования должен относиться к конкретному классу пользователей системы, которые представлены актёром. Актёр может представлять любую сущность, например пользователь системы, сторонняя информационная система, интеллектуальный агент и т.д. При этом один пользователь может входить в разные классы одновременно. В качестве примера можно привести тот случай, когда администратор системы, обладающий правами администратора, может являться и пользователем системы, права которого ограничены. Диаграмма вариантов использования описывает отношения между участниками и различными вариантами использования, которые в нотациях UML называются ассоциациями. Отношения ассоциации не имеют направленности, а представляют собой лишь линии между актёрами и вариантами использования.

5.2.1 Диаграмма ассоциации

Диаграмма вариантов использования описывает отношения между участниками и различными вариантами использования, которые в нотациях UML называются ассоциациями. Отношения ассоциации не имеют направленности, а представляют собой лишь линии между актёрами и вариантами использования.

5.2.2 Диаграмма отношения обобщения

Помимо отношений ассоциации актёры, а также варианты использования могут быть связаны отношениями обобщения. Отношения обобщения всегда

направлены.

Отношение обобщения от актёра - администратора системы к актёру — владельца системы устанавливает то обстоятельство, что роли администратора имеют все функции и свойства присущие владельцам файлов, которые, в частности, позволяют им удалять любые файлы, которые они захотят.

5.2.3 Отношения включения

Вариант использования может зависеть от другого варианта, то есть поведение одного варианта может быть составной частью поведения другого варианта. Например, вариант использования «удалить файл» в обязательном порядке будет включать в себя вариант использования «проверить права доступа». Направление пунктирной линии со стрелкой и надписью «INCLUDE» идёт от основного варианта использования к включённому варианту использования.

5.2.4 Отношения расширения

Вариант использования может, при определённых обстоятельствах, вызывать другой вариант использования, например для предоставления некоторой альтернативы. В таком случае говорят, что данная зависимость является расширенной. Направление пунктирной линии со стрелкой и с надписью «EXTEND» идёт от расширения к расширенному варианту использования.

5.3 Цель

Используя средства UML, необходимо разработать диаграмму использования для системы бронирования билетов на автовокзале. Разрабатываемая система должна обеспечивать возможность покупки билетов через электронный киоск при предъявлении паспорта (паспорт сканируется при начале работы с

киоском). Актёры - пассажир со следующими атрибутами: ФИО, адрес регистрации, номер и серия паспорта, город прибытия. Операции: бронировать билет.

