МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ
федеральное государственное бюджетное образовательное учреждение высшего образования
«Тольяттинский государственный университет»

Кафедра «Прикладная математика и информатика»
(наименование)

09.03.03 Прикладная информатика
(код и наименование направления подготовки / специальности)

Разработка программного обеспечения
(направленность (профиль) / специализация)

ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА (БАКАЛАВРСКАЯ РАБОТА)

на тему «Разработка программного обеспечения для автоматизации тестирования бизнес-приложений»

Обучающийся А.П. Чегодаев
(Инициалы Фамилия) (личная подпись)

Руководитель Н.Н. Казаченок
(ученая степень (при наличии), ученое звание (при наличии), Инициалы Фамилия)

Тольятти 2024

Аннотация

Тема бакалаврской работ «Разработка программного обеспечения для автоматизации тестирования бизнес-приложений».

Ключевые слова: разработка программного обеспечения, автоматизация тестирования, бизнес-приложение.

Объектом исследования бакалаврской работы является процесс тестирования бизнес-приложений.

Предметом исследования является программное обеспечение для автоматизации тестирования бизнес-приложений.

Цель выпускной квалификационной работы – разработка программного обеспечения для автоматизации тестирования бизнес-приложений.

Практическая значимость бакалаврской работы заключается в разработке ПО, обеспечивающего эффективную автоматизацию тестирования бизнес-приложений.

Данная работа состоит из введения, трех глав, заключения и списка используемой литературы и источников.

Первая глава посвящена формированию требований и постановке задачи на разработку ПО для автоматизации тестирования бизнес-приложений. Вторая глава посвящена проектированию ПО для автоматизации тестирования бизнес-приложений. В третьей главе описан процесс реализации и тестирования ПО для автоматизации тестирования бизнес-приложений.

В заключении описываются результаты выполнения выпускной квалификационной работы.

Бакалаврская работа состоит из 41 страницы текста, 14 рисунков, 3 таблиц и 30 источников.

Оглавление

Введение 4
Глава 1 Постановка задачи на разработку программного обеспечения для автоматизации тестирования бизнес-приложений 6
1.1 Методы автоматизированного тестирования бизнес-приложений . 6
1.2 Обзор и анализ аналогов программного обеспечения для автоматизации тестирования бизнес-приложений 7
1.3 Разработка требований к программному обеспечению для автоматизации тестирования бизнес-приложений 12
Глава 2 Проектирование программного обеспечения для автоматизации тестирования бизнес-приложений 15
2.1 Выбор методологии проектирования ПО 15
2.2 Логическое моделирование программного обеспечения для автоматизации тестирования бизнес-приложений 19
2.3 Моделирование данных системы автоматизации тестирования бизнес-приложений 25
Глава 3 Реализация и тестирование программного обеспечения для автоматизации тестирования бизнес-приложений 28
3.1 Реализация программного обеспечения для автоматизации тестирования бизнес-приложений 28
3.2 Тестирование программного обеспечения для автоматизации тестирования бизнес-приложений 31
Заключение 36
Список используемой литературы и используемых источников 38
Приложение А Фрагмент кода ПО для автоматизации тестирования бизнес-приложений 42

Введение

Одним из перспективных направлений в области управления производственно-хозяйственной деятельностью современных предприятий и компаний является активная интеграция в их ИТ-инфраструктуру высокотехнологичных бизнес-приложений.

Как показывает практика, наиболее востребованными в настоящее время являются бизнес-приложения, реализованные на основе современных технологических и CMS-платформ (1С: Предприятие 8.х, 1С-Битрикс, WordPress и др.).

Для обеспечения высокого качества бизнес-приложений необходимо выполнить их тестирование в процессе проектирования. Одним из наиболее эффективных способов решения данной проблемы является использование средств автоматизированного тестирования, разработанных специально для используемых программных платформ.

В этой связи представляет научно-практический интерес разработка программного обеспечения для автоматизации тестирования бизнес-приложений.

Объектом исследования бакалаврской работы является процесс тестирования бизнес-приложений.

Предметом исследования является программное обеспечение для автоматизации тестирования бизнес-приложений.

Цель выпускной квалификационной работы – разработка программного обеспечения для автоматизации тестирования бизнес-приложений.

Для достижения данной цели необходимо решить следующие задачи:

- сформулировать требования и произвести постановку задачи на разработку программного обеспечения для автоматизации тестирования бизнес-приложений;
- выполнить проектирование программного обеспечения для автоматизации тестирования бизнес-приложений;

– реализовать программное обеспечение для автоматизации тестирования бизнес-приложений и провести его тестирование.

Методы исследования – методы и технологии тестирования программного обеспечения (ПО), методы и технологии разработки ПО.

Практическая значимость бакалаврской работы заключается в разработке ПО, обеспечивающего эффективную автоматизацию тестирования бизнес-приложений.

Данная работа состоит из введения, трех глав, заключения и списка используемой литературы и источников.

Первая глава посвящена формированию требований и постановке задачи на разработку ПО для автоматизации тестирования бизнес-приложений.

Вторая глава посвящена проектированию ПО для автоматизации тестирования бизнес-приложений.

В третьей главе описан процесс реализации и тестирования ПО для автоматизации тестирования бизнес-приложений.

В заключении описываются результаты выполнения выпускной квалификационной работы.

Бакалаврская работа состоит из 41 страницы текста, 14 рисунков, 3 таблиц и 30 источников.

Глава 1 Постановка задачи на разработку программного обеспечения для автоматизации тестирования бизнес-приложений

1.1 Методы автоматизированного тестирования бизнес-приложений

Бизнес-приложение (Business application) – это программа или набор приложений, которые выполняют различные бизнес-функции (MES-системы, CRM-системы, ECM-системы и др.). Компании используют программное обеспечение для бизнеса, потому что оно помогает в принятии бизнес-решений, улучшает сотрудничество между заинтересованными сторонами и способствует общему оздоровлению бизнеса [30].

Как правило, бизнес-приложения используются в качестве компонентов корпоративной информационной системы (КИС) предприятия или корпоративного приложения.

Корпоративное приложение (Enterprise application, EA) – это большая программная системная платформа, предназначенная для работы в корпоративной среде, например, в бизнесе или правительстве. Примером EA является ERP-система.

ERP-системы являются сложными, масштабируемыми, основанными на компонентах, распределенными и критически важными. Программное обеспечение ERP-системы состоит из группы программ с общими бизнес-приложениями и утилитами организационного моделирования, предназначенными для непревзойденных функциональных возможностей [21].

«Следует также обратить внимание на популярность подходов к разработке бизнес- и корпоративных приложений на основе программных платформ, которые представляют собой среду исполнения и набор технологий, используемые в качестве основы для разработки специализированного ПО (1С: Предприятие 8, 1С-Битрикс, WordPress и

другие)» [15].

Автоматизированное тестирование – это метод тестирования ПО, основанный на применении инструментов и технологий, обеспечивающих сокращение усилий по тестированию и предоставление высокопроизводительных, быстрых и более доступных программные решения.

Автоматизированное тестирование позволяет создавать более качественное ПО с меньшими затратами [2].

Автоматизированное тестирование бизнес-приложений позволяет проводить глубокое тестирование бизнес-приложений, соответствующее реальным условиям их эксплуатации на различных предприятиях.

1.2 Обзор и анализ аналогов программного обеспечения для автоматизации тестирования бизнес-приложений

«По своим функциональным и архитектурным особенностям ПО для автоматизации тестирования бизнес-приложений относится к средам автоматизированного тестирования.

Среда автоматизированного тестирования (CAT) – это комплекс оборудования, программного обеспечения, данных и конфигураций, необходимых для выполнения тестовых примеров» [22].

1.2.1 Среда автоматизированного тестирования технологической платформы «1С: Предприятие 8»

«Для автоматизации процесса тестирования конфигурации бизнес-приложения, разработанного на платформе «1С: Предприятие 8.х» используется встроенный механизм автоматизации платформы", модель которого изображена на рисунке 1» [4].

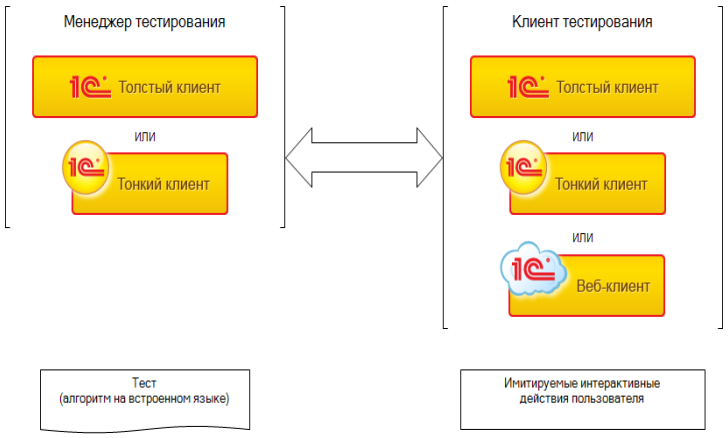

Рисунок 1 – Модель механизма автоматизированного тестирования платформы «1С: Предприятие 8.3»

«Имитация действий пользователя осуществляется при помощи набора объектов встроенного языка, предоставляющих доступ к логической модели интерфейса клиентского приложения и элементов форм.

