МИНОБРНАУКИ РОССИИ
Федеральное государственное бюджетное образовательное учреждение
высшего образования
«САРАТОВСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ
ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ ИМЕНИ Н. Г. ЧЕРНЫШЕВСКОГО»

Кафедра системного анализа и
автоматического управления

РАЗРАБОТКА ОСНОВНЫХ МОДУЛЕЙ ПРИЛОЖЕНИЯ «PSYTESTSGU»

АВТОРЕФЕРАТ БАКАЛАВРСКОЙ РАБОТЫ

студента 5 курса 551 группы
направления 09.03.04 — Программная инженерия
факультета КНиИТ
Гудкова Андрея Ивановича

Научный руководитель
Старший преподаватель _____ М. В. Белоконь

Заведующий кафедрой
к. ф.-м. н., доц. _____ И. Е. Тананко

Саратов 2022

ВВЕДЕНИЕ

Актуальность темы. Широкое применение компьютерных технологий и постоянное увеличение объема информационных потоков вызывает постоянный рост интереса к программам для компьютерного тестирования по самым различным тематикам и областям, в том числе и области психологии.

Преподавателями факультета психологии СГУ в учебных целях использовался конструктор многошкальных психодиагностических опросников «ТестМейкер» ("$TestMaker$"). Данное программное обеспечение было закуплено в начале 2010-х годов и на сегодняшний день его использованию препятствуют следующие причины:

1. Программа работает по сети и часто не запускается на отдельных компьютерах из-за сбившихся настроек.
2. В текущем учебном году компьютеры в аудиториях факультета были переведены на операционную систему (ОС) $GNU/Linux$, в частности, на дистрибутив $Bodhi\ Linux$. Это обусловлено устаревшим аппаратным обеспечением, которое не позволяет комфортно работать с ОС семейства $Windows$. «ТестМейкер» был разработан для функционирования под управлением ОС $Windows$ и не может быть перенесён на $Linux$ без вмешательства разработчиков.

Поэтому было принято решение разработать приложение для создания и проведения тестов (в том числе по психодиагностике). В качестве среды разработки был выбран $Lazarus$ благодаря его кроссплатформенности, быстроте освоения и простоте поддержания дальнейшей разработки приложения.

Цель бакалаврской работы — разработать основные модули приложения $PsyTestSGU$.

Поставленная цель определила **следующие задачи**:

1. изучить теоретические сведения для проведения тестов по психодиагностике;
2. ознакомиться с средой разработки $Lazarus$;
3. изучить методы реализации импорта при помощи $JSON$ в среде разработки $Lazarus$;
4. разработать архитектуру приложения $PsyTestSGU$;
5. реализовать приложение $PsyTestSGU$ в среде разработки $Lazarus$.

Методологические основы написания оконных приложений на языке программирования $Lazarus$ представлены в работах Алексеева Е. Р., Чесноковой О. В., Кучера Т. В. [2], Попова Е. А. [3], Мансурова К. Т. [6], Гурикова С. Р. [7].

Практическая значимость бакалаврской работы. Практическая значимость заключается в важности и необходимости создания приложения $PsyTestSGU$, поскольку оно придет на замену устаревшему программному обеспечению, работающему исключительно под управлением операционной системы Windows. Данная программа позволяет составлять и проходить тесты (в том числе по психодиагностике) на операционной системе Linux, на которую переведены персональные компьютеры, установленные в аудитории факультета психологии, а также благодаря одному из модулей программы имеется возможность просмотра результатов, пройденных студентами тестовых заданий.

Структура и объем работы. Бакалаврская работа состоит из введения, 5 разделов, заключения, списка использованных источников и одного приложения. Общий объем работы — 71 страница, из них 38 страниц — основное содержание, включая 21 рисунок, список использованных источников информации — 21 наименование.

КРАТКОЕ СОДЕРЖАНИЕ РАБОТЫ

Первый раздел «Теоретические основы изучения психодиагностики» посвящен теоретическим основам психодиагностики и особенностям психодиагностического тестирования. В разделе говорится, что психодиагностика представляет собой область психологической науки, при этом, также является важнейшей формой психологической практики, связанной с разработкой и использованием разнообразных методов распознания индивидуальных особенностей человека [1].

Психодиагностика затрагивает сразу три области психологического знания, такие как:

- предметная область (область психологии, которая изучает явления);
- психометрика (наука о измерении индивидуальных различий в диагностируемых переменных);
- практика использования психологического знания.

