КАЗАНСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ
ИНСТИТУТ ВЫЧИСЛИТЕЛЬНОЙ МАТЕМАТИКИ И ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ
Кафедра информационных систем

О.А. НЕВЗОРОВА

МЕТОДОЛОГИИ ПРОЕКТИРОВАНИЯ ИНФОРМАЦИОННЫХ СИСТЕМ

Учебно-методическое пособие

КАЗАНЬ
2025

УДК 004.4'2
ББК 16.33
Н40

Принято на заседании учебно-методической комиссии
Института Вычислительной математики и информационных технологий
(протокол № 6 от 28 февраля 2025 года)

Рецензенты:

кандидат технических наук, доцент **Галимянов А.Ф.**

кандидат технических наук, доцент **Дроздиков В.А.**

Невзорова О.А.

Н40 Методологии проектирования информационных систем: учебно-методическое пособие / О.А. Невзорова. – Казань: Казанский федеральный университет, 2025. – 54 с.

Учебно-методическое пособие знакомит с современными технологиями проектирования информационных систем, моделями, методами и средствами решения функциональных задач и организации информационных процессов. Рассматриваются вопросы, связанные с изучением организационной, функциональной и математической структуры процесса проектирования информационной системы и базовых информационных процессов. Приводятся описания лабораторных работ, выполняемых в рамках данного курса, что способствует формированию практических навыков проектирования информационных систем.
.

УДК 004.4'2
ББК 16.33

ВВЕДЕНИЕ

Тенденции развития современных информационных технологий приводят к постоянному возрастанию сложности информационных систем (ИС), создаваемых в различных областях экономики.

Современные крупные проекты ИС характеризуются, как правило, сложностью описания (имеют достаточно большое количество функций, процессов, элементов данных и различных взаимосвязей между ними), что требует тщательного моделирования и анализа данных и процессов.

Цель создания методологии построения информационных систем заключается в регламентации процесса проектирования ИС и обеспечении управления этим процессом с тем, чтобы гарантировать выполнение требований, как к самой ИС, так и к характеристикам процесса разработки.

Методология реализуется через конкретные технологии и поддерживающие их стандарты, методики и инструментальные средства, которые обеспечивают выполнение процессов жизненного цикла информационных систем.

Можно сформулировать следующий ряд общих требований, которым должна удовлетворять технология проектирования, разработки и сопровождения информационных систем:

- поддерживать полный жизненный цикл информационной системы;
- обеспечивать гарантированное достижение целей разработки системы с заданным качеством и в установленное время;
- обеспечивать возможность разделения крупных проектов на ряд подсистем – декомпозицию проекта на составные части, разрабатываемые группами исполнителей ограниченной численности, с последующей интеграцией составных частей;

• технология должна обеспечивать возможность ведения работ по проектированию отдельных подсистем небольшими группами (3–7 человек);

• обеспечивать минимальное время получения работоспособной системы;

• предусматривать возможность управления конфигурацией проекта, автоматического выпуска проектной документации и синхронизацию ее версий с версиями проекта;

• обеспечивать независимость выполняемых проектных решений от средств реализации системы – системы управления базами данных, операционной системы, языка и системы программирования.

Цель данного курса – получение теоретических знаний о современных технологиях проектирования информационных систем (ИС), моделях, методах и средствах решения функциональных задач и организации информационных процессов, изучение организационной, функциональной и математической структуры процесса проектирования информационной системы и базовых информационных процессов, формирование практических навыков проектирования информационных систем.

Задачи курса:

• получение теоретических сведений о современных методологиях, технологиях и инструментальных средствах анализа и проектирования информационных систем;

• формирование умений и навыков в определении цели, постановки задач проектирования, проведение анализа объекта проектирования, подготовки технических заданий и технико-экономического обоснования проектных работ;

• изучение методологий, технологий и нотаций описания этапов проектирования информационных систем;

• получение умений и навыков использования инструментальных средств проектирования информационных систем на различных этапах жизненного цикла информационной системы.

Обучающийся, освоивший данную дисциплину,

должен знать:

- современные методологии проектирования информационных систем, их основные характеристики и требования;

должен уметь:

- применять различные методологии в процессе проектирования;

должен владеть:

- навыками проведения анализа предметной области и функциональных требований, построения необходимых диаграмм;

должен демонстрировать способность и готовность:

- использовать современные технологии при проектировании информационных систем.

Содержание дисциплины включает следующие темы и рассматриваемые в рамках темы вопросы.

Тема 1. Методология функционального моделирования SADT.

Три класса структурных моделей. Функциональная модель. Информационная модель. Динамическая модель. Процесс моделирования по методологии SADT. Этапы: сбор и анализ информации о предметной области, документирование полученной информации, моделирование (IDEF0), корректура модели в процессе итеративного рецензирования.

Тема 2. Методология RAD.

История принципов методологии проектирования RAD. Особенности методологии RAD. Стадии: моделирование информационных потоков между бизнес-функциями, моделирование данных, преобразование объектов данных, обеспечивающих реализацию бизнес-функций, генерация приложений, тестирование и объединение.

Тема 3. Методология RUP.

История методологии RUP. Процессы и стадии RUP. Принципы RUP. Анализ и построение модели прецедентов. Компонентная архитектура. Подходы RUP: итерационный и инкрементный (наращиваемый), построение системы на базе архитектуры информационной си-

стемы, планирование и управление проектом на основе функциональных требований к информационной системе.

Тема 4. Диаграммы потоков данных DFD.

Диаграммы потоков данных. Процессы, входные и выходные данные. Потоки данных. Дуги, разделение потока данных на части, слияние объектов. Хранилища данных. Агрегатные хранилища данных. Потоки управления. Контекстные диаграммы, иерархия контекстных диаграмм. Детализации процессов, альтернативы. Условное вхождение, итерации.

Тема 5. Методология объектного проектирования на языке UML (UML-диаграммы).

Унифицированный язык моделирования (Unified Modeling Language – UML). Диаграммы классов. Классы, атрибуты, операции классов. Зависимость, связь-сообщение, суперкласс, подкласс. Ассоциация, кратность. Композитные ассоциации. Ограничения целостности OCL (Object Constraints Language). Инварианты класса.

Тема 6. Архитектурный подход к проектированию информационных систем.

Понятие архитектуры информационных систем. Модель корпоративной архитектуры. Бизнес архитектура (Business architecture). ИТ-архитектура (Information Technology architecture), архитектура данных (Data architecture), программная архитектура (Software architecture), техническая архитектура (Hardware architecture).

Данное пособие содержит лабораторные работы по темам 1–6. По результатам выполнения заданий лабораторных работ оформляется отчет. Лабораторная работа засчитывается после защиты отчета. При сдаче отчета студент должен продемонстрировать полученные знания и умения, формулировать ответы на вопросы по теме лабораторной работы.

Лабораторная работа № 1

Методология функционального моделирования работ SADT.
Создание диаграммы верхнего уровня (4 часа)

Постановка задачи. Разработать функциональную модель системы на основе методологии SADT. Разработка должна выполняться поэтапно, в соответствии с определенными заданиями. В задании 1 очерчивается контекст задачи, в задании 2 определяется цель и точка зрения модели, в задании 3 создается диаграмма верхнего уровня, в задании 4 – обобщенная диаграмма верхнего уровня.

Теоретические сведения

Методология SADT (Structured Analysis and Design Technique) была создана и опробована на практике в период с 1969 по 1973 гг. Автором методологии SADT является Дуглас Росс.

Предназначена для моделирования систем на основе принципов структурного анализа. Методология предлагает графический язык проектирования систем, в котором сочетаются декомпозиция и иерархическое упорядочение и для обозначения составляющих системы используется графическая конструкция, называемая SA-блок.

Методология SADT может использоваться для моделирования широкого круга систем и определения требований и функций, а затем для разработки системы, которая удовлетворяет этим требованиям и реализует эти функции. Для уже существующих систем методология SADT может быть использована для анализа функций, выполняемых системой, а также для указания механизмов, посредством которых они осуществляются. Методология SADT может быть направлена как на описания функций, выполняемых системой, так и на описание объектов, составляющих систему.

В первом случае методология SADT представляет собой совокупность методов, правил и процедур, предназначенных для построения функциональной модели объекта какой-либо предметной обла-

сти. Функциональная модель SADT отображает функциональную структуру объекта, т.е. производимые им действия и связи между этими действиями.

Во втором случае методология SADT представляет собой совокупность методов, правил и процедур, предназначенных для описания объектов, входящих в систему, их свойств и взаимосвязей между ними.

Сущность структурного подхода к моделированию систем

Система разбивается на функциональные подсистемы, которые, в свою очередь, делятся на подфункции, подфункции – на задачи и т.д. до конкретных процедур. Для любой системы определяющим является ее функциональное содержание, так как оно определяет ее основные свойства. Поэтому в основе функционального моделирования лежит функциональное содержание системы, в качестве отношений между функциями рассматривается информация об объектах, связывающих эти функции.

Начало моделирования в SADT означает создание диаграмм АО и А–0, которые затем могут быть отрецензированы. Эти две диаграммы полностью рассказывают все об изучаемой системе с минимальной степенью детализации. Создавая их, аналитик предпринимает начальную попытку декомпозировать систему и затем обобщить полученную декомпозицию. Декомпозиция (диаграмма АО) освещает наиболее важные функции и объекты системы. Объединение (диаграмма А–0) трактует систему как «черный ящик», дает ей название и определяет наиболее важные входы, управления, выходы и, возможно, механизмы.

SADT–модель начинается с очерчивания границ системы, определения цели и точки зрения модели и создания диаграмм верхнего уровня.

Рассмотрим этапы, которые выполняют SADT–аналитики в начале создания функциональной модели на примере разработки функциональной модели «Питание семьи».

Задание 1. Создание очерченного контекста для заданной модели.

Действия

1. Вспомните основные понятия SADT-моделирования, как они применяются к очерчиванию объекта моделирования.

2. Начните составлять список всех основных предметов, которые, по вашему мнению, являются частью системы. Дайте свободу ассоциациям. На этом этапе не беспокойтесь о точности.

3. Теперь оцените исходный список критически. Вычеркните названия, не относящихся к системе объектов. Если есть возможность, объединяйте названия в группы, проводя соединительные линии или обводя слова кружками. Добавляйте новые названия по мере развития ваших идей.

4. Теперь проделайте то же самое для функций системы. Для перечисления функций пользуйтесь списком данных, затем оцените новый список. Вычеркните те названия, которые не входят в систему. Группируйте сходные функции, соединяя их названия линиями или обводя кружками. Меняйте список данных по мере постижения работы системы.

5. Остановитесь, когда вы перечислите достаточное для создания диаграммы число объектов и функций.

Образец для модели «Питание семьи» представлен на рисунке 1.

Задание 2. Определение цели и точки зрения модели.

Действия

1. Составьте множество вопросов, на которые должна отвечать модель. Уточните это множество, определив, кто задает вопросы. Запишите по крайней мере 5–10 вопросов. Затем задайте степень точности ответа на каждый из них.

2. С помощью этого набора вопросов определите, как будет использоваться модель. Если вы не можете сформулировать, как она будет использоваться, попробуйте записать еще вопросы или попытайтесь вообразить, кто будет применять модель. В одном предложе-

нии сформулируйте, как она будет использоваться. Это станет целью модели.

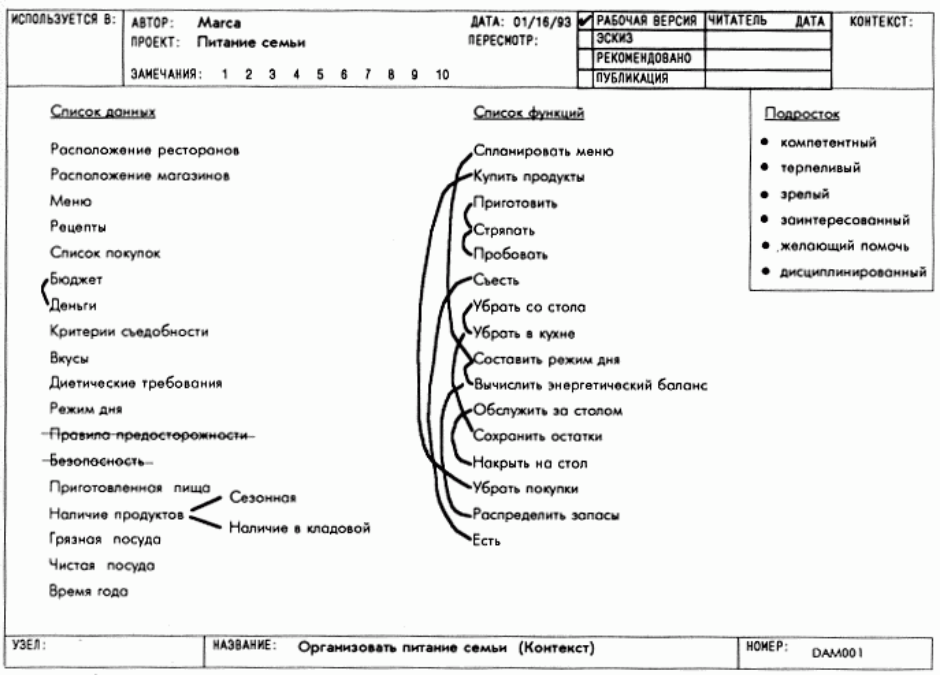

Рис. 1. Контекст модели

3. Теперь решите, кто будет рассказывать о модели. Составьте список кандидатов, чтобы убедиться, что вы выбрали самую подходящую точку зрения. Выберите из всех того, кто сможет ответить на большинство вопросов. Его точка зрения станет точкой зрения модели.