При автоматизированном тестировании между собой взаимодействуют два клиентских приложения.

Первое – это менеджер тестирования, на котором исполняется алгоритм теста.

Второе – это клиент тестирования, который воспроизводит интерактивные действия пользователя.

В качестве менеджера тестирования может выступать толстый или тонкий клиент. Клиентом тестирования может быть любое из клиентских приложений: толстый клиент, тонкий клиент или веб-клиент.

Встроенный язык содержит ряд специализированных объектов, позволяющих на клиенте тестирования имитировать действия пользователя: навигация по прикладному решению, выполнение интерактивных команд системы, ввод данных в поля форм, чтение данных, отображаемых в форме, и т. д.

Результат выполнения автоматизированного теста может контролироваться визуально, либо программно, путем сравнения полученных результатов с эталонными значениями» [4].

1.2.2 Среда автоматизированного тестирования CMS-платформы «1С- Битрикс: Управление сайтом»

В CMS-платформе «1С-Битрикс: Управление сайтом» имеются встроенные средства для автоматизации некоторых видов тестирования бизнес-приложений:

- «модуль «Монитор производительности» предназначен для мониторинга параметров производительности сайта. Позволяет в абсолютных величинах протестировать конфигурацию приложения и общую производительность проекта;
- страница «Проверка системы» (Настройки > Инструменты > Проверка системы) предназначена для всесторонней проверки соответствия параметров системы, на которой осуществляется функционирование «1С-Битрикс: Управление сайтом», минимальным и рекомендуемым требованиям продукта» [1].

На рисунке 2 показана архитектура CMS-платформы «1С-Битрикс: Управление сайтом» со встроенным монитором производительности.

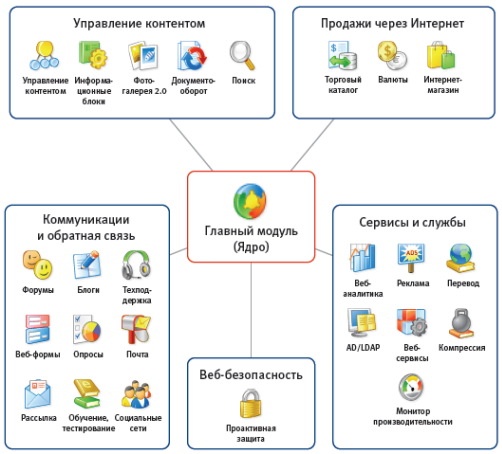

Рисунок 2 – Архитектура CMS-платформы «1С-Битрикс: Управление сайтом»

Данная САТ широко применяется для организации автоматизированного тестирования веб-приложений для бизнеса.

1.2.3 Среда Oracle Testing as a Service (TaaS)

Oracle Testing as a Service (TaaS) – это облачная платформа для предоставления услуг автоматизированного тестирования бизнес-приложений [24].

Это решение самообслуживания, предназначенное для частных облаков, которое организует сквозной процесс тестирования путем автоматизации предоставления тестовых лабораторий, включая тестируемое приложение и инструменты тестирования, выполняет сценарии загрузки и/или функционального тестирования для приложения, предоставляет широкие данные мониторинга и диагностики приложений для анализа, а также имеет сложную функцию возврата средств для измерения и взимания платы за использование тестового облака с конечных пользователей

Архитектура решения показана на рисунке 3.

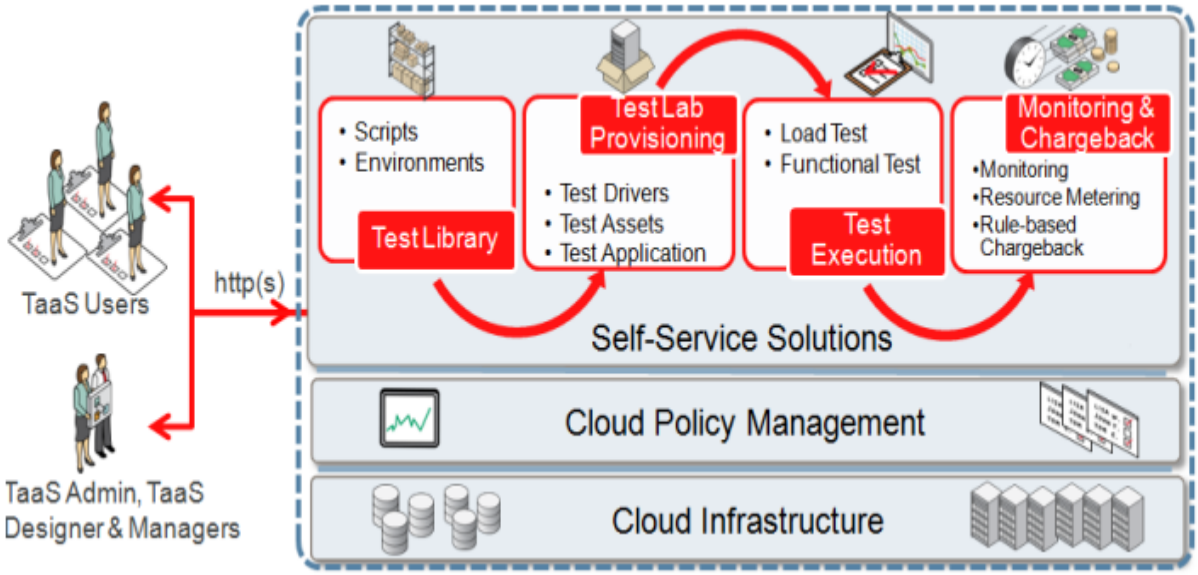

Рисунок 3 – Архитектура решения Oracle Testing as a Service

Поставляемое с помощью Enterprise Manager Cloud Control TaaS позволяет значительно сократить время и затраты на тестирование без ущерба для качества и позволяет организациям быть более гибкими в

доставке критически важных бизнес-приложений своим пользователям.

Рассмотренные CAT предназначены для автоматизации тестирования бизнес-приложений, разработанных на технологических платформах их вендоров.

Для сравнения характеристик рассмотренных аналогов ПО CAT для бизнес-приложений на предмет соответствия основным требованиям составлена таблица 1.

Таблица 1 – Сравнение характеристик аналогов ПО CAT для бизнес-приложений

Характеристика/балл (0-3)	CAT «1С: Предприятие 8»	CAT «1С-Битрикс: Управление сайтом»	Oracle Testing as a Service
Автоматизация тестирования бизнес-приложений	3	3	3
Формирование отчетов тестирования	3	3	3
Широкая область применения	1	1	1
Отсутствие избыточного функционала	1	1	1
Низкая совокупная стоимость владения	2	2	1
Итого	10	10	9

Таким образом, основным требованиям к ПО для автоматизации тестирования бизнес-приложений полностью не соответствует ни одно из представленных ПО CAT.

Главным недостатком, ограничивающим возможности для широкого рименения рассмотренного ПО CAT, является его ориентация на программные платформы конкретных вендоров.

Следует также отметить, что дополнительным ограничением на использования продуктов компании SAP является то, что этот вендор в настоящее время не представлен на российском ИТ-рынке.

В этой связи представляет интерес разработка ПО, которое может использоваться для автоматизированного тестирования бизнес-приложений, разработанных на свободно распространяемых программных CMS платформах [12].

1.3 Разработка требований к программному обеспечению для автоматизации тестирования бизнес-приложений

Для разработки требований к ПО используем методологию FURPS+.

«FURPS+ – это методология проверки приоритетных требований после понимания потребностей клиента.

Методология FURPS+ в классификации требований делает упор на понимание различных типов функциональных и нефункциональных требований.

Функциональные требования описывают, что должно делать ПО, в то время как нефункциональные требования накладывают ограничения на то, как ПО будет это делать» [8].

Рассмотрим в качестве платформы для разработки бизнес-приложений популярную CMS WordPress.

WordPress — это онлайн-система управления контентом (CMS) с открытым исходным кодом. Это самый простой и популярный инструмент для создания сайтов и блогов. Можно создать любой стиль веб-сайта и блога по желанию разработчика. Основное преимущество заключается в том, что для создания веб-сайта не требуются специальные навыки программирования и дизайна [14].

В качестве языка разработки в WordPress используется язык PHP.

На платформе WordPress разрабатываются различные бизнес-

приложения.

Функционал этих бизнес-приложений формируется с помощью плагинов [20].

Плагин — это надстройка к программному обеспечению, которая устанавливается в программу и расширяет ее возможности.