5.4 Выполнение работы

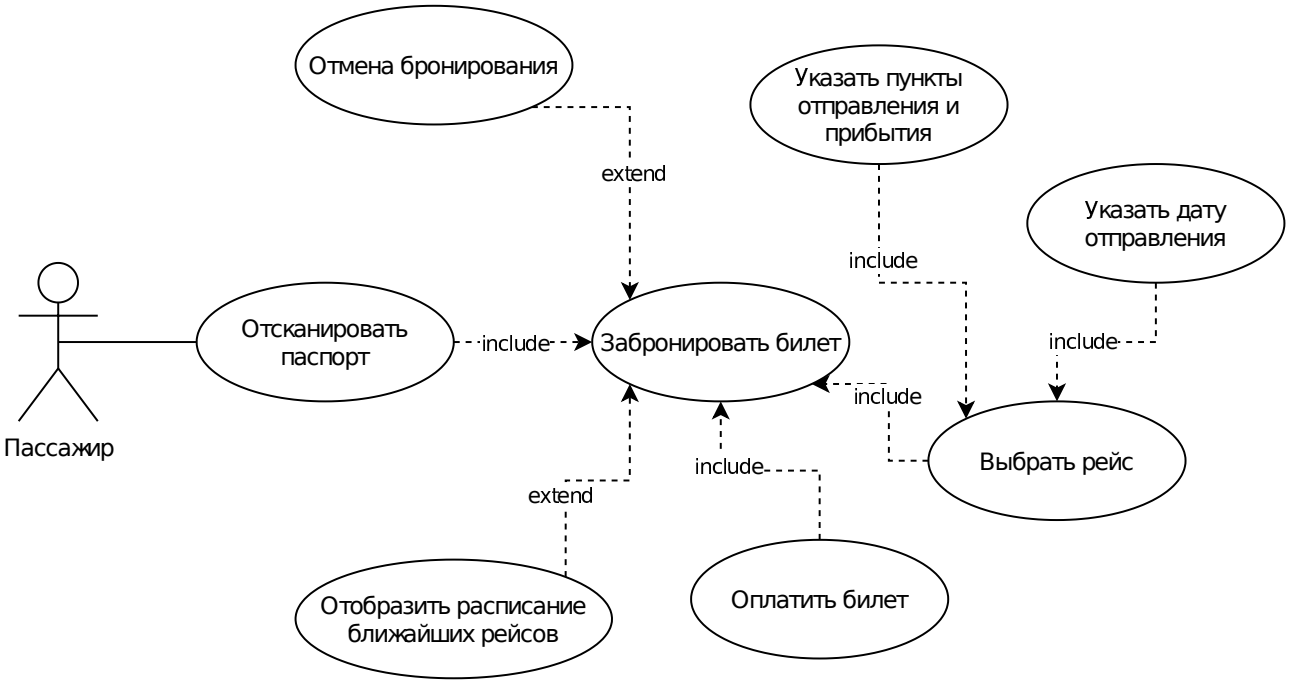

Рис. 5.1: Организационная структура объекта

В ходе выполнения работы была реализована диаграмма 5.1.

Опишем элементы системы.

5.4.1 Актор

Актор — пассажир (человек) с ФИО, адресом регистрации, номером и серией паспорта, городом прибытия. Отношения ассоциации не имеют направленности, а представляют собой лишь линии между актёрами и вариантом использования — элементом «Отсканировать паспорт».

5.4.2 Отсканировать паспорт

Точка входа в систему.

От неё исходит отношение включения к варианту использования «Забронировать билет».

5.4.3 Забронировать билет

В нее входят варианты использования с отношением включения «Оплатить билет» и «Выбрать рейс». В нее входят варианты использования с отношением расширения «Отобразить расписание ближайших рейсов» и «Отмена бронирования».

5.4.4 Выбрать рейс

Третья опорная точка системы, в нее включены варианты использования с отношением включения «Указать пункты отправления и прибытия», «Указать дату отправления».

5.5 Вывод

Ознакомились с формальной семантикой UML-диаграммы вариантов использования и применением диаграмм при проектировании архитектуры программного обеспечения, а также выявлением особенностей их использования на практике.

6 ПРАКТИЧЕСКАЯ РАБОТА № 6 «ДИАГРАММЫ КОНЕЧНОГО АВТОМАТА И ДИАГРАММЫ СОСТОЯНИЙ»

6.1 Цель работы

Целью данной практической работы является ознакомление с семантикой диаграммам схем конечных автоматов и диаграмм состояний в контексте UML.

6.2 Теоретические сведения

Диаграмма конечных автоматов предназначена, прежде всего, для описания процесса изменения состояния конкретного объекта, а если быть точным для описания различных состояний, в которых может находиться объект, и переходы между этими состояниями. Диаграмма конечных автоматов позволяет описать поведение конкретного объекта модели (реакции объекта на возможные внешние и внутренние изменения), посредством описания всех возможных последовательностей состояний и переходов в рамках конкретной модели разрабатываемой системы. Основным достоинством диаграммы состояния является то, что она позволяет описать динамическое поведение сущностей (объектов и т.д.) при воздействии на эти сущности внешнего и внутреннего воздействия, как было выяснено выше, на всем этапе жизненного цикла данной сущности.

Наиболее часто на практике данные диаграммы используются в объектно-ориентированной парадигме для описания динамического поведения конкретного экземпляра класса (объект), но при этом данные диаграммы могут совместно использоваться с различными диаграмми UML в комбинациях, что поз-

воляет, например, в случае комбинации с диаграммами вариантов использования описать функциональные ограничения в поведении актёра.

Диаграммы конечного автомата UML изображают различные состояния, в которых может находиться объект, и переходы между этими состояниями. Фактичес в других языках моделирования этот тип диаграммы обычно называют диаграммой перехода состояний или даже просто диаграммой состояний. В нотации UML используются диаграммы состояний Харела, которые позволили обойти ограничение классических диаграмм состояний, которое связано со снижением удобочитаемости классических диаграмм при увеличении количества узлов и переходов между узлами для всех систем. При этом диаграмма состояний Харела эквивалентна классической диаграмме состояний.

Диаграммы состояний требуют, чтобы описываемая система состояла из конечного числа состояний и позволяют, используя граф специального вида (ориентированный граф), в которых вершины (узлы) обозначают состояния, а дуги обозначают переходы состояний, графически описать представление конечных автоматов.

В нотациях UML состояния представлены в виде скругленных прямоугольников, которые помечены именами состояний. Переходы, обозначены стрелками, помечены запускающими событиями, за которыми необязательно следует список выполненных действий. Начальный переход происходит из сплошного круга и определяет начальное состояние.

6.2.1 Состояния и композитные состояния

Состояние моделирует ситуацию, при которой выполняется некоторое инвариантное условие. В большинстве случаев это условие не определено явно.