Образец для модели «Питание семьи» представлен на рисунке 2.

Задание 3. Построение диаграммы верхнего уровня АО.

Исходное содержание диаграммы АО обеспечивают списки данных и функций.

Действия

1. Объедините 3–6 функций из списка функций очерченного контекста и расположите их по порядку важности. Нарисуйте и назовите блоки по одному для каждой функции в соответствии с порядком доминирования.

2. Нарисуйте и пометьте внутренние дуги, представляющие ограничения для работы каждого блока, используя составленный список данных. Чтобы сделать это, проанализируйте функцию каждого блока и задайте соответствующий вопрос.

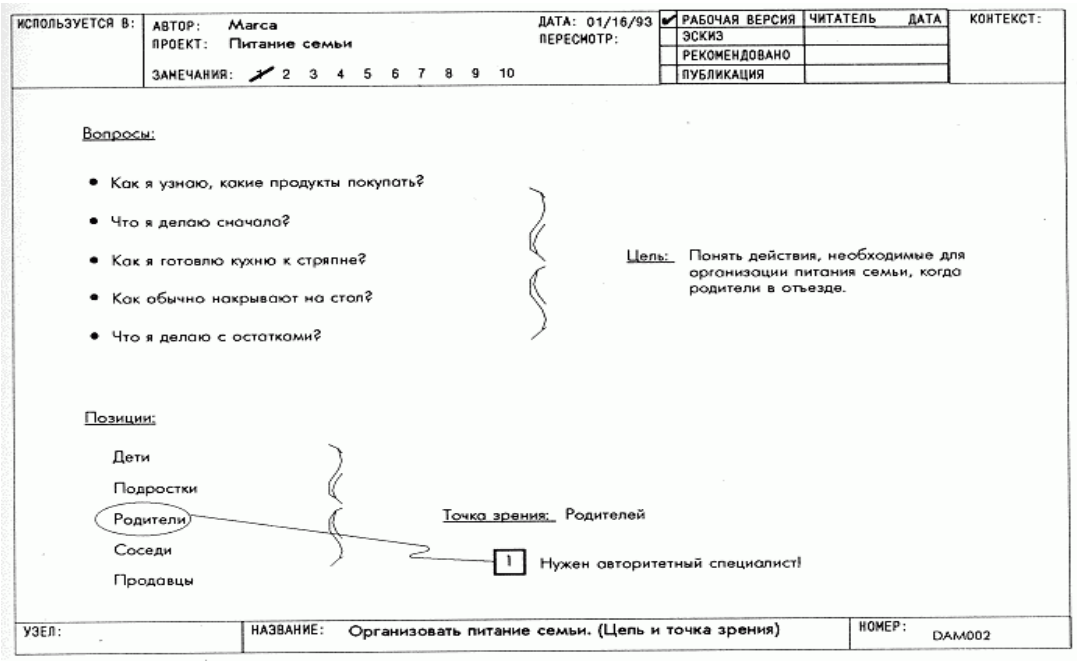

Рис. 2. Цель и точка зрения модели

3. Теперь нарисуйте и пометьте дуги, представляющие ограничения «извне» системы, используя составленный список данных. Подумайте, какого рода объекты влияют на целевую функцию.

4. Изобразите основной поток данных, прокладывая путь от блока к блоку. Используйте список данных и представьте себе, что вы рассказываете о функциональности модели.

Образец для модели «Питание семьи» (рис. 3)

Задание 4. Обобщение диаграммы верхнего уровня (диаграмма А–0).

Обобщение является последним важным шагом начального этапа моделирования. Диаграмма А–0 имеет несколько предназначений. Во-первых, она объявляет общую функцию всей системы. Во-вторых, она дает множество основных типов или наборов данных, которые использует или производит система. В-третьих, А–0-диаграмма указывает взаимоотношения между основными типами данных, проводя их разграничение. Таким образом, А–0-диаграмма представляет собой общий вид изучаемой системы. При создании диаграммы А–0 используется информация, уже зафиксированная на диаграмме АО.

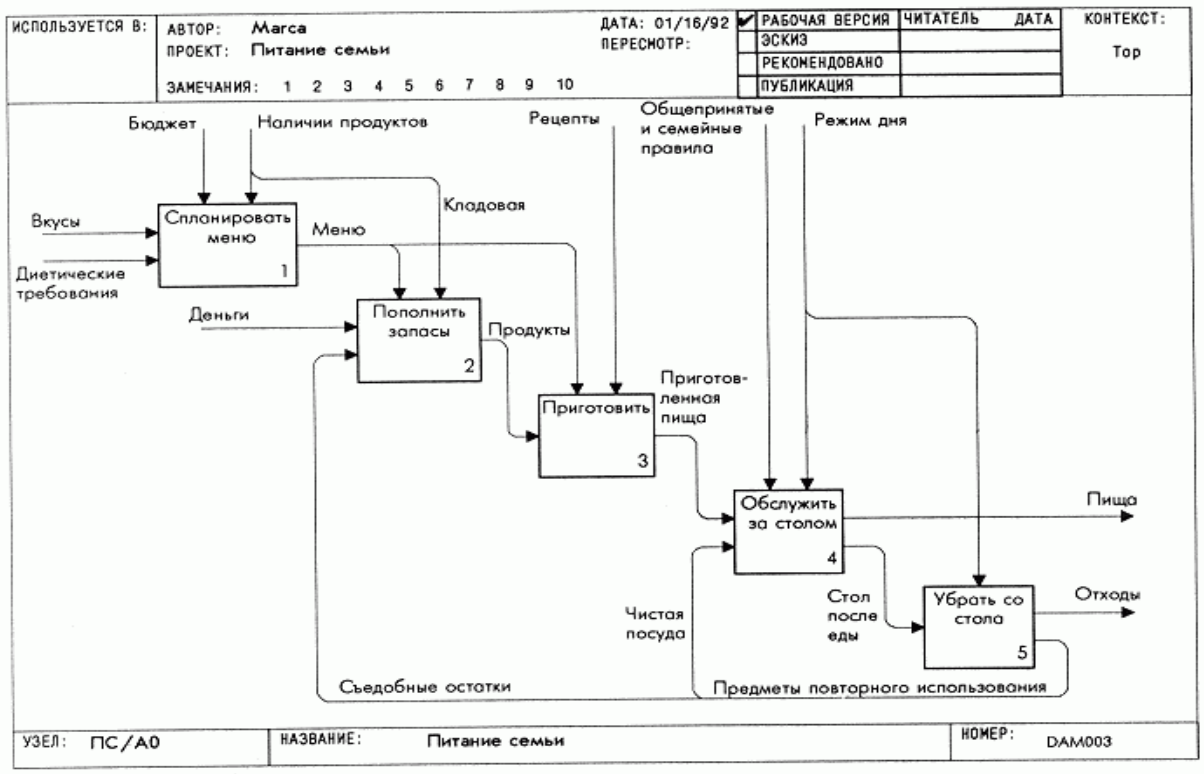

Рис. 3. Диаграмма верхнего уровня модели А0

Цель

Нарисовать единственный блок с его входами, управлениями и выходами, который обобщает всю только что нарисованную диаграмму А0.

Действия

1. Нарисуйте единственный большой блок в середине страницы и пометьте его названием диаграммы А0. Это обобщает все функции системы.

2. Теперь нарисуйте и пометьте все входные дуги, дуги управления и выходные дуги – по одной для каждой внешней дуги диаграммы А0. Это обеспечивает согласованность двух рисунков.

3. Наконец, напишите под большим блоком цель и точку зрения модели. Это определит смысл и направленность модели каждому, кто начнет ее читать.

Образец для модели «Питание семьи» (рис. 4)

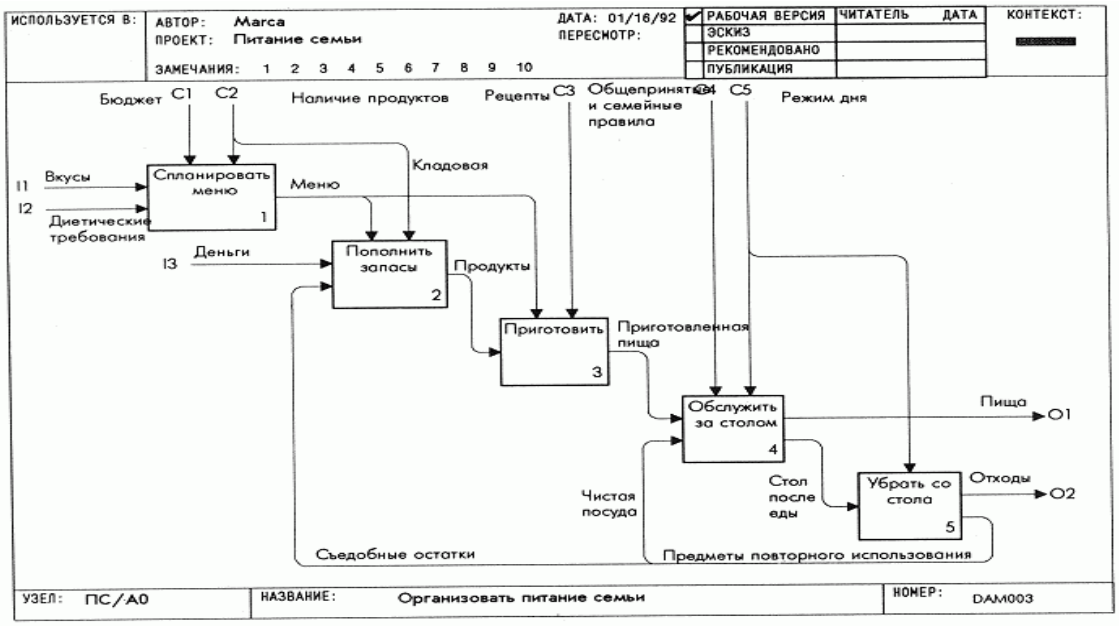

Рис. 4. Обобщение диаграммы верхнего уровня A–0

Требования к отчету по лабораторной работе

Отчет должен содержать:

1. Титульный лист.
2. Цель работы.
3. Общую характеристику функциональной модели.
4. Этапы разработки функциональной модели с приложением соответствующих диаграмм.
5. Выводы по работе.

Лабораторная работа № 2

Методология RAD (4 часа)

Цель работы: применить методологию RAD для разработки программного модуля, определенного в техническом задании на разработку магистерской диссертации.

Описание работы. Разработать программный модуль, определенный в техническом задании на разработку магистерской диссертации. Выделить 4 фазы разработки программного модуля, описать полученные результаты.

Теоретические сведения

RAD – это модель быстрой разработки приложений, ключевыми для которой является скорость и удобство программирования (рис. 5).

Рис. 5. Методология RAD

Первую версию RAD создал Барри Боэм в 1986 году, который назвал её «спиральная модель». Каждый виток спирали разбит на 4 сектора и соответствует разработке фрагмента или версии ПО. С каждым новым витком идёт углубление и уточнение целей, спецификаций проекта. В результате появляется возможность выбрать обоснованный вариант. Использовав идеи Барри, Джеймс Мартин разработал свою систему RAD на протяжении работы в 80-х в IBM и окончательно сформулировал их в книге «Быстрая разработка приложений» в 1991 году.

Спиральная модель RAD – это процесс разработки ПО, комбинирующий проектирование и постадийное прототипирование.

Жизненный цикл программного обеспечения по RAD

В процессе RAD приложение проходит четыре фазы.

1. Фаза анализа и планирования требований.

Определяются требования, функции приложения и их приоритетность, описываются информационные потребности. Фаза выполняется преимущественно пользователями при участии разработчи-

ков. На этой стадии также обозначаются масштаб проекта, временные и финансовые рамки, платформы для запуска ПО.

Например, компании «Beverly Flowers» нужно приложение для онлайн-заказа цветов на дом. На создание отводится 50 дней, бюджет –3000 долларов.

2. Фаза проектирования.

Часть пользователей участвует в техническом проектировании системы под руководством разработчиков. Группы или подгруппы RAD на этой фазе обычно используют комбинацию техник совместной разработки приложений (Joint Application Development, JAD) и CASE-инструменты для воплощения потребностей пользователей в рабочих моделях.

На фазе проектирования пользователи могут понять, модифицировать и определить рабочую модель системы, которая соответствует их нуждам. Каждый процесс рассматривается детально и, при необходимости, создаётся **частичный прототип**.

В результате фазы создаются:

- общая информационная модель приложения;
- функциональные модели системы и подсистем;
- рабочие прототипы экранов, отчётов и диалогов.