Одной из проблем, с которыми может столкнуться разработчик бизнес-приложения, является несовместимость используемой версии PHP и версии PHP устанавливаемого плагина WordPress [17].

Решение данной проблемы заключается в тестировании на совместимость версий PHP с помощью специального ПО для автоматизации тестирования бизнес-приложений.

На основании проведенного анализа выработаны требования к данному ПО.

«Functionality (функциональные требования):

– автоматизация тестирования плагинов на совместимость версий PHP;

– формирование отчетов тестирования;

Usability (требования к удобству использования):

– дружественный интуитивный интерфейс;

– современный дизайн.

Reliability (требования к надежности):

– допустимая частота/периодичность сбоев: 1 раз в 300 часов;

– среднее время сбоев: 1 раб. день:

– возможность восстановления системы после сбоев: 1 раб. день.

Performance (требования к производительности):

– допустимое количество одновременно работающих пользователей: 3;

– время реакции на возникновение аварийной ситуации: 10 с.

Supportability (требования к поддержке):

– время устранения критических проблем: в течение рабочего дня.

Проектные ограничения:

– применение средств автоматизации тестирования, разработанных для используемой технологической платформы;
– широкие функциональные возможности;
– низкая совокупная стоимость владения» [26].

Представленный перечень требований является основой для разработки ПО для автоматизации тестирования бизнес-приложений.

Выводы к главе 1

Результаты, полученные при выполнении главы 1, позволили сделать следующие выводы:

– главным недостатком, ограничивающим возможности для широкого применения рассмотренных CAT является их ориентация на программные платформы конкретных вендоров;
– одним из наиболее эффективных способов решения данной проблемы является использование специализированной CAT, функциональные возможности которой позволяют тестировать используемые CMS совместно с различными плагинами;

Разработаны требования к программному обеспечению эффективной CAT.

Глава 2 Проектирование программного обеспечения для автоматизации тестирования бизнес-приложений

2.1 Выбор методологии проектирования ПО

Методологии разработки программного обеспечения являются основой для разработки и поддержки ПО. Они варьируются от строго предписывающих методов, таких как Agile и Scrum, до весьма общих, таких как методология водопада. Спектр вариантов может затруднить выбор процесса [11].

Однако выбор методологии требует рассмотрения нескольких факторов, включая требования к ПО, опыт и навыки разработки.

В основу каждой методологии разработки ПО положено понятия жизненного цикла ПО, который состоит из нескольких этапов.

Первый этап жизненного цикла разработки ПО, как правило, включает в себя планирование и требует наличия менеджера проекта, который отслеживает все задачи и может предсказать, сколько времени потребуется для выполнения каждой из них. Начальный этап планирования также включает в себя анализ требований, при котором менеджеры проектов сотрудничают с персоналом клиента, чтобы документировать, что необходимо сделать. Кроме того, они смотрят на затраты на персонал и другие ресурсы [7].

На втором этапе разработчики преобразуют требования в практические задачи. Тестирование начинается на третьем и четвертом этапах после завершения написания компьютерных программ, основанных на этих целях.

Разработчики обычно передают завершенные программные проекты группам обеспечения качества на пятом этапе. Эти специалисты гарантируют, что программа работает должным образом, прежде чем передать ее клиентам или сотрудникам компании, которые используют ее в своей повседневной работе.

Рассмотрим и сравним основные методологии разработки ПО: Waterfall, Scrum и RUP.

Водопадный подход (Waterfall) или каскадная модель разработки ПО – это последовательный, простой и систематический процесс, представленный на рисунке 4.

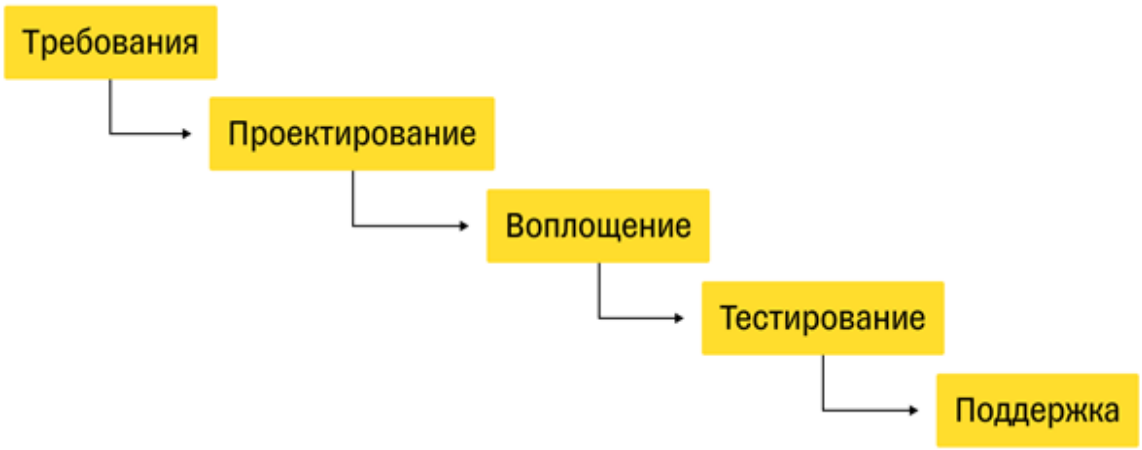

Рисунок 4 – Схема процесса разработки ПО в методологии водопада

При разработке ПО типичный водопадный процесс начинается с этапа планирования, на котором определяются и анализируются требования проекта для определения оптимального решения [29].

Следующий шаг – проектирование, за которым следуют этапы разработки и тестирования.

Наконец, есть реализация и поддержка производства. Этот метод работы придает структуру более крупным проектам, но может занять довольно много времени, поскольку включает в себя множество этапов.

Поскольку команда разработчиков должна завершить каждый шаг, прежде чем перейти к следующему, любые изменения на этом пути требуют значительной переработки предыдущих шагов, что в конечном итоге приводит к задержке поставки и увеличению затрат для всех участвующих сторон.

Scrum – это методология разработки ПО, которая помогает командам более эффективно работать вместе и достигать целей проекта [25].

Жизненный цикл (ЖЦ) проекта разработки ПО в методологии Scrum показан на рисунке 5.

Рисунок 5 – ЖЦ проекта разработки ПО в методологии Scrum

Существует множество различных вариантов Scrum.

Однако у них есть общие характеристики: роли, встречи, артефакты и правила.

Роли включают владельца продукта, Scrum-мастера и членов команды разработчиков и тестировщиков.

У артефактов есть бэклог, бэклог спринта, эпики, списки функций и задачи. Правила охватывают распределение ответственности за задачи, а также методы управления временем, такие как диаграммы сгорания.

Основная цель методологии RUP (Rational Unified Process) – предоставить модель для эффективной реализации коммерчески проверенных подходов к разработке для использования на протяжении всего жизненного цикла разработки ПО [5].

Следует отметить, что RUP – это не конкретная модель разработки, а скорее адаптивная и адаптированная к конкретным потребностям вашего проекта, команды или организации. RUP основана на нескольких фундаментальных идеях, таких как этапы разработки и строительные блоки, которые определяют, кто, что, когда и как будет происходить разработка.

«RUP снижает вероятность непредвиденных обстоятельств. затраты на разработку и предотвращение нерационального использования ресурсов.

RUP предлагает итеративный подход к проектированию и разработке ПО, основанный на спиральном жизненном цикле» [5].

Жизненный цикл ПО в методологии RUP представлен на рисунке 6.