Выделяют следующие типы состояний:

1. Простое состояние (не имеет внутренних вершин или переходов)
2. Составное состояние (содержит хотя бы один регион)

3. Субмашинное состояние

6.2.2 Внутреннее поведение

Внутреннее поведение содержит список внутренних действий или действий состояния (do), которые выполняются, пока элемент находится в этом состоянии. Каждое из этих внутренних действий записывается в формате <метка-действия ’/’ выражение-действия>. Метка действия часто представляет собой то условие при которых данное выражение-действие будет выполнено.

Несколько меток зарезервированы для специальных целей и не могут использоваться в качестве имен событий. Ниже перечислены зарезервированные ярлыки действий:

- entry
- do
- exit

6.2.3 Начальное и конечное состояние

Начальное псевдосостояние представляет собой отправную точку, в которой находится объект в начальный момент времени и служит для отображения той графической области, которая будет являться областью изменения состояния. При этом, в данной области не может быть более одной начальной вершины.

Конечное состояние — это особый вид состояния, которое не содержит никаких внутренних действий.

6.2.4 Переход и виды переходов

Переход представляет собой направленные отношения между исходной и целевой вершинами (состояния). В зависимости от отношения к вершине в

общем случае выделяют 3 основных вида переходов:

1. external — переход выходит из своей исходной вершины (выполняет действие выхода из исходного состояния)
2. local —- переход не выходит из содержащего его состояния (действие выхода из содержащего состояния не будет выполнено), при этом локальный переход может существовать только в составном состоянии
3. internal переход является частным случаем локального перехода с одинаковыми исходным и целевым состояниями

6.3 Задание на практическую работу

Используя средства UML, необходимо разработать диаграмму состояний для класса «Заказ» из предыдущей практической работы.

6.4 Выполнение практической работы

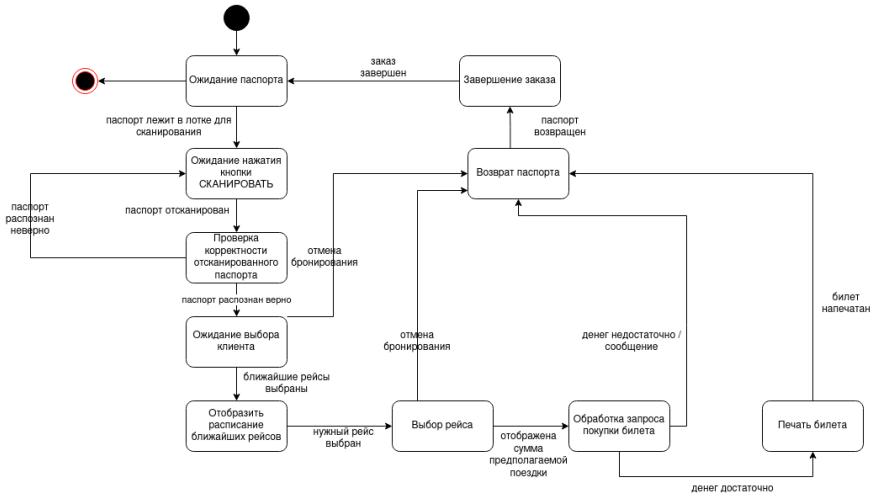

Рис. 6.1: Диаграмма состояний объекта

6.5 Вывод

Ознакомились с семантикой диаграммам схем конечных автоматов и диаграмм состояний в контексте UML.

7 ПРАКТИЧЕСКАЯ РАБОТА № 7 «ДИАГРАММА ДЕЯТЕЛЬНОСТИ»

7.1 Цель работы

Целью данной практической работы является ознакомление с семантикой диаграммам деятельности. В ходе работы студенты должны освоить особенности применения диаграмм деятельности для разработки архитектуры ПО.

7.2 Теоретические сведения

диаграмма деятельности может, при определённых условиях, считаться частным случаем диаграмм состояний, рассмотренных в прошлой практической работе. В отличии от диаграммы состояний, диаграмма деятельности не сосредоточена только на том, чтобы показать для чего система разрабатывается и как она будет использоваться заинтересованными лицами. Основная задача диаграммы деятельности показать какими атомарными действиями будет достигнута та цель, которую формирует каждое заинтересованное лицо. То есть, диаграмма деятельности является, своего рода, поведенческой спецификацией, которая может быть отождествлена с блок-схемами/сетями Петри в объектно-ориентированном контексте. Диаграммы деятельности позволяют прояснить, как координируется деятельность по предоставлению услуги, как взаимосвязаны события, то есть даёт целостную картину бизнес-процессов предметной области. Таким образом, диаграмма деятельности может быть использована не только для описания всех бизнес-процессов системы, но и для описания требований, предъявляемых к разрабатываемой системе. На практике наиболее распространено применение диаграмм деятельности для описания на ранних эта-

пах проекта всей бизнес логики, а также для описания системных функций. Исходя из вышеназванного, можно сделать следующие выводы:

1. Диаграммы деятельности являются эквивалентом блок-схемам/сетей Петри ().
2. Данный вид поведенческих диаграмм описывает динамический процесс работы системы, путём описания (моделирования) потока управления.