Если в предыдущих моделях средства разработки прототипов не соответствовали реальным приложениям и они в дальнейшем не использовались, то в RAD каждый прототип становится частью будущей системы.

Так, в приложении «Beverly Flowers» пользователи должны иметь доступ к возможностям: «Домашняя страница», «Поиск цветов», «Посмотреть список цветов». В качестве платформы разработки выбрали freeware SpringToolSuite, для которой доступно большое количество сэмплов – прописанных кусочков кода. В роли сервера – Apache Tomcat.

3. Фаза построения.

На этой стадии происходит непосредственно быстрая разработка на основе полученных по предыдущим фазам результатов. При этом пользователи продолжают участвовать в развитии системы, предлагая изменения и улучшения приложения. Тестирование приложения тоже происходит во время разработки.

Приложение «Beverly Flowers» собирается из трёх функциональных компонентов – перехода пользователя на «Домашнюю страницу», «Поиск цветов» и «Просмотреть список цветов». Для разработки рабочей модели понадобился 1 специалист и 8 дней.

4. Фаза внедрения.

Охватывает обучение пользователей, тестирование и замену старой системы на новую. Подготовка к этой фазе начинается с этапа проектирования.

Ранее компания «Beverly Flowers» принимала заказы непосредственно в точках сбыта и по телефону. Записав сообщение о возможности заказа через специальное приложение и разместив информационные стенды в точках продажи, за 2 недели удалось переключить часть покупателей на новый канал сбыта. При этом доля заказов по телефону пропорционально снизилась, а значит удалось сократить штат менеджеров по работе с клиентами.

Требования к отчету по лабораторной работе

Отчет должен содержать:

1. Титульный лист.
2. Цель работы.
3. Общую характеристику программного модуля.
4. Фазы разработки модуля и полученные результаты.
5. Выводы по работе.

Лабораторная работа № 3

Методология RUP (4 часа)

Цель работы: применить методологию RUP для разработки программного модуля, определенного в техническом задании на разработку магистерской диссертации.

Описание работы. Разработать программный модуль, определенный в техническом задании на разработку магистерской диссертации. Работа включает три задания.

Задание 1. Выделить стадии разработки проекта и результаты, полученные на каждой стадии. Определить сценарии использования (Use Case), архитектурные решения и компоненты, которые будут использоваться совместно для поддержки нескольких бизнес-сценариев.

Задание 2. Разработать статическую структуру проекта (выделить роли, действия и рабочие продукты).

Задание 3. Определить варианты использования разрабатываемого модуля и действующих лиц.

Подготовить отчет по выполненной работе.

Теоретические сведения

Методология RUP (Rational Unified Process – рациональный унифицированный процесс) предназначена для разработки больших программных систем. Цель – обеспечить производство качественного программного обеспечения (ПО), которое отвечает потребностям конечных пользователей, в рамках предсказуемого графика и бюджета.

Технология RUP – это итеративный процесс, состоящий из фаз проекта разработки ПО: начало, разработка, строительство и переход (рис. 6). Каждая фаза включает одну или несколько итераций, на которой создается исполняемый файл, но это может быть неполная система. На каждой итерации команда выполняет действия (рабочие процессы) с разной степенью детализации.

Базовые принципы методологии RUP

1. Разрабатывайте итеративно (Controlled Iterative). Прототипная спиральная разработка; главное – работающий правильно и эффективно программный код. Этот подход обеспечивает большую гибкость при изменяющихся требованиях и тактических коррективах в бизнес-целях, что позволяет более эффективно и заблаговременно идентифицировать и снижать проектные риски.

2. Управляйте требованиями (Use-case driven). Обеспечьте выполнение требований заказчиков: документирование и строгое исполнение требований, обеспечивать между всеми участниками проекта единое понимание ожидаемых функциональных возможностей,

3. Используйте компонентную архитектуру (Architecture Centric). Стройте систему из компонентов – применение повторно-используемых компонентов (объектно-ориентированный подход). Проектирование, реализация и тестирование архитектуры системы на ранних стадиях использования.

4. Моделируйте визуально (Visual Modeling Techniques). Моделирование по методологии RUP является объектно-ориентированным и базируется на понятиях объектов, классов и зависимостей между ними с помощью унифицированного языка моделирования **Unified Modelling Language™ (UML)**.

5. Непрерывно проверяйте качество (Continuos Quality Management). Сделайте качество образом жизни, а не запоздалой идеей – Управление качеством (тестирование на всех фазах, а не только в конце развертывания). Оценка качества всех работ, выполняемых любыми участниками проекта, использование объективных метрик и критериев. Методология RUP создавалась с прицелом на поддержку управления качеством в рамках требований SEI CMM/CMMI.

6. Управление изменениями с самого начала разработки и на всех фазах жизненного цикла (ЖЦ) – **Requirements Configuration and Change Management.** Приспособление к изменениям с самого

начала проекта. Закладывание основы используемой архитектуры как можно раньше.

7. Работайте вместе как одна команда – Командный принцип разработки (архитекторы, аналитики, программисты, тестировщики в одном проекте)

На рисунке 6 приведена модель жизненного цикла методологии RUP и распределение основных процессов по стадиям разработки.

Новация 1. Бизнес-моделирование и связанные с ним ***сценарии использования*** (Use Case). Будущая система представляется «чёрным ящиком», который удовлетворяет все потребности Пользователя/Бизнеса. На его основе разрабатываются системные сценарии использования, описывающие функции системы, которые будут поддерживать выполнение бизнес-сценария.

Новация 2. Принимаются архитектурные решения и выделяются компоненты, которые будут использоваться совместно для поддержки нескольких бизнес-сценариев.

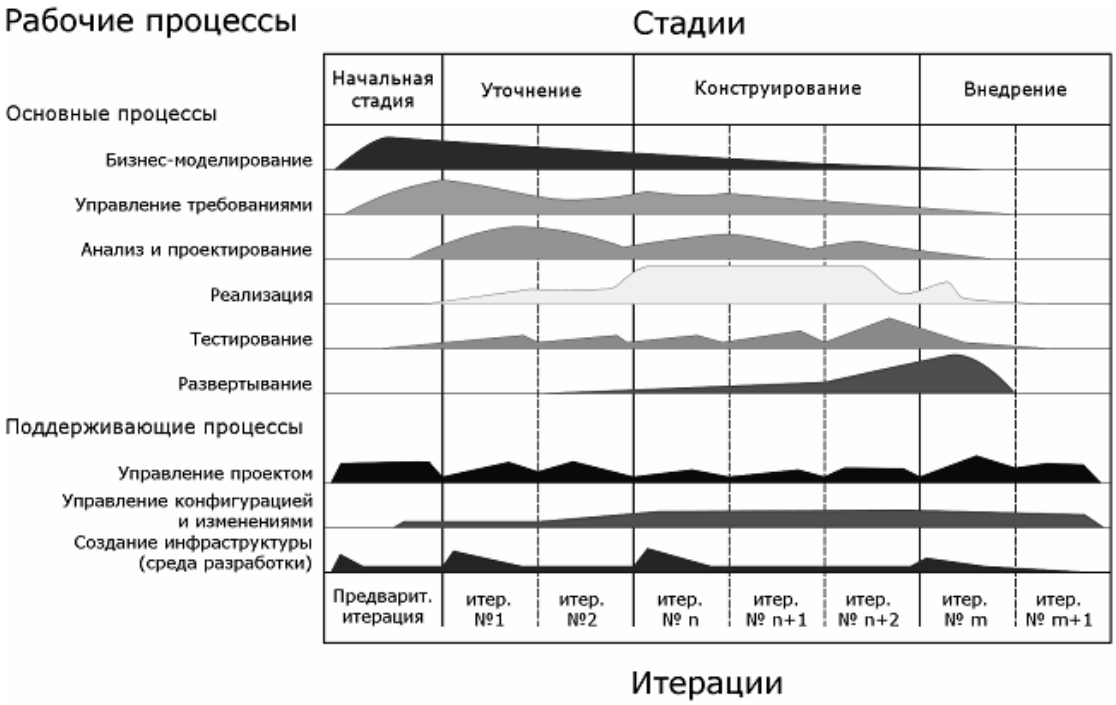

Рис. 6. Модель жизненного цикла RUP

Выделяются следующие новации RUP (см. рис. 7).

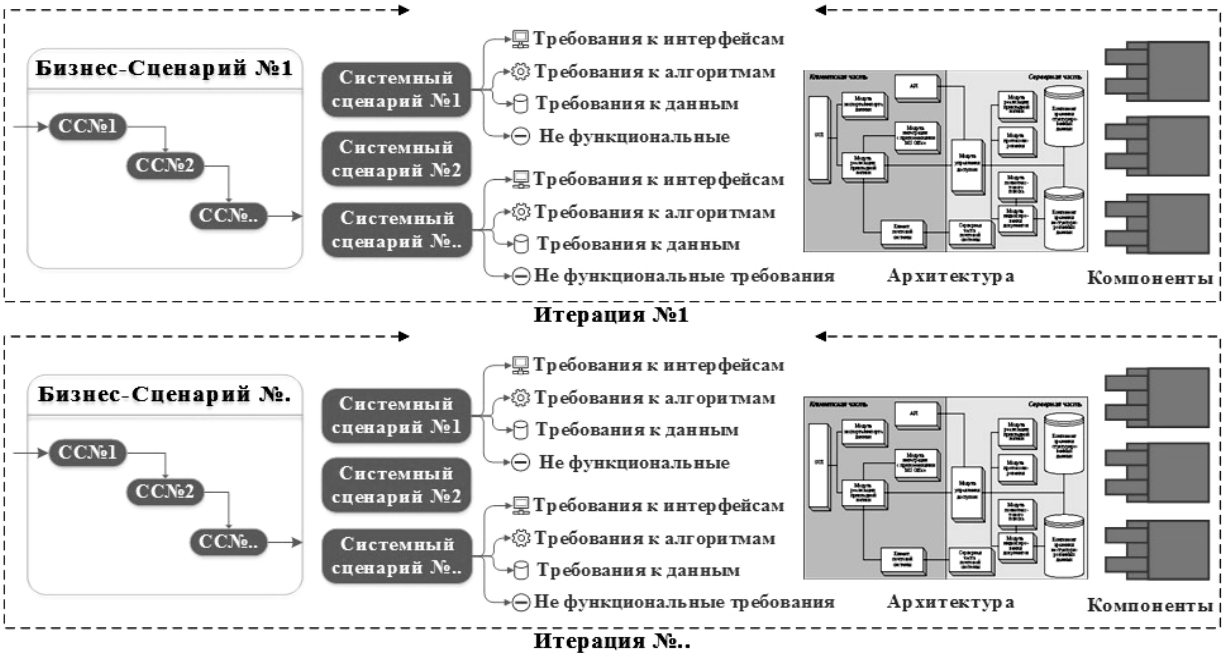

Рис. 7. Новации методологии RUP

Новация 3. Разделение бизнес-сценариев на ***первичные*** и ***вторичные***, которые основываются на результатах выполнения первичных сценариев. Отсюда появляется возможность выделить итерации и разложить реализацию всех сценариев по ним.

При разработке используются контрольные точки (вехи) проекта (пример, рис. 8).

Рис. 8. Контрольные точки (вехи) проекта

К нефункциональным требованиям относятся производительность, время доступности, скорость ответа и т. д. На основе выделенных требований определяется будущая архитектура системы и её функциональные модули (сервисные подсистемы). Подсистемы разбиваются на компоненты. Компоненты содержат исполняемый код.

Задание 1. Выделить стадии разработки проекта.

Итерационность технологии объектно-ориентированного проектирования RUP включает следующие стадии разработки проекта:

1. Начальная стадия (inception).

Цели:

- Понять границы проекта, установить ключевые высокоуровневые требования к системе;
- Разработать экономическое обоснование (business case), его концепцию (vision), список рисков;
- Добиться соглашения между заинтересованными сторонами для дальнейшего продвижения.

Веха целей жизненного цикла (LCO – Lifecycle Objective).

Результаты стадии:

- Общее описание системы (основные требования);
- Начальная диаграмма вариантов (прецедентов использования) – модель бизнес-процессов;
- Начальный проектный глоссарий;
- Начальный бизнес-план;
- План проекта, отражающий стадии и итерации;
- Один прототип или несколько.

2. Стадия уточнения (проектирования) – Elaboration.

Цели:

- Свести к минимуму технические риски;
- Создать базовую архитектуру (спроектировать, реализовать и протестировать ключевые компоненты, интерфейсы и архитектурные механизмы) функциональности;

- Понять затраты построения системы.

Веха архитектуры жизненного цикла (LCA – Lifecycle Architecture).

Результаты:

- Детальные требования к ПО в виде диаграммы вариантов использования – 80 %;
- Перечень нефункциональных требований;
- Описание **базовой архитектуры**:

a.Модель предметной области (классов объектов);

b. Технологическая платформа, определяющая основные элементы технологии реализации и их взаимодействия;

- Работающий прототип (20 % вариантов использования);
- Уточненный бизнес-план (затраты, риски);
- План разработки всего проекта, отражающий итерации и критерии для каждой итерации (детально следующая итерация).