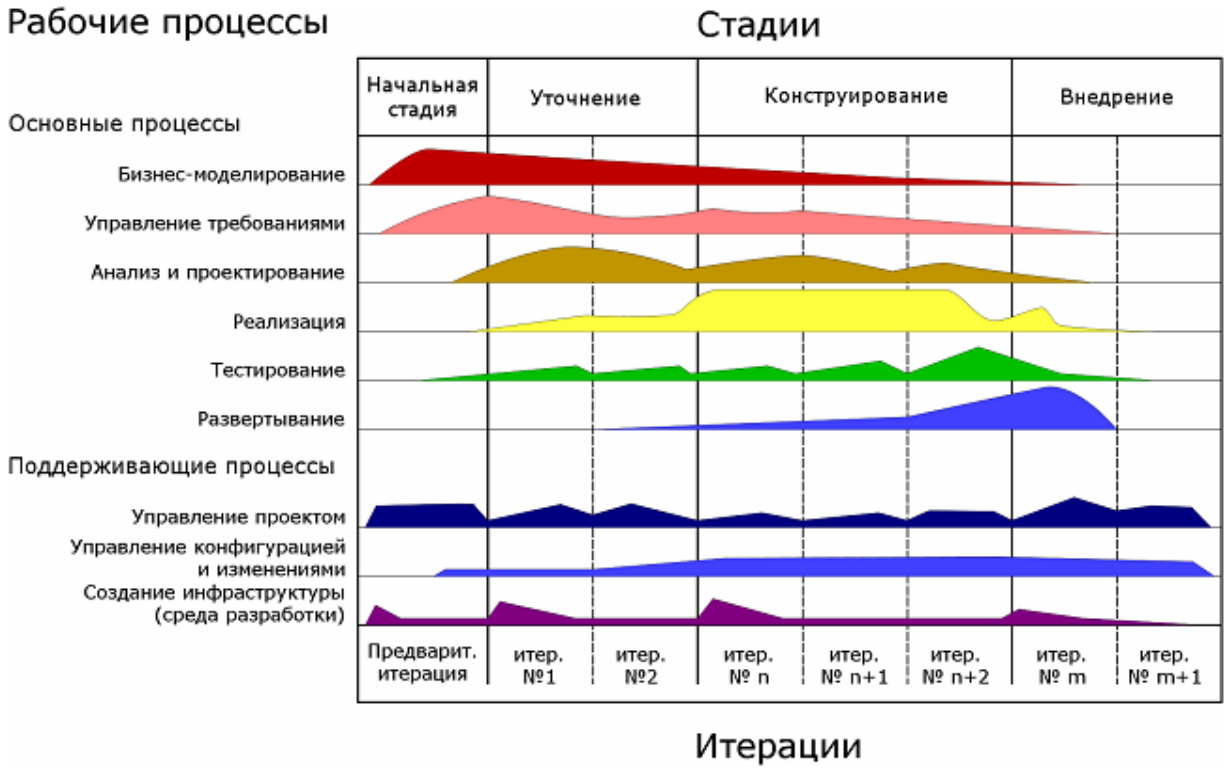

Рисунок 6 – Жизненный цикл ПО в методологии RUP

Для сравнения рассмотренных методологий используем таблицу 2.

Таблица 2 – Сравнение методологий разработки ПО

Методология разработки ПО	Достоинства	Недостатки
Waterfall	Отличная документация; плавная интеграция новых сотрудников. Высокоточная оценка затрат на разработку. Поддерживает этап тестирования. Конечный продукт соответствует ожиданиям.	Менее гибкий процесс. Более длительный срок доставки.
Scrum	Легкое устранение ошибок; подходит для динамичных проектов. Удобная процедура тестирования. Улучшает мотивацию команды Гибкость.	Строгое управление. Не подходит для большой команды.
RUP	Использует лучшие части модели Waterfall и включает их в более итеративный процесс, позволяющий вносить изменения. Подходит для небольших проектов.	Как и Waterfall, RUP также требует много процессов и может слишком сильно полагаться на отзывы заинтересованных сторон. Даже будучи итеративным процессом, он может быть слишком медленным для некоторых типов проектов.

По результатам сравнения достоинств и недостатком рассмотренных методологий выбираем для разработки ПО методологию RUP.

Главным преимуществом методологии RUP для рассматриваемого случая является возможность применения для небольших проектов.

2.2 Логическое моделирование программного обеспечения для автоматизации тестирования бизнес-приложений

Одним из основных этапов начальной фазы проектирования RUP является логическое моделирование ПО.

«На данной фазе проектирования определяются основные требования, ограничения и ключевая функциональность продукта. Создается базовая версия модели прецедентов.

Цель – определение основной функциональности ПО автоматизации тестирования бизнес-приложений.

Ключевые участники – системный аналитик и тестировщик.

Входная информация: перечень функциональных требований к ПО, методики тестирования бизнес-приложений.

Результат – функциональная модель ПО автоматизации тестирования бизнес-приложений, представленная в виде диаграммы вариантов использования ПО» [9].

Спецификация элементов диаграммы вариантов использования ПО представлена в таблице 3.

Таблица 3 – Спецификация элементов диаграммы вариантов использования ПО автоматизации тестирования бизнес-приложений

Элементы диаграммы	Описание
Акторы	Тестировщик, Разработчик, ПО для автоматизации тестирования
Варианты использования	Ввод заявки, Регистрация заявки, Распределение заявок, Выполнение заявки, Регистрация выполненной заявки, Формирование управленческого отчета

На рисунке 7 представлена диаграмма вариантов использования ПО автоматизации тестирования бизнес-приложений.

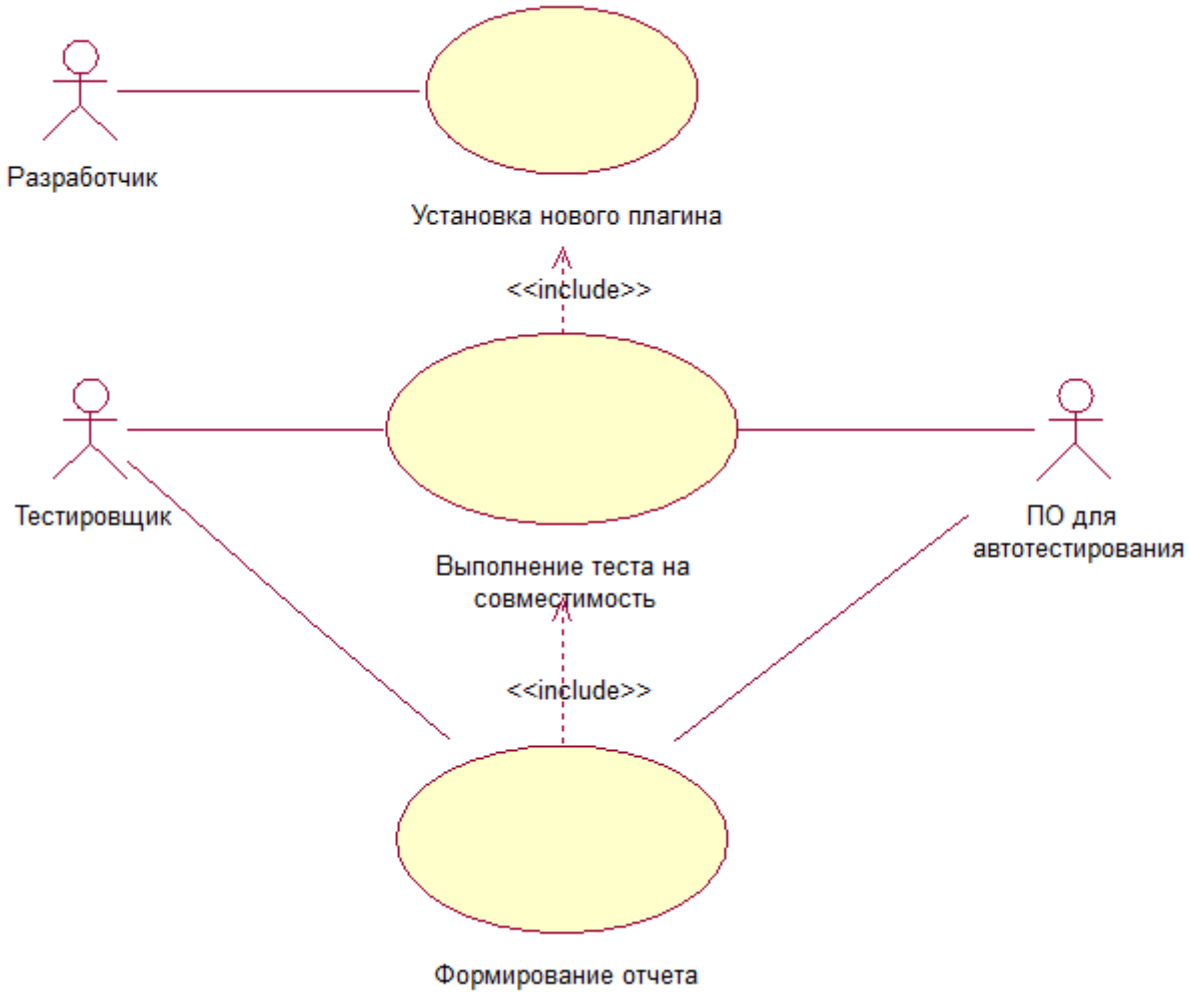

Рисунок 7 – Диаграмма вариантов использования ПО автоматизации тестирования бизнес-приложений

Диаграмма вариантов использования построена с точки зрения Тестировщика.

Разработанная диаграмма вариантов использования отражает функциональный аспект и является функциональной моделью ПО автоматизации тестирования бизнес-приложений.

Для отражения структурного аспекта ПО необходимо разработать его диаграмму классов.

Диаграмма классов относится к статическому виду диаграмм UML, который описывает структуру системы, а не ее поведение. Диаграмма классов помогает анализировать, проектировать и документировать программные системы.

На диаграмме классов классы представляются в виде прямоугольников, разделенных на три части: имя класса, атрибуты и операции. Отношения между классами показываются с помощью линий, которые могут иметь разные символы на концах, обозначающие тип и кратность отношения. Например, ассоциация, агрегация, композиция, наследование, реализация и т.д. [9].

Диаграмма классов ПО автоматизации тестирования бизнес-приложений представлена на рисунке 8.