Существует несколько классификаций психодиагностических методик, наиболее полной из является классификация, разработанная Столиным В. В и Бодаевой А. А, согласно которой различают:

— диагностические методы, которые основаны на заданиях, предполагающих правильный ответ или же на заданиях, относительно которых правильных ответов не существует;
— вербальные и невербальные психодиагностические методики.

Особенности психодиагностического тестирования заключаются в том, что психодиагностический метод отличается определенной спецификой по отношению к традиционным исследовательским методам психологии – не экспериментальному (описательному) и экспериментальному. Основной особенностью его является измерительно-испытательная направленность, за счет которой достигается количественная (и качественная) квалификация изучаемого явления. Это возможно за счет выполнения определенных требований:

— Стандартизация инструмента измерения в основе которой лежит понятие нормы, поскольку индивидуальная оценка (например, успешность выполнения того или иного вида задания) может быть получена путем сопоставления с результатами других обследуемых.
— Надежность – неотъемлемая характеристика методики, отражающая точность психодиагностических измерений, а также устойчивость результатов теста к действию посторонних случайных факторов.
— Валидность – комплексная характеристика теста, включающая сведения об области исследуемых явлений и репрезентативности диагностической процедуры по отношению к ним.

Также в разделе рассматривается шкала оценки потребности в достижении – тест, который можно пройти в разработанном приложении.

Второй раздел «Среда разработки $Lazarus$**»** посвящен описанию основных возможностей среды разработки $Lazarus$ и ее преимуществ. Также в разделе говорится, что $Lazarus$ представляет собой специальную программу с интегрированной средой для разработки на основе компилятора $Free\ Pascal$ [10]. Интегрированная среда разработки предоставляет возможность кроссплатформенной разработки приложений в $Delphi$-подобном окружении, а также позволяет достаточно несложно переносить $Delphi$-программы с графическим интерфейсом в различные операционные системы: $Linux$, $FreeBSD$,

Mac OS X, *Microsoft Windows* [2–3].

Lazarus основан на библиотеке визуальных компонентов *Lazarus Component Library* (*LCL*). На данный момент поддерживается следующими типами интерфейсов: *WIN32 GDI*, *GTK* + 1.2.*x* (*Unix*, *Mac OS X*), *GTK* + 2.*x*, *Qt* 4 (*C* + +) и *Windows*. Поддерживает преобразование проектов *Delphi* [6–7]. Реализован основной набор элементов управления. Редактор форм и инспектор объектов максимально приближены к *Delphi*. Внешний вид показан на рисунке 1.