3. Стадия конструирования (реализации) – Construction.

Цели: Построить первую работающую версию продукта (несколько внутренних и альфа-релизов, выпуск бета-версии со средствами инсталляции, справочной документации, учебного материала).

Веха начальной функциональной готовности (IOC – Initial Operational Capability).

Результаты:

- ПО, интегрированное на требуемых платформах (бета-версия);
- Руководства пользователей;
- Описание текущей реализации;
- План развертывания.

4. Стадия внедрения (ввода в действие) – Transition.

Цели: Создать окончательную версию продукта и отправить заказчику (тестирование продукта, настройка на эксплуатацию).

Веха готового продукта (PR – Product Release).

Результаты:

- Бета тестирование;
- Параллельное функционирование с существующей системой;
- Конвертирование баз данных;
- Оптимизация производительности;
- Обучение пользователей и специалистов IT-служб.

Задание 2. Разработать статическую структуру проекта (выделить роли, действия и рабочие продукты). Пример статической структуры проекта на рисунке 9.

Задание 3. Определить варианты использования разрабатываемого продукта и действующих лиц.

Приведем типовую схему действий и шагов по созданию вариантов использования.

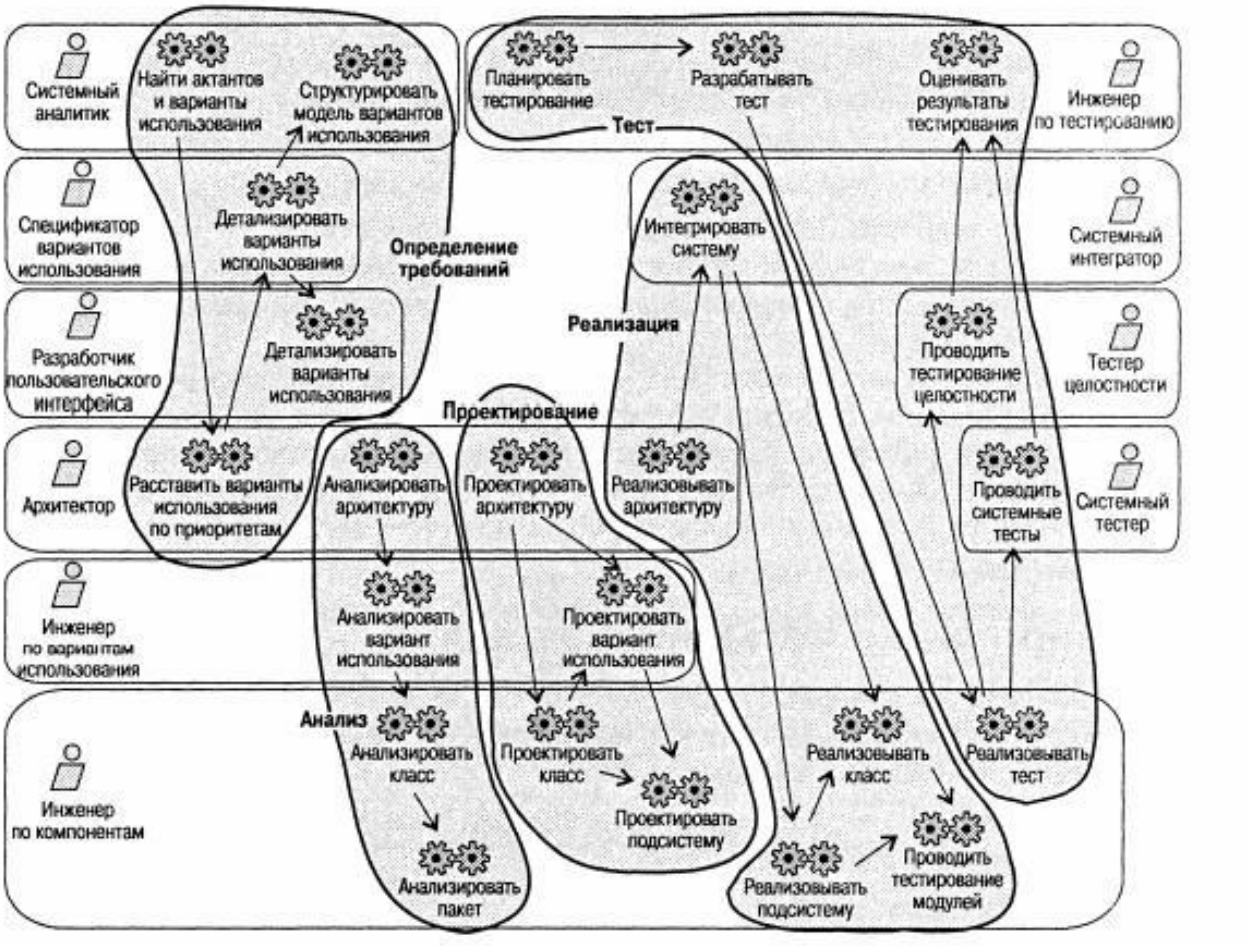

Рис. 9. Статическая структура проекта

Действие: Определение вариантов использования и действующих лиц.

Шаг 1: Определение действующих лиц.

Шаг 2: Определение вариантов использования.

Шаг 3: Описание взаимодействия вариантов использования и действующих лиц.

Шаг 4: Объединение вариантов использования и действующих лиц в пакеты.

Шаг 5: Построение диаграммы вариантов использования.

Шаг 6: Обзор модели вариантов использования.

Шаг 7: Оценка полученных результатов.

Требования к отчету по лабораторной работе

Отчет должен содержать:

1. Титульный лист.

2. Цель работы.

3. Общую характеристику разрабатываемого программного модуля (цели, этапы, задачи и вехи).

4. Сценарии использования, архитектурные решения и компоненты, используемые для поддержки разработанных бизнес-сценариев.

5. Описание статической структуры проекта.

6. Выводы по работе.

Лабораторная работа № 4

Диаграммы потоков данных DFD (2 часа)

Лабораторная работа содержит необходимую информацию о диаграммах DFD, включая основные определения, историю возникновения, а также символы и способы нотации. Требуется изучить разные уровни диаграмм DFD, различать их логические и физические варианты. Для построения диаграмм удобно использовать бесплатную версию редактора диаграмм Lucidchart или аналогичные программные продукты.

Диаграмма DFD наглядно отображает течение информации в пределах процесса или системы. Для изображения входных и выходных данных, точек хранения информации и путей ее передвижения между источниками и пунктами доставки в таких диаграммах применяются стандартные фигуры, такие как прямоугольники и круги, а также стрелки и краткие текстовые метки. Диаграммы DFD варьируются от простейших набросков процессов (включая нарисованные вручную) до подробных многоуровневых схем с глубоким анализом способов обработки данных.

Самые распространенные системы нотации DFD-схем названы в честь их создателей:

- Йордон и Коуд;
- Йордон и Де Марко;
- Гейн и Сарсон.

Основное различие между этими системами заключается в том, что методы Йордона-Коуда и Йордона-Де Марко для обозначения процессов применяют круги, а метод Гейна-Сарсона – прямоугольные блоки со скругленными углами (которые иногда называют «конфетками»). Подчиняясь правилам и инструкциям выбранной системы, символы отображают четыре компонента диаграммы DFD:

1. Внешние сущности.

Внешние системы, из которых поступает или куда направляется информация в результате взаимодействия с изображаемой системой. Иными словами, это источники и пункты доставки информации, которая приходит или уходит из системы. Такими сущностями могут быть внешние организации, лица, компьютерные или бизнес-системы.

2. Процессы.

Любые процессы, которые ведут к изменению информации и созданию выходных данных. Например, выполнение подсчетов, сортировка данных согласно установленной логике или направление информационного потока в соответствии с бизнес-правилами. Для опи-

сания процессов используются краткие метки, например, «Отправка платежа».

3. Хранилища данных.

Файлы или репозитории, где хранится информация для последующего использования, например, базы данных или формы заявки на участие. Хранилища данных сопровождаются простыми метками, например, «Заказы».

4. Потоки данных.

Маршруты, по которым информация перемещается между внешними сущностями, процессами и хранилищами данных. Потоки данных иллюстрируют взаимодействие между другими компонентами и отображаются в виде стрелок, как правило, с краткими метками, например, «расчетная информация».

Правила по построению диаграмм DFD

- Каждый процесс должен сопровождаться как минимум одним входным и одним выходным потоком;
- В каждое хранилище должен впадать как минимум один поток данных и как минимум один – вытекать;
- Данные, хранимые в системе, должны проходить через процесс;
- Каждый процесс диаграммы DFD должен вести либо к другому процессу, либо к хранилищу данных.

Уровни и слои DFD-схем: от контекстных схем до псевдокода

С помощью слоев и уровней диаграмму DFD можно дополнять всё большими и большими подробностями, фокусируя внимание на одном конкретном участке. Уровни диаграммы обозначаются цифрами 0, 1 или 2, причем иногда нумерация может продолжаться (3 и так далее). Необходимый уровень детализации зависит от стоящих перед разработчиками целей.

DFD нулевого уровня также называется **контекстной схемой** (см. рис. 10). Это простейший способ изображения анализируемых или моделируемых систем и процессов. Такие схемы показывают

общую картину и представляют систему в виде единого процесса, наделенного связями с внешними сущностями. Схемы нулевого уровня будут понятны широкой аудитории, включая участников проекта, бизнес-аналитиков и разработчиков.

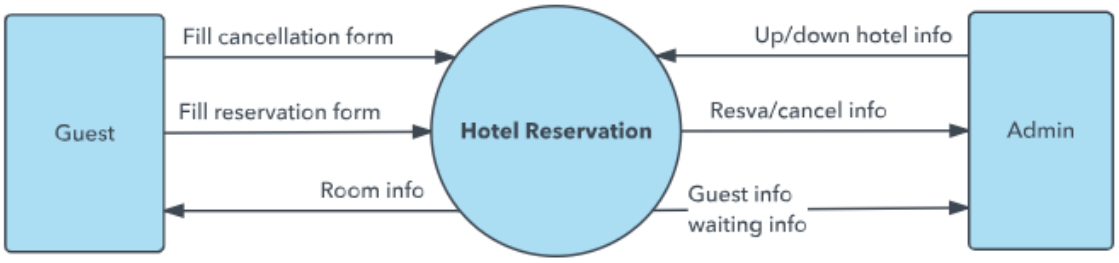

Рис. 10. Пример диаграммы DFD нулевого уровня

DFD первого уровня дает более детальное представление об элементах контекстной схемы (см. рис. 11). Разбив обобщенный процесс контекстной схемы на подпроцессы, тем самым выделяются основные функции системы.

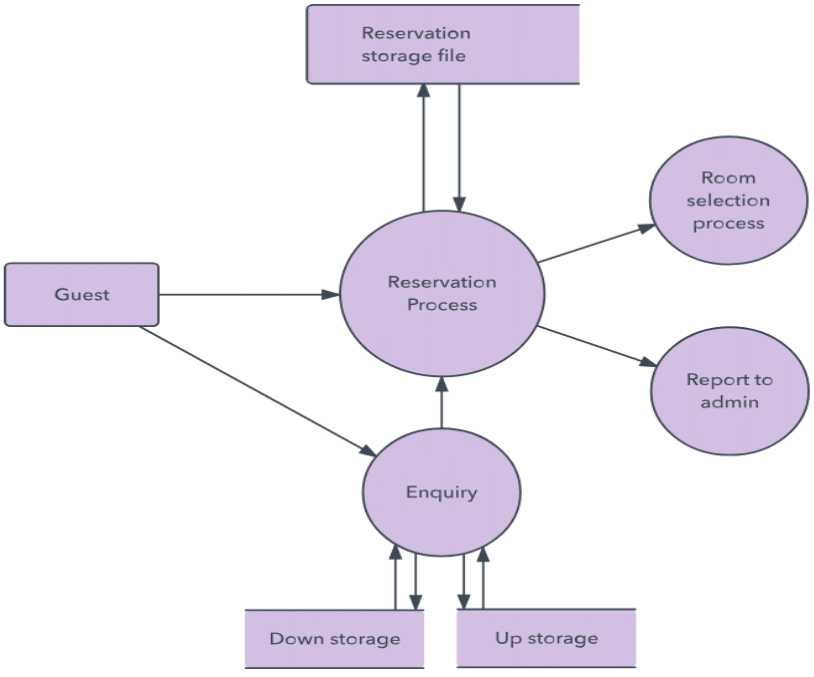

Рис. 11. Пример диаграммы DFD первого уровня

DFD второго уровня обеспечивает еще более глубокое погружение в систему (см. рис. 12). Чтобы достаточно подробно описать ее устройство, требуется включить в схему немного больше текста.