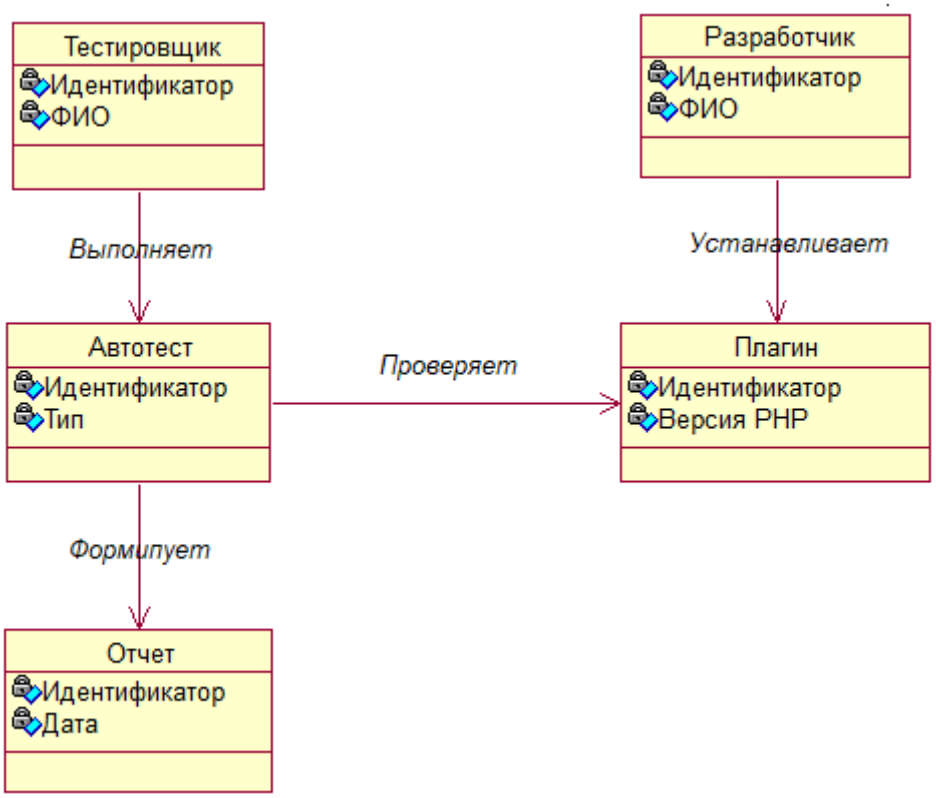

Рисунок 8 – Диаграмма классов ПО автоматизации тестирования бизнес-приложений

На фазе уточнения RUP создается архитектура ПО.

Цель – разработка программной ПО для обеспечения стабильной основы для части работ по проектированию и внедрению.

Ключевые участники – тестировщик и разработчик.

Входная информация: модель вариантов использования ПО.

Результат – архитектура ПО.

Для построения архитектуры ПО автоматизации тестирования бизнес-приложений используем диаграмму компонентов UML [28].

Диаграмма компонентов – это вид диаграмм UML, который изображает компоненты ПО и их взаимодействие. В ПО САТ компонентами являются подсистемы (модули), из которых она состоит.

Диаграмма компонентов ПО автоматизации тестирования бизнес-приложений показана на рисунке 9.

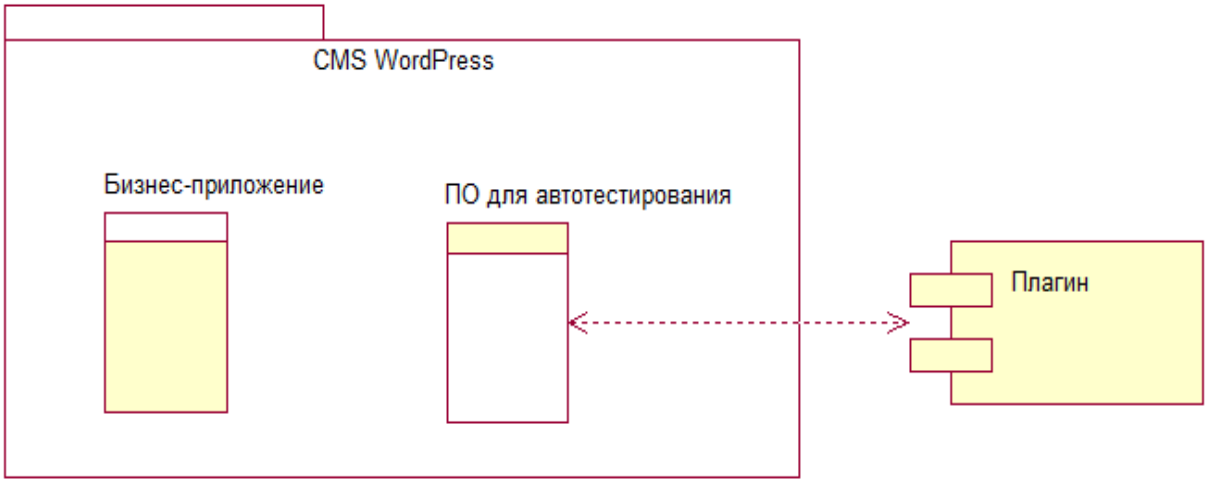

Рисунок 9 – Диаграмма компонентов ПО автоматизации тестирования бизнес-приложений

Архитектура ПО относится к набору структур, необходимых для построения программной системы [3].

Каждая структура включает элементы программного обеспечения, их отношения и свойства, связанные как с элементами, так и с отношениями.

«Для отображения динамического аспекта ПО используем диаграмму последовательности UML.

Диаграмма последовательности — это подмножество диаграмм

взаимодействия, которая представляет собой класс шаблона, созданный в UML.

Диаграммы последовательности — это диаграммы взаимодействия, которые показывают, как набор объектов взаимодействует друг с другом и в каком порядке» [28].

Эти шаблоны часто используются разработчиками программного обеспечения и бизнес-профессионалами для документирования существующего процесса и описания того, как группа объектов работает вместе.

По этой причине диаграммы последовательности иногда называют диаграммами событий или сценариями событий.

Диаграмма последовательности сценария выполнения автотеста плагина показана на рисунке 10.

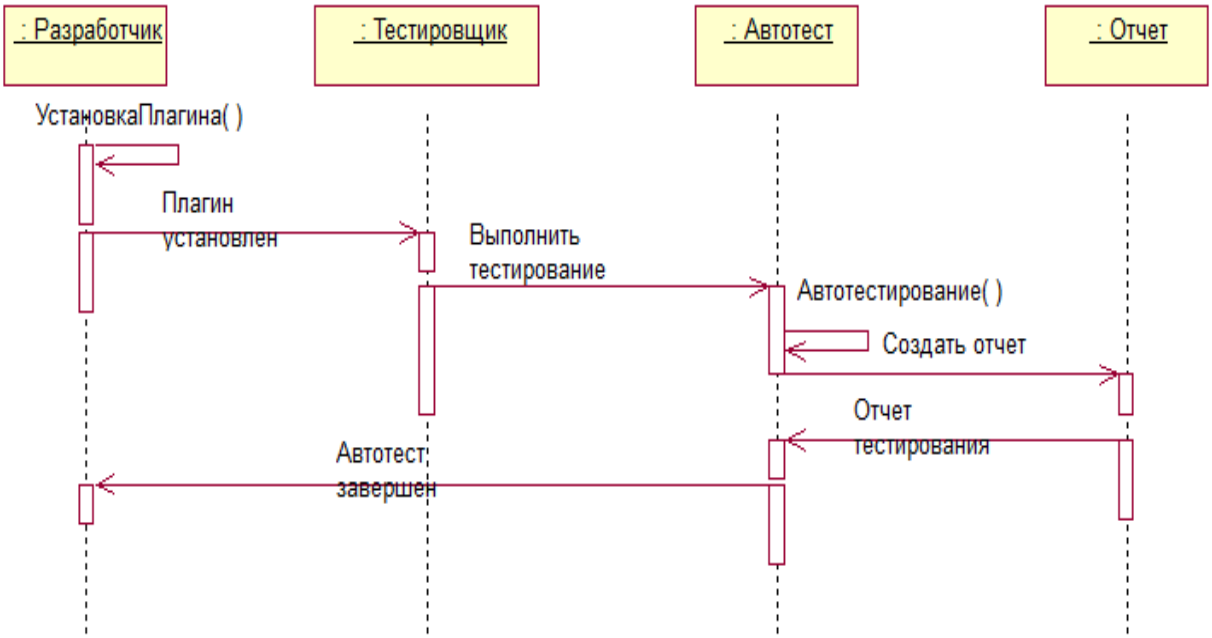

Рисунок 10 – Диаграмма последовательности сценария выполнения автотеста плагина

Представленная диаграмма отражает динамический аспект разработанного ПО.