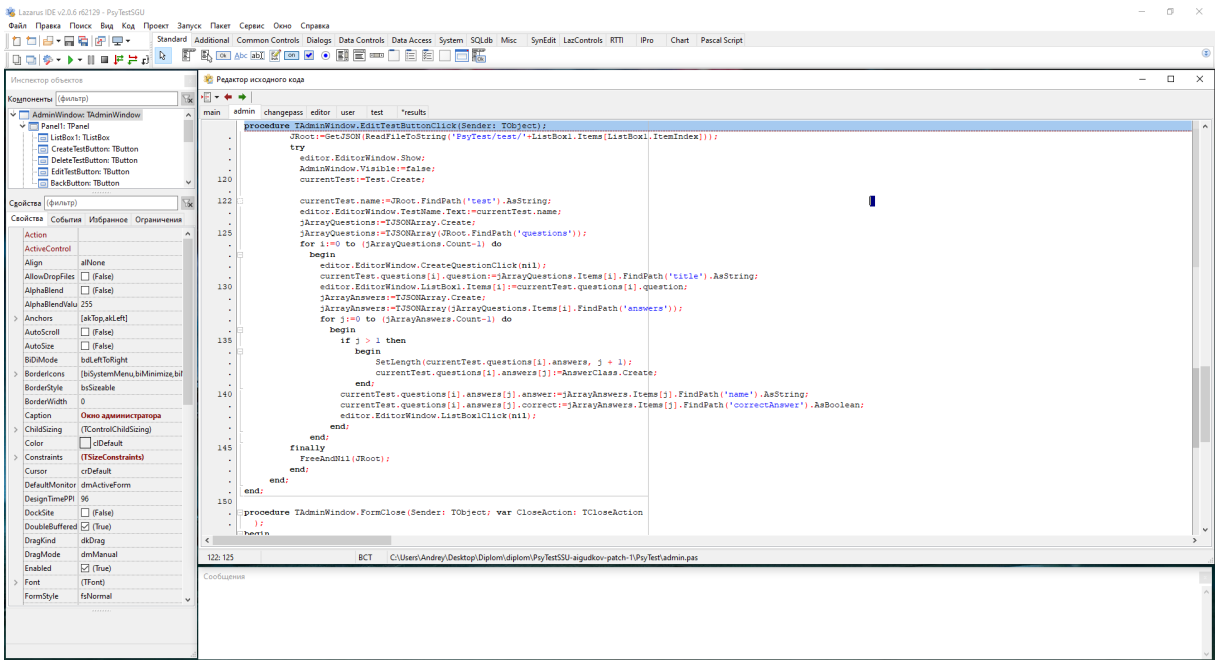

Рисунок 1 – Внешний вид среды разработки

Lazarus – это плод труда большого общества программистов со всего мира, в котором российские программисты занимают одно из ключевых мест. Любой может участвовать в этом проекте и безвозмездно пользоваться им, без опаски стать нелегалом в компьютерном мире.

Третий раздел «Импорт данных при помощи использования *JSON***»** посвящен основным понятиям и способам описания *JSON*, а также приведен пример использования *JSON* в среде разработки *Lazarus*.

В подразделе 3.1 описываются основные теоретические сведения формата *JSON*. Говорится о том, что *JSON* (*JavaScript Object Notation*) – простой формат обмена данными, удобный для чтения и написания, а также, что данный формат основан на двух структурах данных:

— коллекция пар ключ/значение (в разных языках, эта концепция реализова-

на как объект, запись, структура, словарь, хэш, именованный список или ассоциативный массив);

— упорядоченный список значений (в большинстве языков это реализовано как массив, вектор, список или последовательность) [4–5].

В подразделе также описывается общий вид каждого поддерживаемого $JSON$'ом формата представления данных.

Подраздел 3.2 посвящен реализации $JSON$ в среде разработки $Lazarus$. В подразделе рассмотрена библиотека $fp - json$ и ее возможности, а также приведен пример $JSON$-объекта и подробно расписан способ импорта данного объекта в среде разработки $Lazarus$.

Четвертый раздел «Разработка архитектуры приложения $PsyTestSGU$» посвящен реализации диаграмм отображающих структуру приложения и описанию работы основных модулей программы. В разделе говорится, что залог успешно и качественно выполненного проекта, заключается в правильном составление и минимизации изменений архитектуры приложения. Для разработки архитектуры приложения $PsyTestSGU$ было выбрано два варианта описания архитектуры: сначала схематично, для отображения основных модулей, затем описательно.

В приложении $PsyTestSGU$ для изображения основных структур выбрано два варианта диаграмм:

— диаграмма активности – демонстрирует последовательность состояний и действий системы;

— диаграмма классов – демонстрирует общую структуру иерархии классов системы, их кооперации, атрибутов, методов, интерфейсов и взаимосвязей между ними [8–9].

При построении диаграммы активности которая показана на рисунке 2, было выделено семь основных окон для работы приложения:

1. Главная форма $PsyTestSGU$.
2. Окно администратора.
3. Окно пользователя.
4. Результаты.
5. Изменение пароля.
6. Окно редактора.
7. Окно прохождения теста.