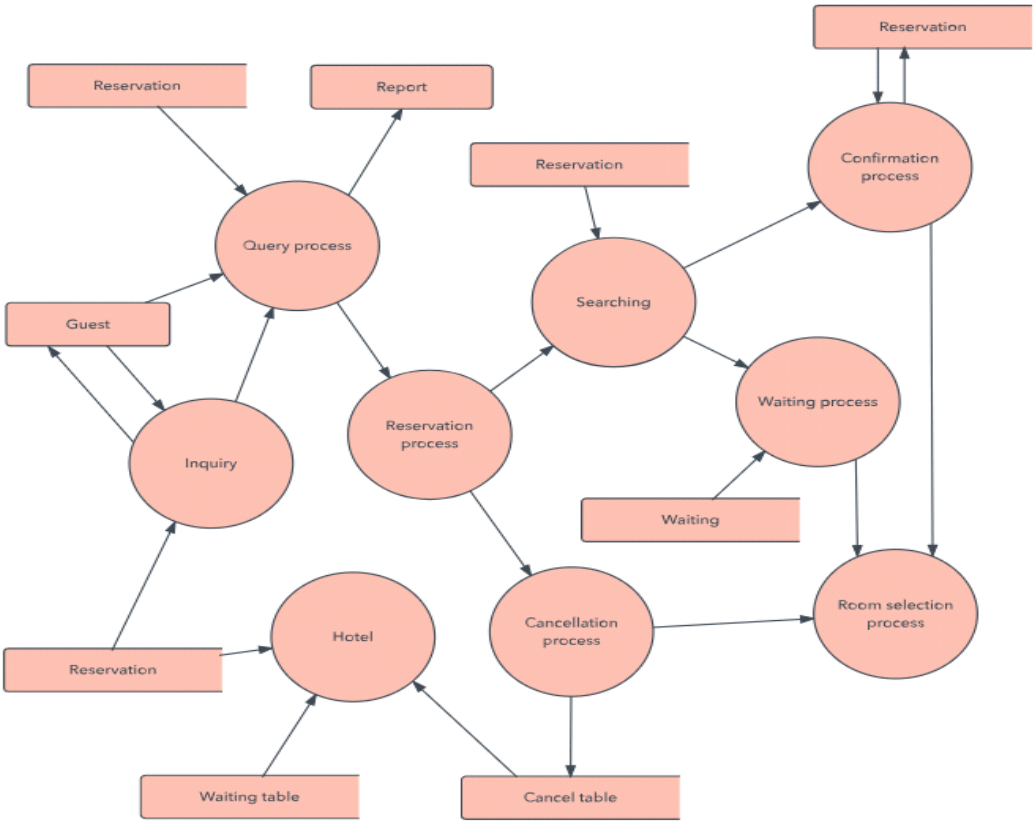

Рис. 12. Пример диаграммы DFD второго уровня

Слои позволяют собрать выпадающие уровни непосредственно в DFD-схеме: указанный метод сочетает глубокий анализ с ясностью изложения.

При достаточно высокой детализации разработчики и дизайнеры могут применить диаграммы DFD для написания псевдокода, который представляет собой сочетание программного и естественного языка. Задача псевдокода – упростить работу по написанию полноценного кода.

Как создать диаграмму DFD

1. ***Ввод и вывод информации.***

Большинство процессов и систем начинаются с ввода информации из внешнего источника и заканчиваются выводом данных в другую сущность или базу. Выявив ключевые пункты ввода и вывода информации, вы получите общую картину своей системы и ее основных задач. Важно определиться с вводом и выводом информации на

раннем этапе работы, так как это тот самый фундамент, на котором будет выстраиваться оставшаяся часть вашей диаграммы DFD.

2. ***Создание контекстной схемы.***

После того как вы определились с основными пунктами ввода и вывода информации, дальнейшее построение контекстной схемы не составит труда. Добавьте узел единичного процесса и соедините его с необходимыми внешними сущностями. Этот узел символизирует самый обобщенный процесс, через который проходит информация от ввода до вывода.

3. ***Расширение контекстной схемы до DFD первого уровня.***

Узел единичного процесса, представленный в вашей контекстной диаграмме, дает довольно бедное представление о системе, поэтому рекомендуем разбить его на подпроцессы. Диаграмма DFD первого уровня должна включать несколько узлов процессов, а также основные базы данных и все внешние сущности. Отследите путь информационного потока: откуда поступает информация и что с ней должно случиться до перехода к очередному хранилищу?

4. ***Углубление диаграммы DFD до второго уровня и далее.***

Если вы хотите создать более подробную схему потока данных, следуйте инструкции из пункта 3. Процессы, входящие в состав диаграммы DFD первого уровня, можно точно так же разбить на более подробные подпроцессы. Не забудьте включить в диаграмму все необходимые хранилища и потоки данных. На этом этапе вы должны получить довольно подробную картину своей системы.

5. ***Проверка окончательной схемы.***

Когда схема будет готова, проверьте ее от начала до конца. Обращайте особое внимание на течение информации: логично ли всё изложено? Все ли хранилища данных на месте? Финальный вариант схемы должен давать четкое представление о том, как устроена ваша система.

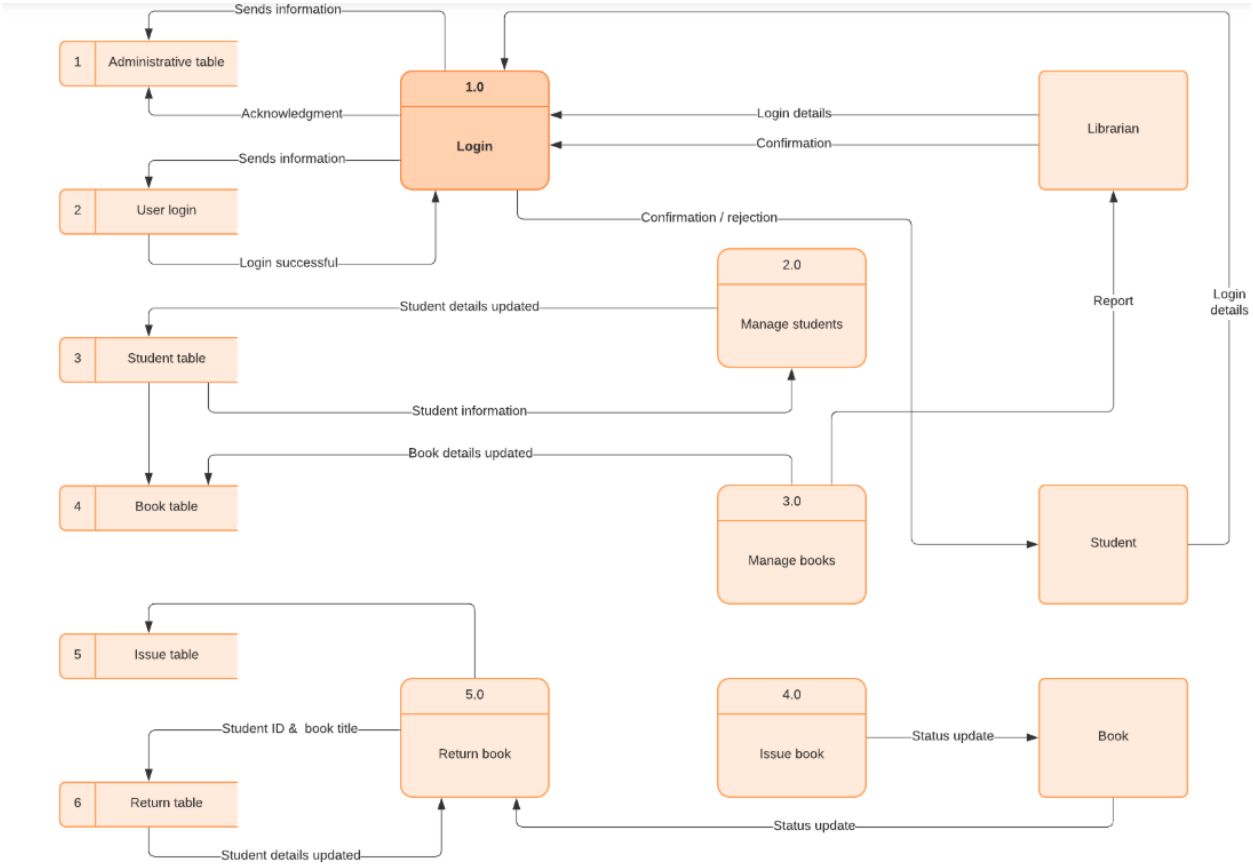

Рис. 13. Пример финальной DFD диаграммы для студенческой библиотеки

Требования к отчету по лабораторной работе

Отчет должен содержать:

1. Титульный лист.
2. Цель работы.
3. Общую характеристику варианта задания.
4. Диаграммы DFD.
5. Выводы по работе.

Лабораторная работа № 5

Методология объектного проектирования на языке UML (UML-диаграммы) (2 часа)

Задания выполняются с применением программы StarUML (возможно использование аналогичных программ). Необходимо объяс-

нить преподавателю ход работы и ответить на вопросы. Варианты – в конце описания работы.

Теоретические сведения

Канонические диаграммы UML (рис. 14)

Разработка диаграмм осуществляется в соответствии с синтаксисом языка UML. Для изучения синтаксиса диаграмм используйте литературу, приведенную в конце пособия. Ниже приведены примеры канонических диаграмм.

Рис. 14. Диаграммы UML

1. Пример диаграммы прецедента (рис. 15).

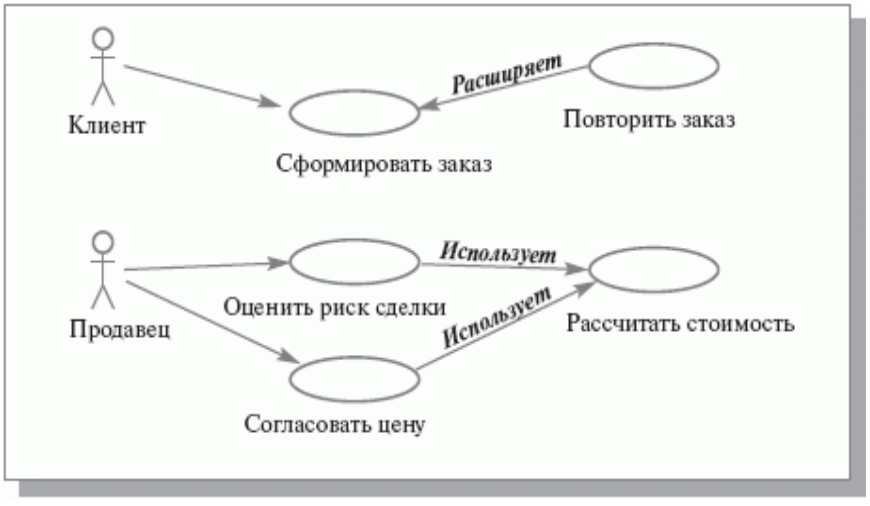

Рис. 15. Пример диаграммы прецедента

2. Пример диаграммы классов (рис. 16).

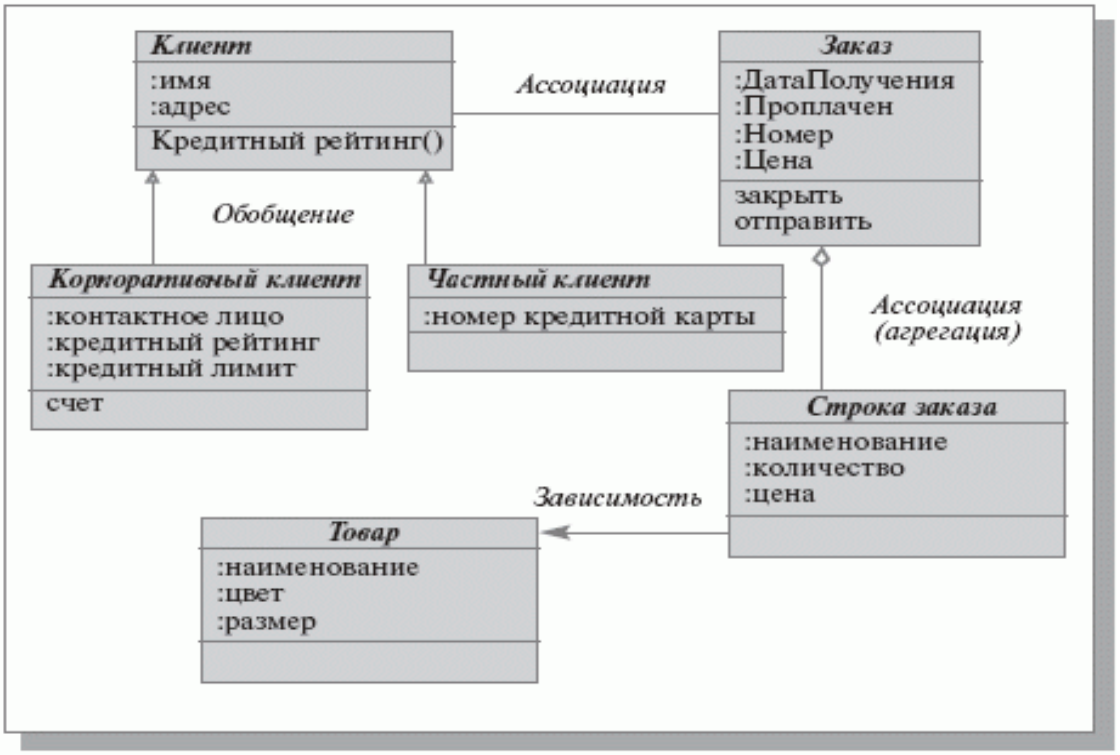

Рис. 16. Пример диаграммы классов

3. Пример диаграммы последовательности (рис. 17).