2.3 Моделирование данных системы автоматизации тестирования бизнес-приложений

Для разработки модели данных САТ бизнес-приложений используем CASE-средство Workbench MySQL, которое поддерживает стандарт IDEF1X [6].

Выбор Workbench обусловлен тем, что в CMS WordPress используется СУБД MySQL.

Отличительной особенностью Workbench является то, что в данной среде отсутствует разделение модели базы данных на логическую и физическую [23].

Разработанная в Workbench модель представляет оба типа моделей базы данных.

В процессе разработки модели данных САТ бизнес-приложений из диаграммы классов ее ПО выделены следующие сущности:

- пользователь;
- автотест;
- плагин;
- отчет.

Для приведения данных к третьей нормальной форме введены сущности [10]:

- роль (тестировщик или разработчик);
- вид тестирования (проверка на совместимость версий PHP).

Между сущностями установлены следующие связи:

- пользователь-автотест – «один ко многим»;
- вид тестирования-автотест – «один ко многим»;
- плагин-автотест – «один ко многим»;
- автотест-отчет – «один ко многим»;
- роль-пользователь – «один ко многим».

На рисунке 11 показана модель данных CAT бизнес-приложений.

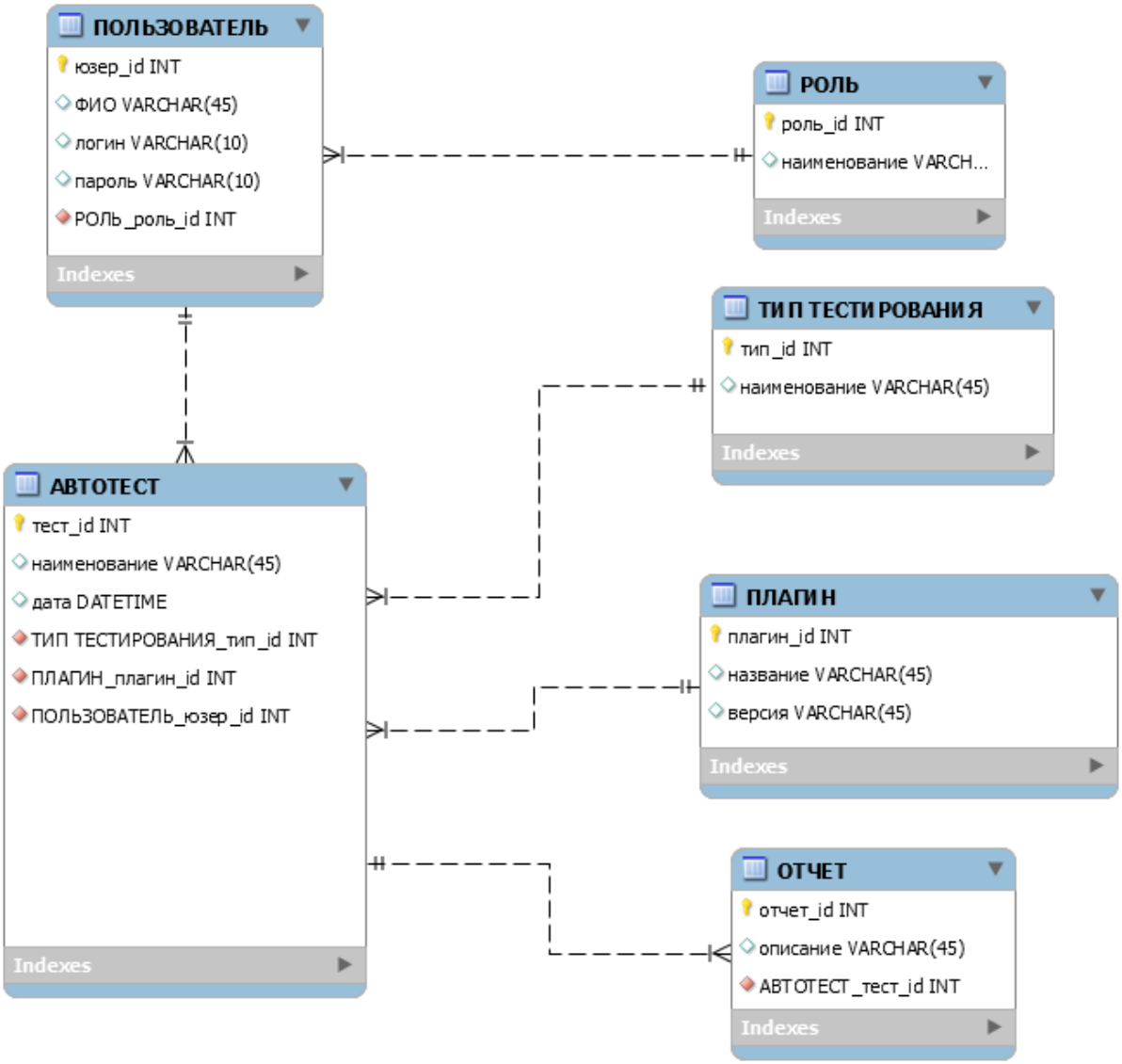

Рисунок 11 – Модель данных CAT бизнес-приложений

Все связи между сущностями модели – неидентифицирующие.

Выводы по главе 2

Результаты проделанной работы позволили сделать следующие выводы:

– выбор методологии проектирования ПО требует рассмотрения

нескольких факторов, включая требования к ПО, опыт и навыки разработки;

- логическое моделирование в значительной степени связано со структурированием рабочих процессов вокруг хорошо организованных объектов, представленных графически посредством визуального представления, обеспечиваемого диаграммами UML;
- по результатам сравнения достоинств и недостатком рассмотренных методологий выбрана методология RUP. Главным преимуществом методологии RUP для рассматриваемого случая является возможность применения для небольших проектов;
- архитектура ПО относится к набору структур, необходимых для построения программной системы.

Для разработки модели данных CAT бизнес-приложений выбрано CASE-средство Workbench MySQL, которое поддерживает стандарт IDEF1X.

Выбор Workbench обусловлен тем, что в CMS WordPress используется СУБД MySQL.

Глава 3 Реализация и тестирование программного обеспечения для автоматизации тестирования бизнес-приложений

3.1 Реализация программного обеспечения для автоматизации тестирования бизнес-приложений

WordPress написан на PHP, и поэтому он должен иметь возможность работать как минимум на минимальной поддерживаемой версии PHP, доступной веб-хостам.

Хотя WordPress рекомендует определенную минимальную версию PHP, старые версии PHP со временем устаревают и в ближайшем будущем не будут получать никаких обновлений безопасности.

Например, текущая минимальная рекомендуемая версия PHP для запуска WordPress — 7.4, срок эксплуатации которой истек 28 ноября 2022 года.

Само ядро WordPress считается совместимым (с некоторыми явными исключениями) с PHP 8.0 и PHP 8.1 и бета-совместимым с PHP 8.2 и версией PHP 8.3. Однако они не могут гарантировать, что все плагины будут совместимы с текущими или будущими версиями PHP [27].

Поэтому разработчику плагинов важно иметь процесс проверки плагинов на совместимость версий PHP.

Для реализации ПО для автоматизации тестирования бизнес-приложений использованы языки PHP, JavaScript и технология AJAX [13], [18].

Выбор данных средств обусловлен особенностями разработки ПО в CMS WordPress.

Кроме того, сочетание технологий PHP, JavaScript и AJAX может дать несколько преимуществ для веб-разработки:

– улучшенная производительность. AJAX может снизить трафик сервера и повысить скорость работы веб-приложений, позволяя

асинхронно обновлять части веб-страницы без перезагрузки всей страницы;

- улучшение пользовательского опыта. JavaScript обеспечивает интерактивные функции на веб-страницах, а AJAX может улучшить взаимодействие с пользователем, делая веб-приложения более отзывчивыми и быстрыми;
- динамический контент. PHP — это серверный язык сценариев, который может генерировать динамическое содержимое страницы. При использовании с AJAX он позволяет создавать динамические и интерактивные веб-страницы, отдельные части которых можно обновлять без полной перезагрузки страницы;
- асинхронная обработка: AJAX допускает асинхронные вызовы к серверу. Это означает, что пользователь может продолжать взаимодействовать со страницей, пока сервер обрабатывает запросы в фоновом режиме, что приводит к неблокирующему пользовательскому интерфейсу;
- сокращение использования полосы пропускания. Поскольку AJAX отправляет и получает только необходимые данные, он может значительно снизить использование полосы пропускания, что выгодно для пользователей с ограниченными тарифными планами, а также способствует сокращению времени загрузки;
- проверка данных в реальном времени. С помощью JavaScript и AJAX вы можете выполнять проверку форм в реальном времени, обеспечивая немедленную обратную связь с пользователями, что может улучшить общее качество данных и удовлетворенность пользователей.

Таким образом, при совместном использовании эти технологии позволяют создавать мощные и эффективные веб-приложения, обеспечивающие удобство работы пользователей.