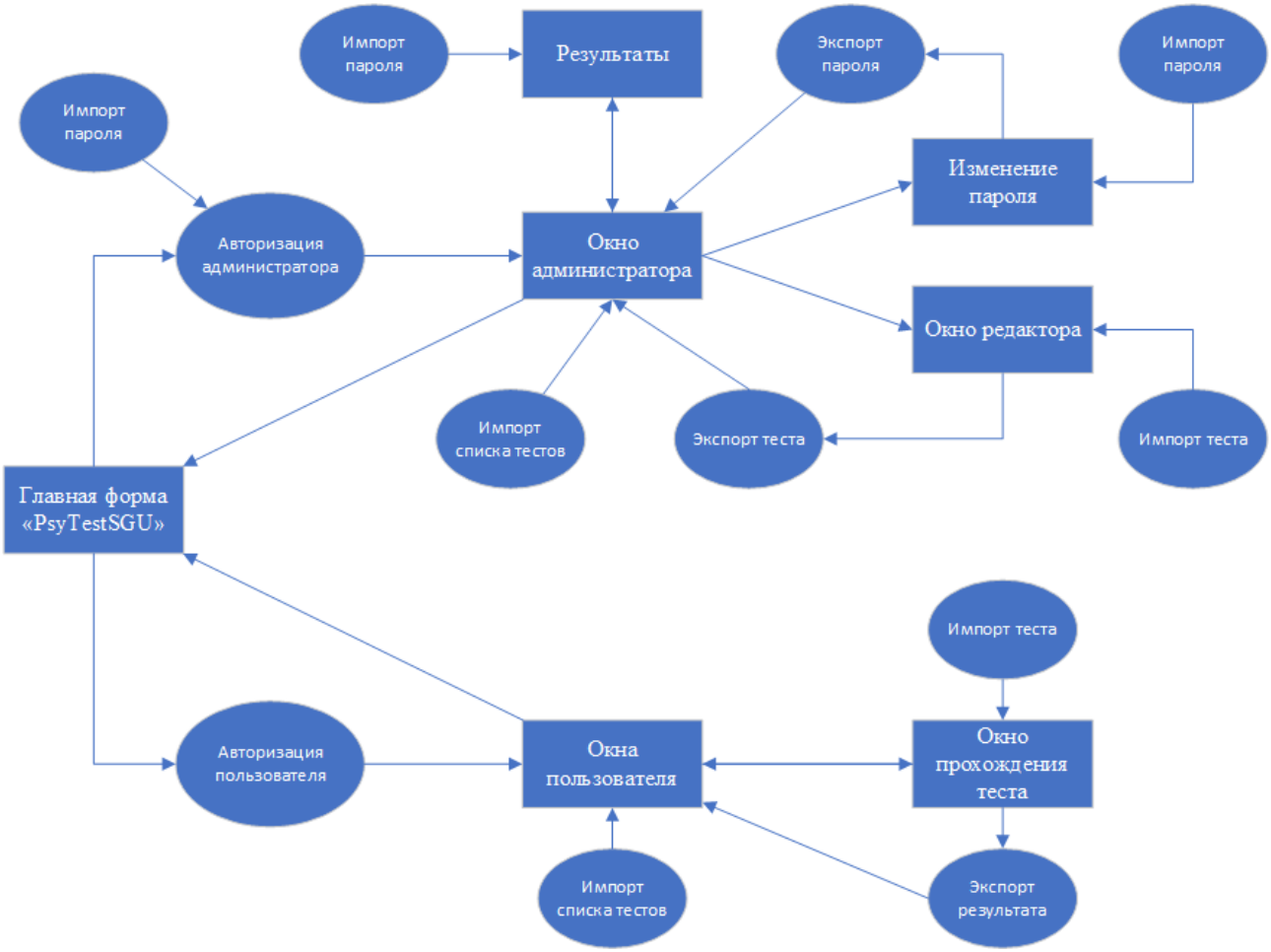

Рисунок 2 – Диаграмма активности

Также была построена диаграмма классов показанная на рисунке 3.