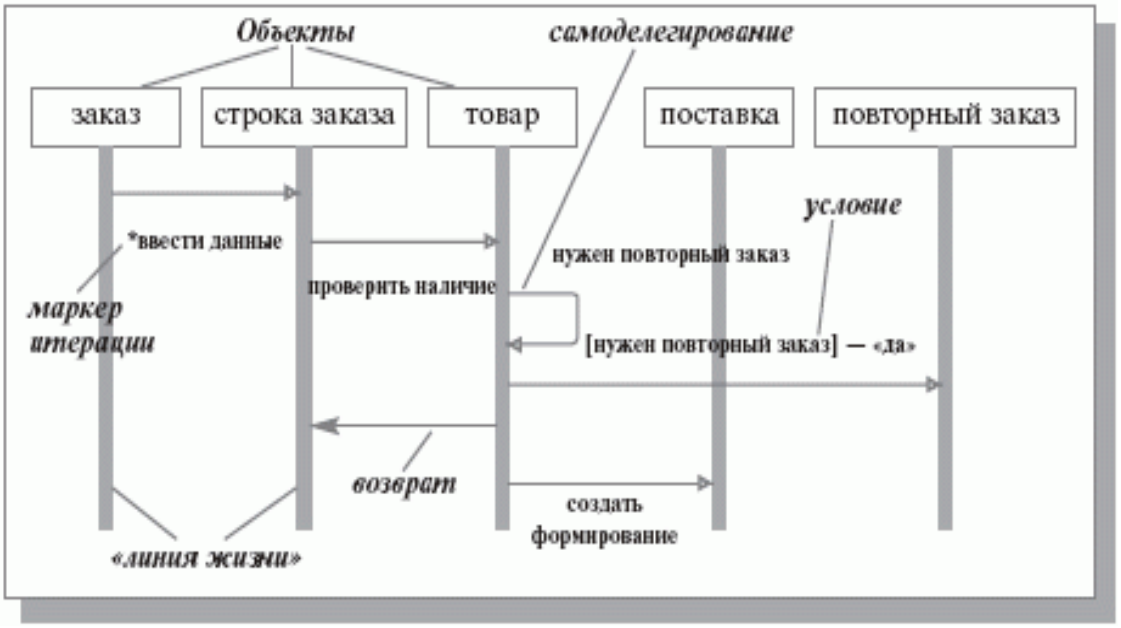

Рис. 17. Пример диаграммы последовательностей

4. Пример диаграммы состояний (рис. 18).

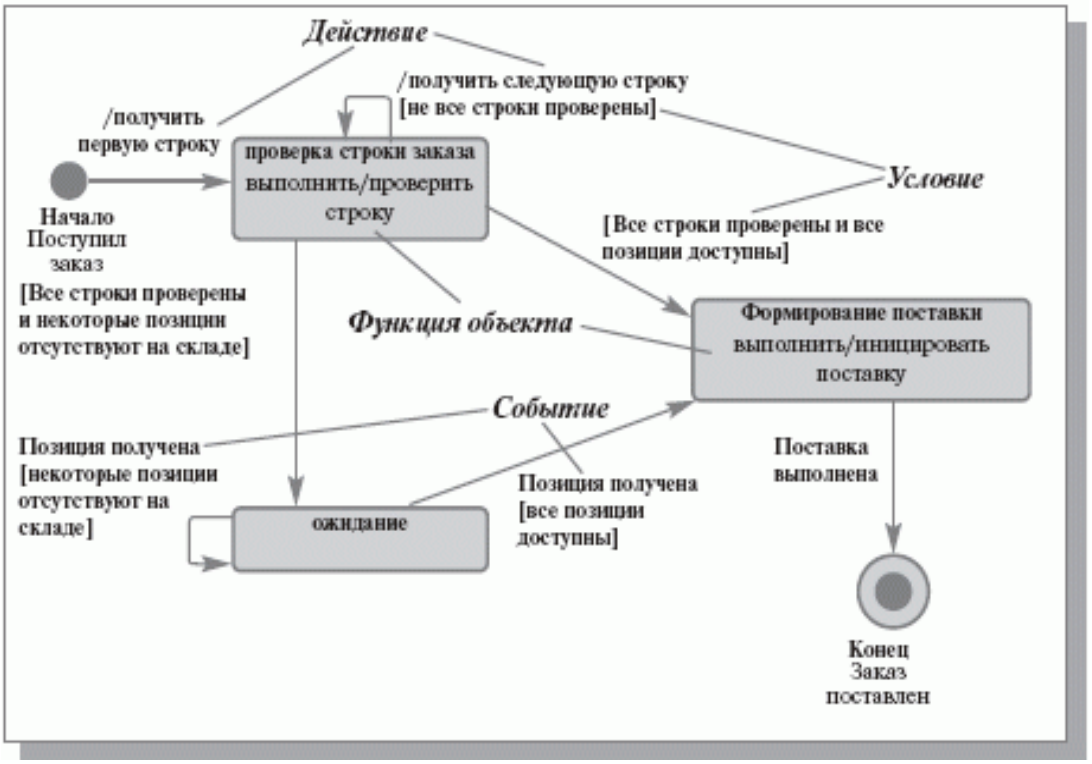

Рис. 18. Пример диаграммы состояний

5. Пример диаграммы деятельности (рис. 19).

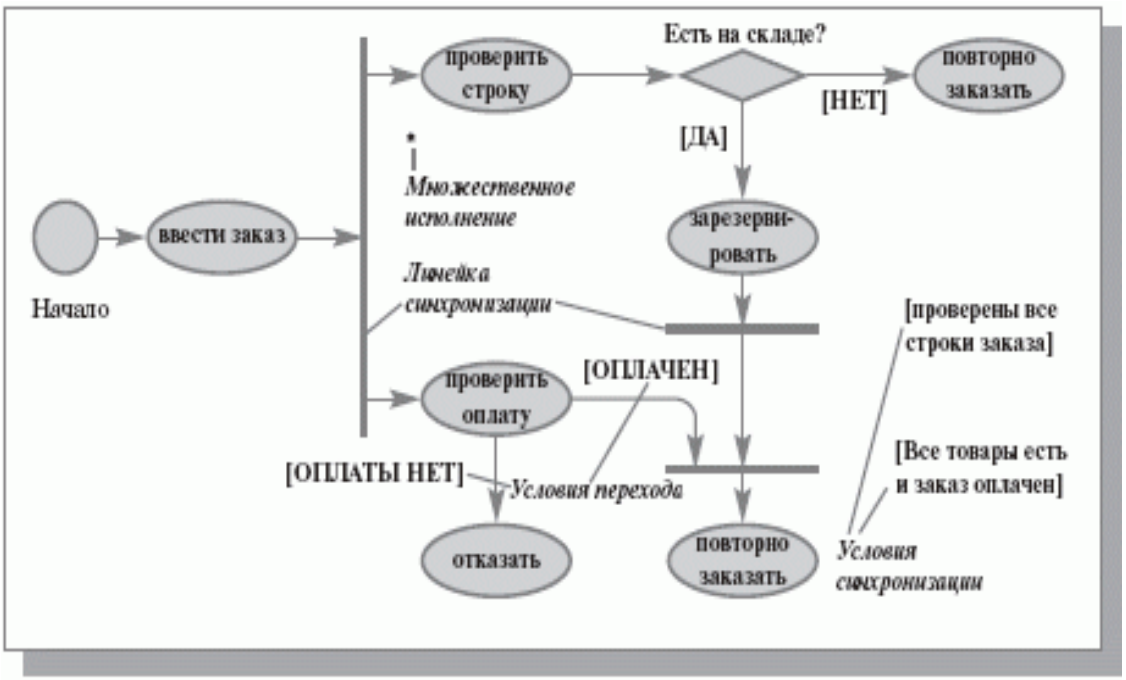

Рис. 19. Пример диаграммы деятельности

Лабораторная работа включает выполнение 4-х заданий, приведенных ниже и оформление отчета по требованиям.

Задание 1.

1) Определить рамки модуля (информационной системы). Построить контекстную диаграмму (пример – рис. 20), уточнить функциональные требования к модулю (информационной системе). Выписать их отдельным списком.

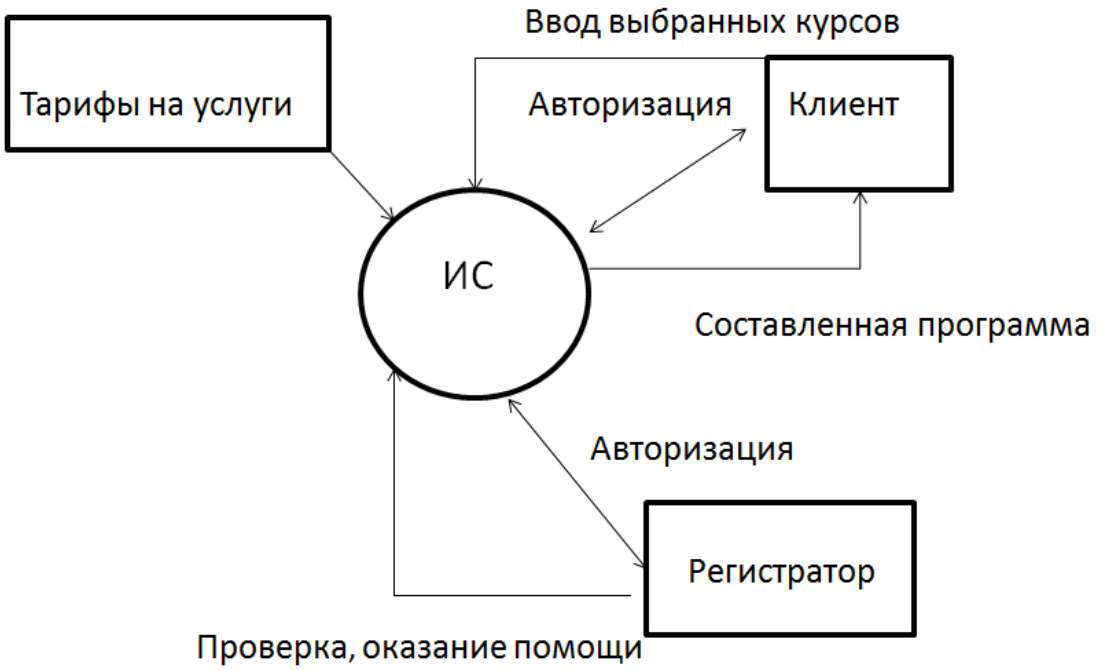

Рис. 20. Пример контекстной диаграммы ИС «Запись на курсы»

2) Выявить субъекты и прецеденты, построить диаграмму прецедентов. Составить документ описания для одного прецедента по примерному образцу, приведенному ниже.

Прецедент «Вычисление площади, периметра, длины выбранного объекта» (из технического проекта территориальной ИС).

Краткое описание

Пользователь осуществляет выбор пространственного объекта на карте, запускает инструмент расчета характеристик выбранного пространственного объекта и получает результат расчета.

Основной поток событий

1. Пользователь в окне карты выделяет географические объекты.

2. Система запускает функцию выделения географических объектов на карте активного векторного слоя.

3. Пользователь запускает операцию расчета характеристик выделенных географических объектов.

4. Система производит вычисление размеров географического объекта.

5. Система выводит результаты расчета в окно карты. Выполнение прецедента завершается.

Альтернативные потоки

Не выделены географические объекты. Выполнение прецедента завершается.

Предусловия

Пользователь должен быть аутентифицирован и авторизован в системе.

Постусловия

В результате выполнения основного потока событий система производит расчет размеров выделенных географических объектов и выводит результат расчета в текущем окне карты.

Специальные требования

Специальные требования отсутствуют.

Дополнительная информация

Дополнительная информация отсутствует.

Задание 2. Выявить классы, их атрибуты и операции. Определить виды отношений. Построить диаграмму классов.

Задание 3. Построить диаграмму деятельности (для одного прецедента). Построить диаграмму последовательностей (для одного прецедента).

Задание 4. Построить диаграмму состояний для одного объекта (класса), который может находиться более чем в двух состояниях. Построить диаграммы компонентов и развертывания.

Лабораторная работа № 6

Архитектура информационных систем

Цель работы: разработать концептуальную схему архитектурного описания системы в соответствие со стандартом ГОСТ Р 57100-2016/ISO/IEC/IEEE 42010:2011.

Варианты заданий на разработку даются преподавателем.

Архитектура системы – это принципиальная организация системы, воплощенная в её элементах, их взаимоотношениях друг с другом и со средой, а также принципы, направляющие её проектирование и эволюцию.

Для подробного описания принципов построения архитектуры используется стандарт ГОСТ Р 57100-2016/ISO/IEC/IEEE 42010:2011 «Системная и программная инженерия. Описание архитектуры» (Systems and software engineering. Architecture description).