Кода автотеста для проверки плагина WordPress представлен в

листинге 1.

Листинг 1 – Кода автотеста для проверки плагина WordPress

```
<?php
«public function post_fetcher() {
$this->posts = get_posts();
}
public function fetch_posts() {
$post_html = '<div class="post">';
foreach ( $this->posts as $post ) {
if ( property_exists( $post, 'post_title' ) ) {
$post_html .= sprintf(
'<h4><a href="%s">%s</a></h4>',
get_permalink( $post->ID ),
$post->post_title

);
}
}
$post_html .= '</div>';
return $post_html;
}
}
add_shortcode( 'wp_learn_php_compatibility',
'wp_learn_php_compatibility_shortcode_render' );
function wp_learn_php_compatibility_shortcode_render() {
$post_fetcher = new post_fetcher();
$post_html = $post_fetcher->fetch_posts();
return $post_html;
```

}» [18].

Фрагмент кода ПО для автоматизации тестирования бизнес-приложений представлен в Приложении А.

3.2 Тестирование программного обеспечения для автоматизации тестирования бизнес-приложений

ПО для автоматизации тестирования бизнес-приложений проверяет текущую версию PHP приложения и после проверяет каждый плагин на его совместимость. И если какой-то плагин не поддерживает определенную версию, он выдает ошибку.

Проверяемый плагин сначала нужно установить в систему. После этого его можно протестировать с помощью разработанного ПО для автоматизации тестирования бизнес-приложений.

Для проведения тестирования разработанного ПО использован метод функционального тестирования [19].

Функциональное тестирование гарантирует, что программное обеспечение соответствует своим требованиям. Это делается путем тестирования каждой отдельной функции программного обеспечения, чтобы убедиться, что оно работает должным образом.

Рассмотрим процесс функционального тестирования ПО для автоматизации тестирования бизнес-приложений на примере тестирования PHP CORE (ядро) для WordPress CRM.

Разработка веб-приложения исключительно на ядре PHP имеет свои преимущества и недостатки [16]:

- это дает разработчику лучший контроль и гибкость;
- в то же время более крупное приложение, разработанное только с использованием ядра PHP, может стать громоздким, сложным управлении и сопровождении.

Плагин WP-CRM — это полнофункциональная CRM WordPress, которая позволяет использовать уже знакомый интерфейс WordPress для простого управления клиентами, проектами и задачами, необходимым для ведения бизнеса.

Выбираем опцию «Плагины-Запустить тесты активных плагинов и тем».

Нажимаем на кнопку «Запустить тесты», как показано на рисунке 12.

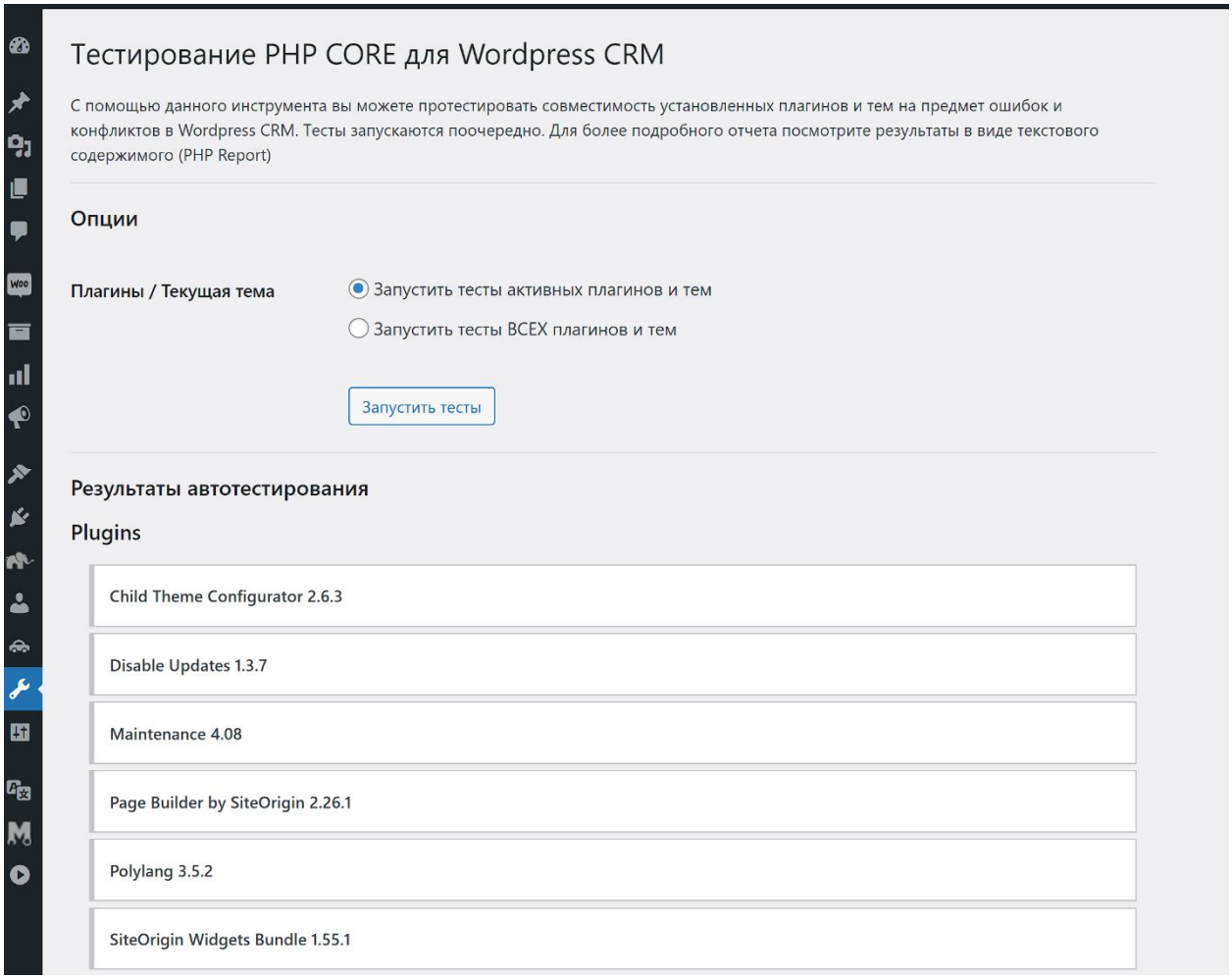

Рисунок 12 – Скриншот окна выполнения автотестов

Устанавливаем флажок «Посмотреть отчет».

Результаты тестирования выводятся в соответствующей области окна, как показано на рисунке 13.