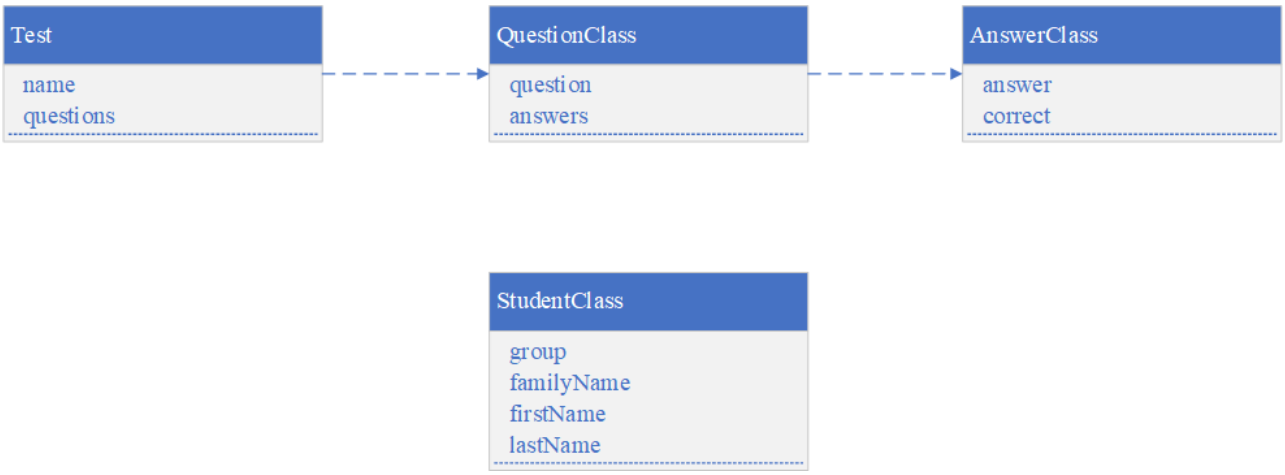

Рисунок 3 – Диаграмма классов

Построение диаграммы активности позволило четко описать алгоритм работы продукта, а построение диаграммы классов позволило разбить целый проект на задачи. Благодаря чему получилось выстроить более предсказуемый

процесс разработки, улучшить качество приложения, а также определить более точные временные рамки.

Пятый раздел «Программная реализация приложения $PsyTestSGU$**»** посвящен описанию функциональных возможностей каждому из семи модулей:

1. Главная форма $PsyTestSGU$.
2. Окно администратора.
3. Окно пользователя.
4. Результаты.
5. Изменение пароля.
6. Окно редактора.
7. Окно прохождения теста.

А также в разделе подробно описывается реализация модуля «Окно администратора» (внешний вид окна представлен на рисунке 4), включающее следующие элементы:

— кнопка «Добавить»;
— кнопка «Удалить»;
— кнопка «Редактировать»;
— кнопка «Назад»;
— поле $ListBox$ для вывода списка тестов;
— компонент $MainMenu$.

Кнопка «Добавить» необходима для создания нового теста путем открытия формы «Окно редактора». На кнопку установлено событие $OnClick$.

Кнопка «Удалить» необходима для удаления выделенного в $ListBox$ теста. На кнопку установлено событие $OnClick$.

Кнопка «Редактировать» необходима для редактирования выделенного в $ListBox$ теста. На кнопку установлено событие $OnClick$.

Кнопка «Назад» необходима для закрытия текущей формы и возврата на предыдущую в ней реализована одна команда $close$.

Поле $ListBox$ необходимо для вывода всех существующих тестов в директории, заполняется во время создания формы «Окно администратора», либо после добавления нового теста.