Свод знаний по системной инженерии (Systems Engineering Body of Knowledge, SEBoK) делит архитектуру на две части: логическую и физическую.

Логическая архитектура поддерживает функционирование системы на протяжении всего её жизненного цикла на логическом уровне. Она состоит из набора связанных технических концепций и принципов. Логическая архитектура представляется с помощью методов, соответствующих тематическим группам описаний, и как минимум, включает в себя функциональную архитектуру, поведенческую архитектуру и временную архитектуру.

Функциональная архитектура представляет собой набор функций и их подфункций, определяющих преобразования, осуществляемые системой при выполнении своего назначения.

Поведенческая архитектура – соглашение о функциях и их подфункциях, а также интерфейсах (входы и выходы), которые определяют последовательность выполнения, условия для управления или потока данных, уровень производительности, необходимый для удо-

влетворения системных требований. Поведенческая архитектура может быть описана как совокупность взаимосвязанных сценариев, функций и/или эксплуатационных режимов.

Временная архитектура является классификацией функций системы, которая получена в соответствии с уровнем частоты её исполнения. Временная архитектура включает в себя определение синхронных и асинхронных аспектов функций.

Цель проектирования физической архитектуры заключается в создании физического, конкретного решения, которое согласовано с логической архитектурой и удовлетворяет установленным системным требованиям.

Архитектура может быть зафиксирована с помощью полного архитектурного описания (АО) (рис. 21). Стандарт ISO/IEC/IEEE 42010-2011 предписывает различать концептуальную архитектуру системы и одно из описаний данной архитектуры, являющееся конкретным продуктом или артефактом.

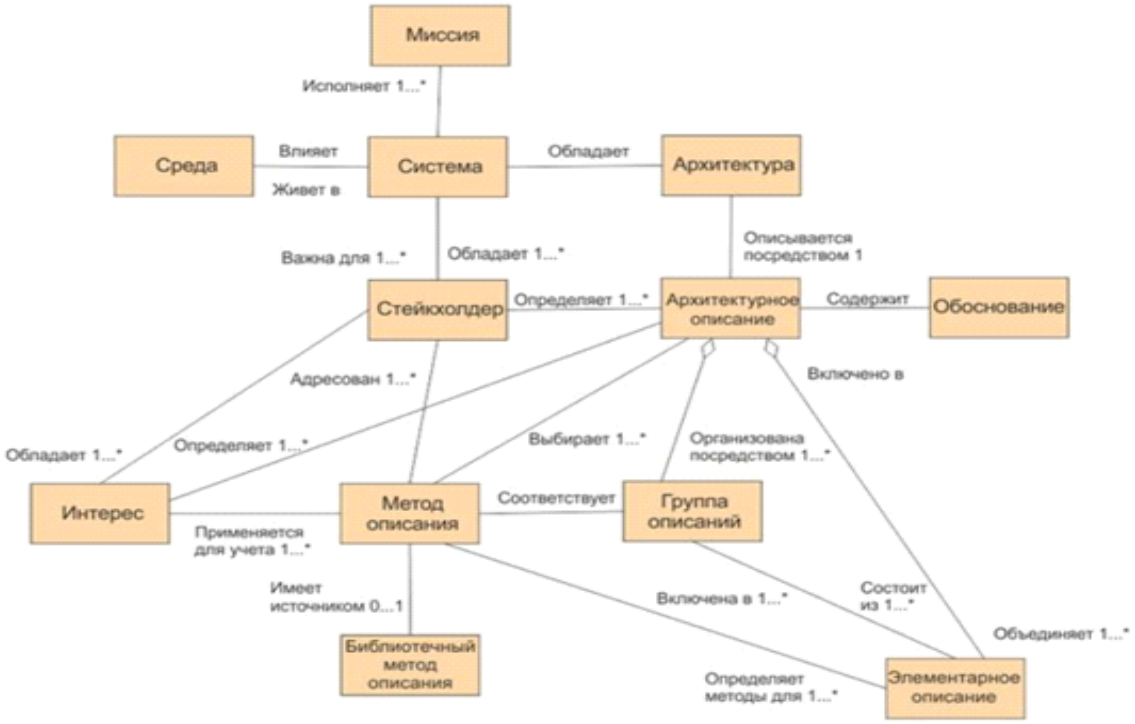

Рис. 21. Концептуальная схема архитектурного описания (ISO/IEC 42010)

Концептуальный подход определяет термины и понятия, относящиеся к содержанию и применению архитектурного описания. На рисунке изображены основные понятия и их взаимосвязи. Все понятия определены в контексте архитектуры определенной системы и соответствующего архитектурного описания.

На рисунке прямоугольники изображают классы сущностей. Линии, соединяющие прямоугольники, изображают связи между сущностями. Связь включает две роли (по одной в каждом направлении). Каждая роль может по желанию быть именована меткой. Роль, направленная от A к B, [помечена] ближе к B, и наоборот. Например, роли между «системой» и «средой» могут читаться: «система живёт в среде» и «среда влияет на систему». На рисунке роли обладают арностью 1:1, если не указано иное. Роль может обладать множественной арностью, например, роль, обозначенная как «1..*», применяется для обозначения многих, как в связях «один ко многим» или «многие к одному». Ромб (на конце линии связи) обозначает отношение части целого. Например, «группы описаний» являются частью «архитектурного описания». Эта нотация заимствована из UML.

Рассмотрим каждое составляющее концептуальной схемы подробнее. В контексте рассматриваемой схемы система распространяется на отдельные прикладные программные средства, системы в традиционном смысле, подсистемы, системы систем, продукты, семейства продукции, организации в целом и другие интересующие совокупности.

Система обитает в некоторой среде. Среда некоторой системы может влиять на данную систему. Её среда, или контекст, определяет обстановку и обстоятельства разработки, эксплуатации, политических и иных влияний на данную систему. Такая среда может включать другие системы, взаимодействующие с целевой системой, как напрямую через интерфейсы, так и косвенно иными путями. Такая среда определяет границы, определяющие предмет целевой системы по отношению к другим системам.

У каждой системы есть один или более стейкхолдеров (заинтересованных лиц). Каждый стейкхолдер обычно принимает участие в системе, или имеет интересы к данной системе. Интересы предполагают учёт таких аспектов системы как производительность, надежность, безопасность, распределённость и способность к эволюции. Любая система существует для реализации в своей среде одной или более миссий.

В концептуальном подходе архитектурное описание организовано как одна или более архитектурных групп описаний.

Требования к отчету по лабораторной работе

Отчет должен содержать:

1. Титульный лист.
2. Цель работы.
3. Общую характеристику программной системы.
4. Концептуальную схему архитектурного описания системы в соответствие со стандартом ГОСТ Р 57100-2016/ISO/IEC/IEEE 42010:2011.
5. Выводы по работе.

Варианты заданий для лабораторных работ

Вариант 1. Модуль «Школьная библиотека».

Примерное описание: Читатели (ученики и работники школы) получают книги в библиотеке для временного пользования. В каталоге книги распределены по разделам: учебная – по разным классам, методическая – по разным предметам, художественная литература – по классам и общая. На каждого читателя оформляется электронный формуляр. Необходимо вести учет книг библиотеки, читателей, список выданных книг, предусмотреть возможность напоминания через электронную почту о дате возврата книги, возможность пополнения каталога, оформления отчетов, печати нужных документов.

Вариант 2. Модуль «Прием в университет».

Примерное описание: При поступлении в университет абитуриенты подают заявление в электронной форме, где указываются персональные данные (фамилия, имя, отчество, адрес, дата рождения и т.д.), данные об обучении в учебных заведениях, номер аттестата или диплома. Прием в университет идет по результатам ЕГЭ. На творческие специальности возможен дополнительный экзамен. Необходимо вести учет абитуриентов, предусмотреть возможность отправки запроса в федеральную базу для получения результатов ЕГЭ, составление списков абитуриентов по сумме баллов, возможность отправки электронных писем абитуриентам с напоминаниями о предоставлении оригиналов, печать отчетов и требуемых документов.

Вариант 3. Модуль «Подготовительные курсы в университете».

Примерное описание: При университете организованы подготовительные курсы. Абитуриенты выбирают нужные дисциплины из предложенного списка, производят оплату. С абитуриентом составляется договор. При оплате курса, состоящего из нескольких дисциплин, предусмотрены скидки. Дисциплины закрепляются за преподавателями. Необходимо вести учет абитуриентов, составленных договоров, поступивших оплат, списков групп, закрепленных за преподавателем дисциплин, предусмотреть возможность предупреждения студентов по электронной почте при переносе или отмене занятий, возможность оформления и печати отчетов.

Вариант 4. Интернет-магазин по продаже книг.

Примерное описание: Клиент выбирает книгу на Web-странице магазина. Он оформляет заказ. Система должна проверить наличие выбранных книг и оформить счет к оплате. Постоянным клиентам и/или при крупном заказе предусмотрены скидки. После проведения оплаты система должна проверить поступление оплаты и оформить необходимые документы для доставки книги. Необходимо вести учет клиентов, книг, поступивших оплат, оформленных и выполненных заказов.

Вариант 5. Интернет-магазин по продаже школьных обучающих программ.

Примерное описание: Клиент выбирает обучающую программу на Web-странице магазина. Он оформляет заказ, указывая, нужен ли ему диск, или он желает получить программу через Интернет. Система должна оформить счет к оплате. Постоянным клиентам и/или при крупном заказе предусмотрены скидки. После проведения оплаты система должна проверить поступление оплаты и оформить необходимые документы для доставки диска или предоставить возможность скачивания программы. Необходимо вести учет клиентов, поступивших оплат, оформленных и выполненных заказов.

Вариант 6. Модуль «Внешкольный учебный центр».

Примерное описание: Внешкольный учебный центр предоставляет возможность пройти обучение на платных курсах. Обучающиеся выбирают предметы, вносят оплату. Определенная часть оплаты перечисляется преподавателю. Необходимо вести учет обучающихся, внесенных оплат, закрепления курсов за преподавателями, проведенных занятий, предусмотреть возможность оформления и печати отчетов, предупреждения обучающихся по электронной почте об отмене/переносе занятий.

Вариант 7. Модуль «Кафедра».

Примерное описание: Преподаватели кафедры могут вести занятия на нескольких факультетах университета. Преподавание ведется по учебному плану, составленному для каждого направления подготовки. Учебные дисциплины могут входить в обязательную часть или вариативную (профильную), устанавливаемую университетом. Каждая дисциплина закрепляется за преподавателем. Необходимо вести учет нагрузки преподавателей, распределения дисциплин, закрепления студентов за научными руководителями при выполнении курсовых и дипломных работ, своевременного возвращения выданной студентам методической литературы, предусмотреть возможность оформления отчетов.

Вариант 8. Модуль «Деканат».

Примерное описание: После приема в университет в деканат передаются личные дела первокурсников, содержащих персональные данные (фамилия, имя, отчество, дата рождения, адрес и т.д.). В деканате происходит распределение по группам. Необходимо вести учет успеваемости, посещаемости студентов. Модуль должен хранить результаты экзаменов по курсам и семестрам, пройденные дисциплины. Предусмотреть возможность автоматического формирования и печати ведомостей и справок, учета выданных документов.

Вариант 9. Модуль «Школьный психолог».

Примерное описание: Школьный психолог проводит индивидуальные занятия со школьниками и тестирования. Индивидуальные занятия проводятся для желающих по записи. Тестирование проходит для всего класса по определенному графику. Модуль должен хранить данные об учениках (персональные данные, данные о родителях), характеристики, составленные психологом, тесты, результаты тестирования, отчеты о проведенных индивидуальных занятиях. Предусмотреть возможность оформления и печати отчетов, возможность уведомления классного руководителя и родителей в необходимых случаях, возможность составления или импорта новых тестов.

Вариант 10. Модуль «Бухгалтерия школы».

Примерное описание: Каждый из работников школы имеет табельный номер. Работниками являются учителя, администрация, технические работники. Зарплата зависит от разряда или квалификации, нагрузки (для учителей и членов администрации – в часах за неделю), отработанных дней, наличия/отсутствия больничных. Могут быть надбавки к зарплате за оказанные дополнительные платные услуги. Необходимо вести учет работников, расчет зарплаты, налоговых отчислений, формировать для каждого работника листок с данными о зарплате.

Вариант 11. Модуль «Дополнительные платные занятия».

Примерное описание: Университет предоставляет возможность пройти обучение на дополнительных платных курсах. Обучающиеся выбирают дисциплины, вносят оплату. Определенная часть оплаты перечисляется преподавателю. Необходимо вести учет обучающихся, внесенной оплаты, закрепления курсов за преподавателями, проведенных занятий, предусмотреть возможность оформления и печати отчетов, предупреждения обучающихся по электронной почте об отмене/переносе занятий.

Вариант 12. Модуль «Школьное питание».

Примерное описание: В школьной столовой предусмотрены комплексные обеды. Имеется список продуктов. Из данных продуктов по рецептам приготавливаются блюда. Модуль должен хранить информацию о продуктах (наличие/отсутствие, пищевая ценность – белки, жиры углеводы, калорийность, цена), рецепты, сформированные комплексные обеды, возможность автоматического расчета пищевой ценности и стоимости блюда, комплексного обеда. Предусмотреть возможность ввода новых рецептов, печати меню, нужных документов и отчетов.