Стоит учесть, что стандарт UML не описывает всю семантику всех элементов диаграммы деятельности. В зависимости от типа решаемой задачи, используются различные комбинации базовых и расширенных элементов, для достижения требуемого уровня детализации.

7.3 Элементы

Базовыми элементами диаграммы деятельности являются деятельность, которая используется для представления набора действий и которая представляется на диаграмме, как прямоугольник с закругленными углами, и поток управления, который показывает последовательность выполнения деятельности, который отображается в виде направленной стрелки. Обычно на практике, одна деятельность интерпретирует один шаг алгоритма [2].

К базовым элементам диаграммы деятельности также относятся начальное состояние, которое отображает начало действий (потоки) и конечное состояние деятельности, которое останавливает весь поток управления данной деятельности.

Также выделяют конечный узел потока, который обозначается как маленький круг с X внутри. Конечный узел потока останавливает только данный поток, не влияя при этом на другие потоки. В некоторых случаях поток может не завершится, а приостановиться на время, в результате некоторого события

или выполняться в определённые временные рамки. Примером такой ситуации является чтение данных из файла по таймеру, что показано на рисунке 16. У данного события нет входящих рёбер, поэтому оно включено по умолчанию, так как у него есть одно действие.

В некоторых случаях поток может не просто завершится, а завершиться в результате какого-то события. Обычно ситуацию обрыва потока обозначается в виде молнии.

Также выделяют поток объектов. Поток объектов относится к созданию и модификации объектов по видам деятельности. Стрелка потока объекта от действия к объекту означает, что действие создаёт объект или влияет на него. Стрелка потока объекта от объекта к действию указывает, что действие использует объект.

Расширенными элементами диаграммы активности также являются такие сущности, как отправленные и полученные сигналы, узел принятия решения (ветвление), узел слияния, узел узел соединения, отправленные и полученные сигналы, дорожки (Swimlane).

Сигналы представляются обычно, как действия которые могут быть получены системой извне (из другой системы). Они обычно появляются в парах отправленных и полученных сигналов, потому что состояние не может измениться до тех пор, пока не будет получен ответ на сигнал. Данный механизм похож на синхронные сообщениям на диаграмме последовательности. В качестве примера можно рассмотреть ситуацию, когда моделируется система оплаты для интернет магазина. В данном случае отправляется сигнал запроса на оплату, затем происходит ожидание подтверждения, посредством получения сигнала подтверждения оплаты. Механизм ожидания сигнала подтверждения включается только после того, как будет отправлен сигнал запроса на оплату. Данное расширение было добавлено в версии UML 2.0.

Слияние событий объединяет несколько потоков, которые не являются одновременными.

Узел fork (разделения) потоков используется для разделения поведения на набор параллельных потоков деятельности.

Узел объединения потоков используется для объединения нескольких потоков в один поток деятельности.

Совместное использование улов слияния и разделения часто называют синхронизацией.

«Плавательные дорожки»/Swimlanes представляют собой графические линии, которые разделяют деятельность по различным категориям. Swimlanes группируют связанные действия в одну колонку. В практической деятельности дорожки применяются, как для моделирования бизнес-процессов, так и для описания фукциональных характеристик разрабатываемой системы. Часто дорожки используются для того, чтобы показать роль конкретного пользователя или В общем виде дорожки могут быть сформированы как по горизонтали, так и по вертикали.

7.4 Задание

Вам, как архитектору программного обеспечения, поступила задача на разработку архитектуры информационной системы для автоматизации библиотечной деятельности, которая позволит вести централизированный учёт, выданных книг, регистрационных карточек посетителей, а также контролировать новые поступления в библиотеку. Необходимо разработать диаграмму деятельности по предоставлению услуги выдачи книги новому пользователю библиотеки. Для этого необходимо изучить предметную область, а также все бизнес процессы библиотеки. По желанию, можно расширить функционал системы, например, добавив возможность вносить в библиотечный фонд документы поступающие по почте для их хранения в библиотечных фондах, и нарисовать соответствующие диаграммы деятельности.