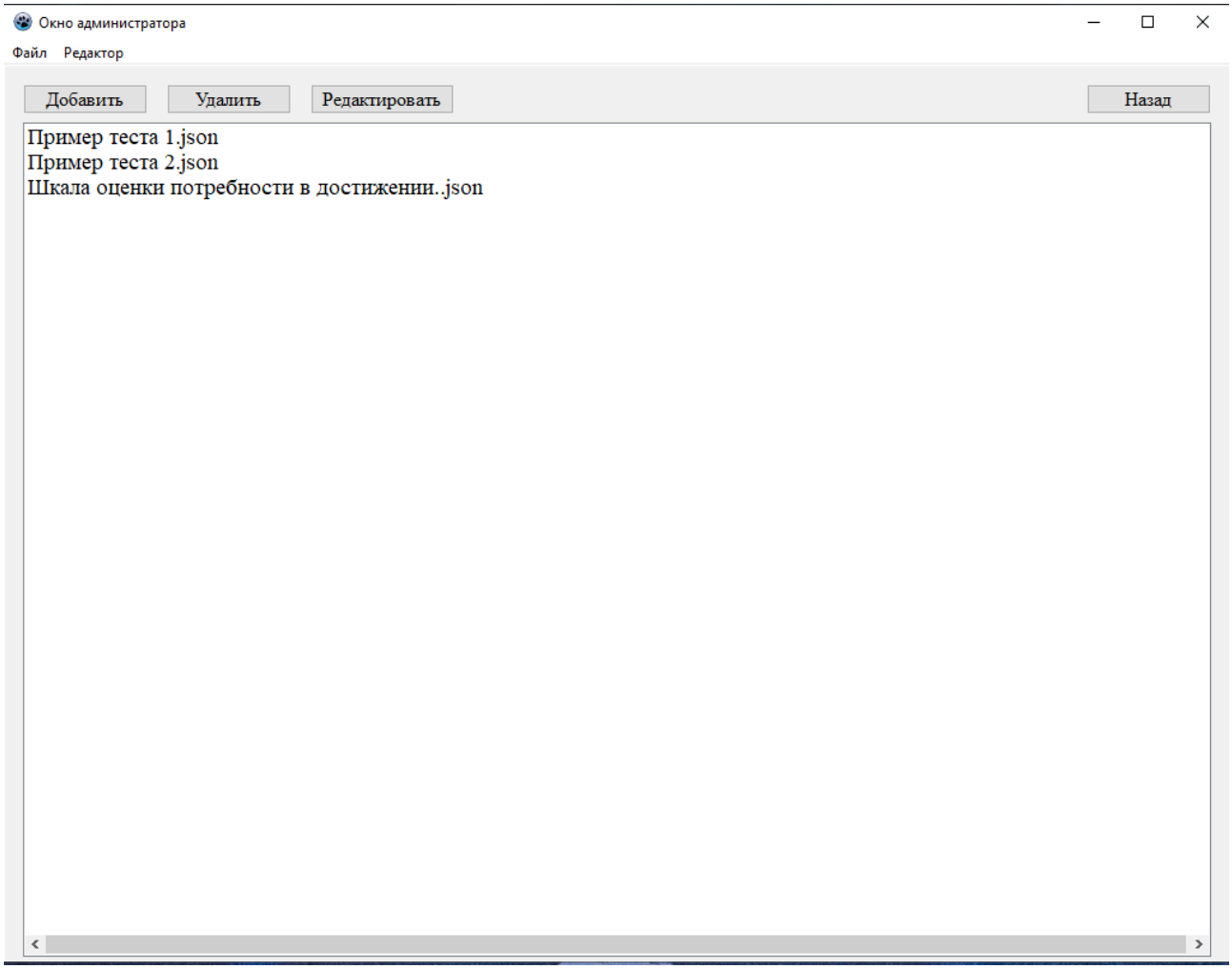

Рисунок 4 – Окно администратора

Компонент $MainMenu$ состоит из двух элементов «Редактора», который дублирует кнопки «Добавить», «Удалить», «Редактировать», и «Файла», в котором находятся под элементы «Результат», «Изменить пароль» и «Назад» (дублирует кнопку на форме с таким же названием). Под элементы «Результат» и «Изменить пароль» имеют по одному событию $OnClick$ выполнение которого соответствует вызову соответствующей формы.

ЗАКЛЮЧЕНИЕ

В результате выполнения данной выпускной квалификационной работы были изучены и реализованы основные понятия и методы создания приложений в среде разработки $Lazarus$, разработана архитектура приложения $PsyTestSGU$, а также написана программа по полученной архитектуре. В дополнении были приобретены знания и практические навыки, которые дают возможность применять изученные методы решения в дальнейшем.

В заключении были сделаны следующие выводы:

1. $Lazarus$ является удобной средой разработки для создания кроссплатформенных приложений на языке $Pascal$;
2. формат данных $JSON$ относительно прост, легко читается и записывается, а также поддерживается большим количеством языков;
3. построение диаграмм при проектировании архитектуры позволяет лучше понять алгоритм работы продукта, а также разбить целый проект на задачи, что способствует установлению более предсказуемых процессов разработки, улучшает качество приложения и помогает установить более точные временные рамки;
4. залог успешно и качественно выполненного проекта, заключается в правильном составлении и минимизации изменений архитектуры приложения.

Основные источники информации:

1. Карелин А. Большая энциклопедия психологических тестов. // М.: Эксмо, 2016. – 497 с.
2. Алексеев Е. Р., Чеснокова О. В., Кучер Т. В. Free Pascal и Lazarus. Учебник по программированию. // М.: ДМК Пресс, 2019. – 126 с.
3. Попов Е. А. Экспресс курс программирования в Lazarus. // СПб.: Питер, 2020. – 142 с.
4. Wallace J. JSON Quick Syntax Reference. // СПб.: Питер, 2016. – 76 с.
5. Smith B. Beginning JSON. // М.: ДМК Пресс, 2019. – 103 с.
6. Мансуров К. Т. Основы программирования в среде Lazarus. // М.: Нобель Пресс, 2018. – 381 с.
7. Гуриков С. Р. Основы алгоритмизации и программирования в среде Lazarus. // М.: Инфра-М, 2022. – 442 с.
8. Фаулер М. Шаблоны корпоративных приложений. // М.: Наука, 2016. – 126 с.
9. Мартин Р. Чистая архитектура. Искусство разработки программного обеспечения. // СПб.: Питер, 2022. – 357 с.
10. Абрамов В. Г. Введение в язык паскаль. // М.: Наука, 2018. – 672 с.