Вариант 13. Модуль «Классный руководитель».

Примерное описание: У классного руководителя хранится информация об учениках (фамилия, имя, отчество, дата рождения, адрес, посещаемые кружки и секции), о родителях (фамилия, имя, отчество, адрес, место работы, контактные телефоны и т.д.). Модуль должен хранить информацию об успеваемости учеников, о посещении занятий, данные об участии в мероприятиях, информацию о проведенных мероприятиях в классе, характеристики учеников. Предусмотреть возможность оформления и печати нужных документов, возможность информирования о предстоящих/проведенных мероприятиях родителей по электронной почте.

Вариант 14. Модуль «Заместитель директора школы».

Примерное описание: Заместитель директора школы отвечает за учебный процесс, успеваемость учеников и посещение ими занятий. Распределяет нагрузку учителей (закрепление предметов за учителями). Необходимо вести учет нагрузки учителей, учет проведенных уроков, учет оценок, пропущенных уроков по классам, по параллелям, по школе. Предусмотреть возможность автоматического формирования и печати отчетов, содержащих данные о средних оценках по предмету, о средней оценке у каждого учителя, по классам.

Вариант 15. Модуль «Административно-хозяйственная часть дошкольного учреждения».

Примерное описание: Заведующий административно-хозяйственной частью (АХЧ) производит закупку материальных средств. Возможна закупка мебели, оргтехники, инструментов, книг, игрушек, моющих средств и др. Каждый закупленный товар закрепляется за работником учреждения. Возможно, что оргтехника передается другому работнику или проводится ремонт. Необходимо вести учет закупленных средств, их стоимости, проведенного ремонта, материально-ответственных лиц. При окончании срока службы производится списание средства.

Вариант 16. Модуль «Административно-хозяйственная часть школы».

Примерное описание: Административно-хозяйственная часть школы производит закупку материальных средств. Возможна закупка школьной мебели, оргтехники, компьютеров, книг, моющих средств. Каждый закупленный товар закрепляется за работником учреждения. Мебель, компьютеры, оргтехника дополнительно закрепляются за определенным кабинетом. Возможна их перестановка, которая должна учитываться у заведующего административно-хозяйственной частью. Возможен ремонт. Необходимо вести учет закупленных средств, их стоимости, материально-ответственных лиц, состояния, проведенно-

го ремонта. По окончанию срока службы производится списание средства.

Вариант 17. Модуль «Школьный медпункт».

Примерное описание: Школьный врач/медсестра для каждого школьника ведет карточку, куда вводятся данные о состоянии здоровья, проведенных медосмотрах, антропометрические данные (рост, вес и пр.), выполненные прививки, перенесенные заболевания, наличие/отсутствие аллергии. Необходимо вести учет всех перечисленных данных, предусмотреть информирование родителей и классного руководителя, выдачу направления к врачу-специалисту в поликлинику, оформление справки о состоянии здоровья, сертификата о прививках, учет выданных справок.

Вариант 18. Модуль «Спортивные секции».

Примерное описание: В спортивном комплексе ведется учет студентов, посещающих секции. Имеются платные и бесплатные занятия. За бесплатные занятия деньги перечисляет университет, в котором учится студент. Для этого студент должен предъявить студенческий билет. Определенный процент поступившей оплаты перечисляется в виде надбавки преподавателям-тренерам. Для постоянных клиентов предусмотрены скидки. Необходимо вести учет студентов, занятость спортивных залов, проведенных занятий, руководителей секций, внесенной оплаты.

Вариант 19. «Музыкальная школа».

Примерное описание: В музыкальной школе несколько отделов: фортепиано, струнных инструментов, духовых инструментов. Ученики по специальности (инструмент) прикрепляются к отдельному учителю, занятия проходят индивидуально. По таким дисциплинам, как хор и сольфеджио, занимаются в группе. Необходимо вести учет успеваемости и посещения учеников, результатов сдачи экзаменов, учет предметов, которые ведутся конкретными учителями. Предусмотреть возможность информирования родителей через электронную почту о предстоящих собраниях и концертах.

Вариант 20. Модуль «Курсы по подготовке к школе».

Примерное описание: При школе организованы курсы для будущих первоклассников. Родители вносят оплату, с ними составляется договор. Предметы закрепляются за учителями. Необходимо вести учет обучающихся, посещения занятий, результатов выполненных контрольных работ и тестов, договоров, поступивших оплат, списков групп, закрепленных за учителем предметов, предусмотреть возможность отправки сообщений по электронной почте при переносе или отмене занятий, возможность оформления и печати отчетов.

Вариант 21. Модуль «Прокат спортивного инвентаря».

Примерное описание: В спорткомплексе университета осуществляется прокат спортивного инвентаря. Студенты данного университета могут пользоваться услугами проката бесплатно. Посторонние клиенты могут зарегистрироваться через сайт или подойти лично. Для постоянных клиентов предусмотрены скидки. Необходимо вести учет клиентов, их персональных данных, имеющегося в наличии и выданного инвентаря, поступивших оплат, пожеланий клиентов. Предусмотреть возможность отправки сообщения по электронной почте о сроках возврата, о новых услугах, возможность формирования и печати отчетов.

Вариант 22. Модуль «Документация школы».

Примерное описание: Администрация школы хранит персональные данные учеников и их родителей, учебные планы по классам и календарные планы по предметам. Ввод информации осуществляет делопроизводитель. Данные об учениках может просматривать классный руководитель. Учебные планы для просмотра доступны всем. Календарные планы по предметам доступны для просмотра учителям-предметникам. Необходимо предусмотреть хранение всех введенных данных, возможность формирования и печати списков, отчетов, возможность отправки сообщения учителям при обнаружении неточностей в документах.

Вариант 23. Интернет-тестирование школьников.

Примерное описание: По графику управления образования проходит тестирование школьников по основным предметам. Для тестирования выделяется определенное время, всем участникам отправляются логины и пароли. Необходимо вести учет вопросов тестов, самих тестов, данных об участниках, результатов тестирования, определения средних баллов школьников по школам, классам, предметам. Предусмотреть возможность формирования и печати отчетов, отправки сообщения школьникам и школам результатов тестирования по электронной почте.

Вариант 24. Модуль «Студенческое общежитие».

Примерное описание: По направлению деканата в общежитие приходят студенты. Комендант расселяет их по свободным комнатам, по возможности учитывая, на каком факультете и курсе обучается студент. Проживающие могут подать заявку об устранении неисправностей оборудования. Необходимо вести учет комнат (этаж, площадь и т. д.), учет проживающих, их персональных данных, поступивших оплат за проживание, выявленных нарушений, заявок на ремонт оборудования или мебели. Предусмотреть возможность формирования и печати отчетов, отправки данных о свободных местах в деканаты.

ЛИТЕРАТУРА

Тема 1. Методология функционального моделирования SADT.

1. Марка Д.А. Методология структурного анализа и проектирования SADT / Д.А. Марка, К. МакГоуэн. – М.: МетаТехнология, 1993. – 243 с.

2. Калянов Г.Н. CASE. Структурный системный анализ (автоматизация и применение) / Г.Н. Калянов. – М.: Лори, 1996.

3. Заботина Н.Н. Проектирование информационных систем / Н.Н. Заботина. – М.: ДРОФА, 2013. – 336 с.

4. Грекул В.И. Проектирование информационных систем: учебник и практикум / В.И. Грекул, Н.Л. Коровкина, Г.А. Левочкина. – М.: Юрайт, 2017. – 386 с.

5. Ипатова Э.Р. Методологии и технологии системного проектирования информационных систем: учебник / Э.Р. Ипатова, Ю.В. Ипатов. – 2-е изд., стер. – М.: Флинта, 2016. – 257 с.

Тема 2. Методология RAD.

1. Бессмертная классика Waterfall – URL: https://worksection.com/blog/waterfall.html (дата обращения 25.12.2024).

2. What is Rapid Application Development Model (RAD) – URL: https://hackr.io/blog/rapid-application-development-model (дата обращения 25.12.2024).

3. Вольфсон Б.И. Гибкое управление проектами и продуктами / Б.И. Вольфсон. – СПб.: Питер, 2014. – 144 с.

4. A Practical Guide to Seven Agile Methodologies, Part 2. – URL: http://www.devx.com/architect/article/32836/1954 (дата обращения 25.12.2024).

5. Грекул В.И. Проектирование информационных систем. Практикум: учебное пособие / В.И. Грекул, Н.Л. Коровкина, Ю.В. Куприянов. – М.: ИНТУИТ, 2012. – 187 с.

6. Гудков Е.А. Анализ каскадной, итерационной и спиралевидной моделей внедрения корпоративных информационных систем / Е.А. Гудков, А.М. Деревнина, Н.С. Катасонова // Корпоративные информационные системы. – 2018. – № 1. – С. 18–29. – URL: https://corpinfosys.ru/archive/issue-1/48-2018-1-models (дата обращения 25.12.2024).

Тема 3. Методология RUP.

1. Крачтен Ф. Введение в Rational Unified Process / Ф. Кратчен. – М.: Вильямс, 2002. – 240 с.

2. Гвоздева Т.В. Проектирование информационных систем / Т.В. Гвоздева, Б.А. Баллод. – Ростов-на-Дону: Феникс, 2009. – 512 с.

3. Адизес И.К. Управление жизненным циклом корпорации / И.К. Адизес. – М.: Манн, Иванов и Фарбер, 2014. – 500 с.

4. Бергстрем С. Rational Unified Process – путь к успеху: рук. по внедрению RUP / С. Бергстрем, Л. Роберг. – М.: Кудиц-Образ, 2004. – 256 с.

5. Иванов Д.Е. Жизненные стадии и циклы организации / Д.Е. Иванов. – М.: Парта, 2005. – 75 с.

Тема 4. Диаграммы потоков данных DFD.

1. Методология DFD. – URL: https://sites.google.com/site/anisimovkhv/learning/pris/lecture/tema6/tema6_3 (дата обращения 25.12.2024).

2. Калашян А.Н. Структурные модели бизнеса – DFD-технологии / А.Н. Калашян, Г.Н. Калянов. – М.: Финансы и статистика, 2009.

3. Диаграммы потоков данных (DFD): как использовать их для описания движения данных в бизнес-процессах. – URL: https://www.antonpiskun.pro/diagrammy-potokov-dannyh-dfd-kak-ispolzovat-ih-dlya-opisaniya-dvizheniya-dannyh-v-biznes-proczessah/ (дата обращения 25.12.2024).

Тема 5. Методология объектного проектирования на языке UML (UML-диаграммы).

1. Леоненков А.В. Самоучитель UML 2 / А.В. Леоненков. – СПб.: БХВ-Петербург, 2007. – 576 с.

2. Леоненков А.В. Объектно-ориентированный анализ и проектирование с использованием UML и IBM Rational Rose: учебное пособие / А.В. Леоненков. – М.: ИНТУИТ, Ай Пи Ар Медиа, 2020.

3. Боггс У. UML и Rational Rose / У. Боггс, М. Боггс. – М.: ЛОРИ, 2001. – 582 с.

4. Якобсон А. Унифицированный процесс разработки программного обеспечения / А. Якобсон, Г. Буч, Дж. Рамбо. – СПб.: Питер, 2002. – 496 с.

5. Введение в UML. – URL: https://intuit.ru/studies/courses/1007/229/info (дата обращения 25.12.2024).

7. Нотация и семантика языка UML – URL: http://www.intuit.ru/studies/courses/32/32/info (дата обращения 25.12.2024).

Тема 6. Архитектурный подход к проектированию информационных систем.

1. Архитектура информационных систем / Б.Я. Советов, А.И. Водяхо, В.А. Дубенецкий, В.В. Цехановский. – М.: Академия, 2012. – 288 с.

2. Гринфилд Дж. Фабрики разработки программ (Software Factories): потоковая сборка типовых приложений, моделирование, структуры и инструменты = Software Factories: Assembling Applications with Patterns, Models, Frameworks, and Tools: пер. с англ. / Дж. Гринфилд, К. Шорт, С. Кук и др. – М.: Диалектика, 2006. – 592 с.

3. Гамма Э. Приемы объектно-ориентированного проектирования. Паттерны проектирования / Э. Гамма, Р. Хелм, Р. Джонсон, Дж. Влиссидес. – СПб.: Питер, 2001. – 368 с.

4. Фаулер М. Архитектура корпоративных программных приложений: пер. с англ. / М. Фаулер. – М.: Вильямс, 2006. – 544 с.

5. Хоп Г. Шаблоны интеграции корпоративных приложений / Г. Хоп, Б. Вульф. – М.: Вильямс, 2007. – 672 с.

СОДЕРЖАНИЕ

Учебное издание

Невзорова Ольга Авенировна

МЕТОДОЛОГИИ ПРОЕКТИРОВАНИЯ ИНФОРМАЦИОННЫХ СИСТЕМ

Учебно-методическое пособие