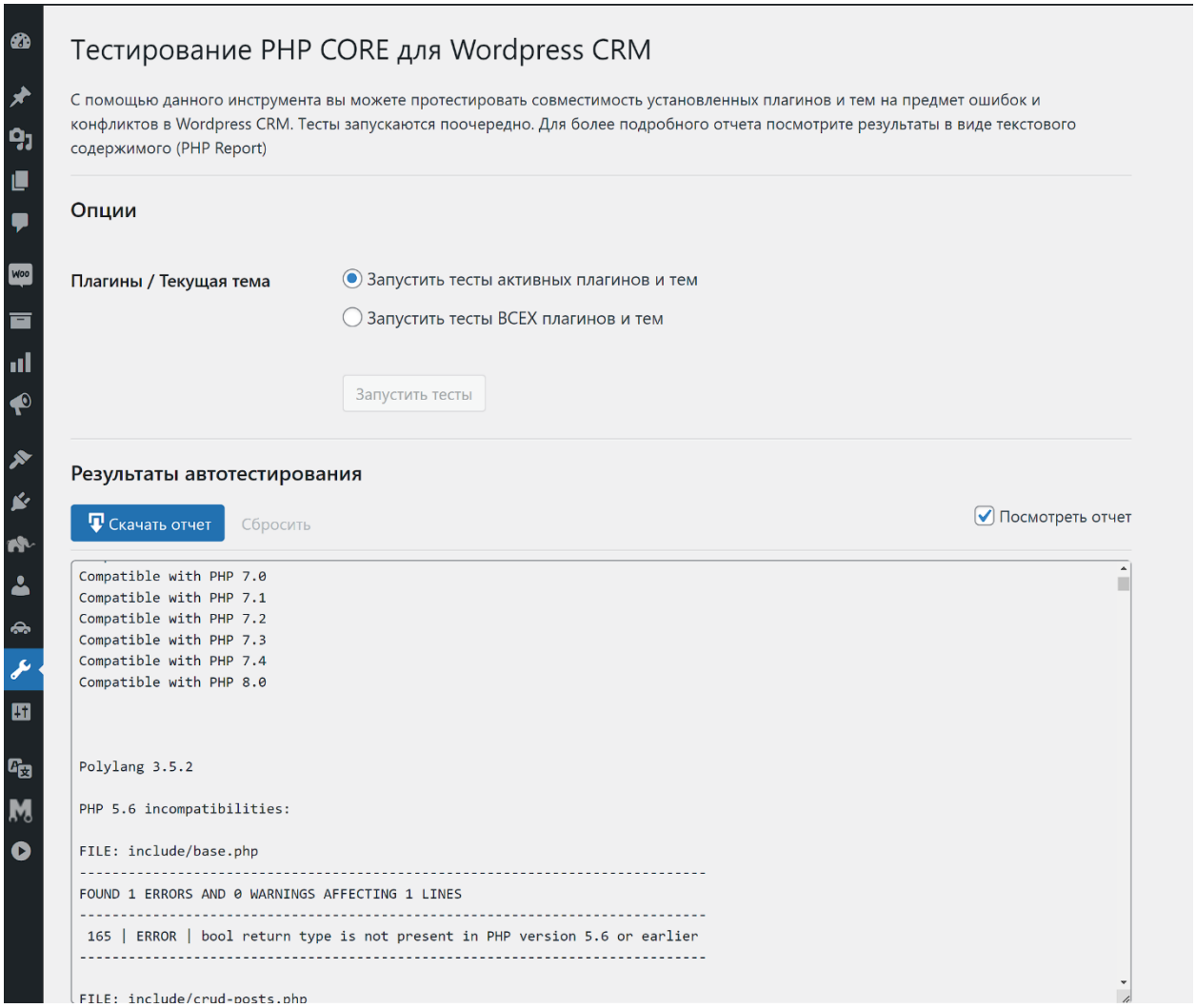

Рисунок 13 – Скриншот просмотра результатов автотестирования

Отчет также можно скачать в виде TXT-файла.

Для этого необходимо установить флажок «Скачать отчет».

Можно просмотреть историю автотестирования различных плагинов, как показано на рисунке 14.

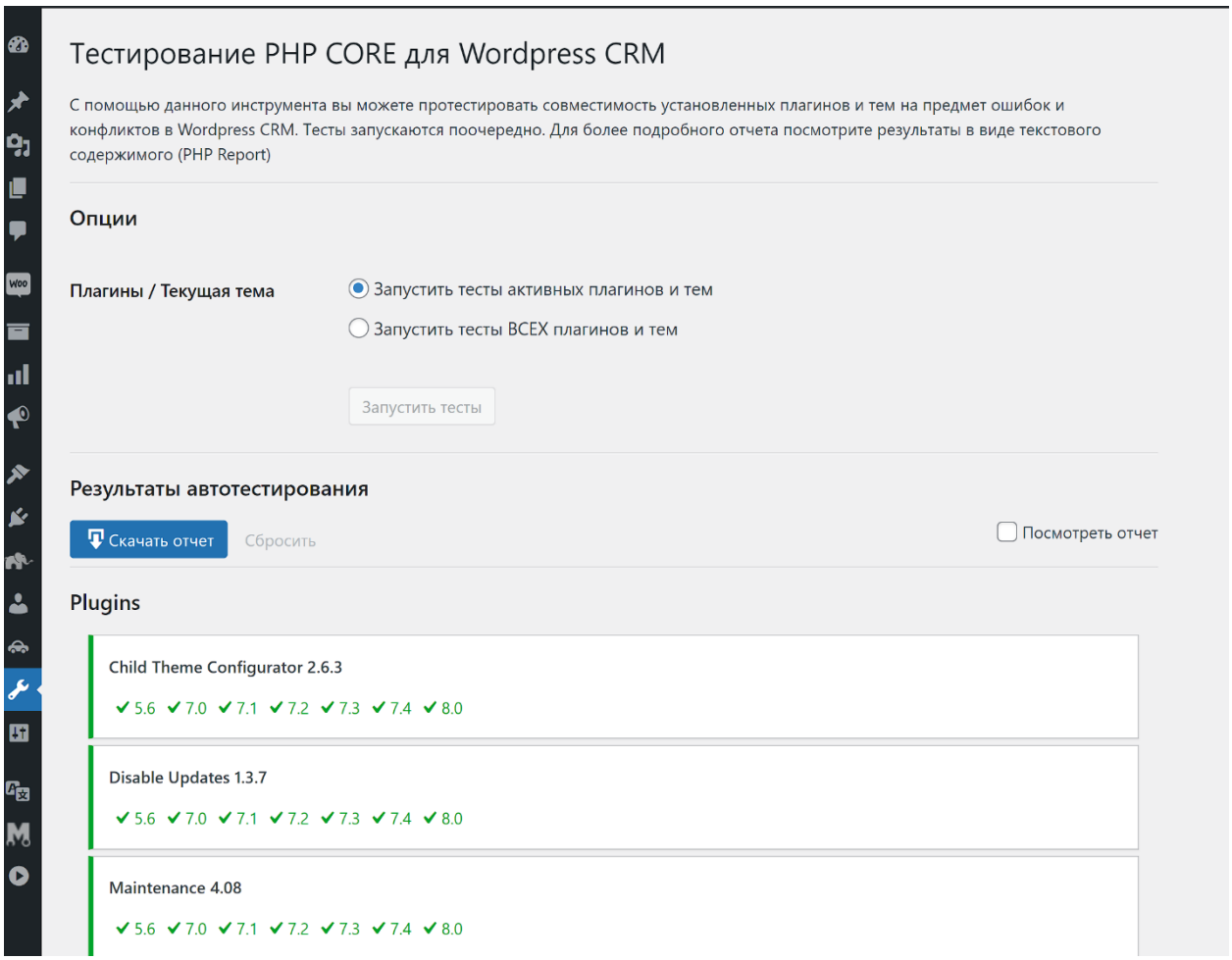

Рисунок 14 – Скриншот просмотра истории автотестирования плагинов

Таким образом тестирование ПО для автоматизации тестирования бизнес-приложений подтвердило его работоспособность.

Выводы по главе 3

Результаты проделанной работы позволили сделать следующие выводы:

– Для реализации ПО для автоматизации тестирования бизнес-приложений использованы язык PHP, JavaScript и технология AJAX. Выбор данных средств обусловлен особенностями разработки ПО в CMS WordPress;
– для проведения тестирования разработанного программного

обеспечения использован метод функционального тестирования продукта;

- функциональное тестирование гарантирует, что программное обеспечение соответствует своим требованиям;
- программа для автоматизации тестирования бизнес-приложений проверяет текущую версию PHP приложения и после проверяет каждый плагин на его совместимость. И если плагин какой-то не поддерживает определенную версию, она выдает ошибку;
- плагин WP-CRM — это полнофункциональная CRM WordPress, которая позволяет использовать уже знакомый интерфейс WordPress для простого управления клиентами, проектами и задачами, необходимым для ведения бизнеса.

Результаты тестирования подтвердили работоспособность разработанного ПО для автоматизации тестирования бизнес-приложений.

Заключение

Бакалаврская работа посвящена актуальной проблеме разработки программного обеспечения для автоматизации тестирования бизнес-приложений.

Актуальность бакалаврской работы обусловлена необходимостью обеспечения высокого качества бизнес-приложений путем их тестирования в процессе проектирования. Одним из наиболее эффективных способов решения данной проблемы является использование специализированной CAT, функциональные возможности которой позволяют тестировать используемые CMS совместно с различными плагинами.

В результате выполнения бакалаврской работы были решены следующие задачи:

- поставлена задача на разработку ПО. Как показал анализ, по своим функциональным и архитектурным особенностям ПО для автоматизации тестирования бизнес-приложений относится к средам автоматизированного тестирования. Главным недостатком, ограничивающим возможности для широкого применения известных аналогов CAT бизнес-приложений является их ориентация на программные платформы конкретных вендоров. Представляет интерес разработка ПО, которое обеспечивает расширение функциональных возможностей CAT. Сформулированы основные требования к разрабатываемому ПО;
- выполнено проектирование ПО. По результатам сравнения достоинств и недостатком рассмотренных методологий выбрана методология RUP. Главным преимуществом методологии RUP для рассматриваемого случая является возможность применения для небольших проектов. Как показал анализ, логическое моделирование в значительной степени связано со структурированием рабочих процессов вокруг хорошо

организованных объектов, представленных графически посредством визуального представления, обеспечиваемого диаграммами UML. Архитектура ПО относится к набору структур, необходимых для построения программной системы. Для разработки модели данных САТ бизнес-приложений выбрано CASE-средство Workbench MySQL, которое поддерживает стандарт IDEF1X. Выбор Workbench обусловлен тем, что в CMS WordPress используется СУБД MySQL;

– реализовано и протестировано ПО. Для реализации ПО для автоматизации тестирования бизнес-приложений использованы язык PHP, JavaScript и технология AJAX. Выбор данных средств обусловлен особенностями разработки ПО в CMS WordPress. Для проведения тестирования разработанного ПО использован метод функционального тестирования продукта.

Результаты тестирования подтвердили работоспособность разработанного ПО для автоматизации тестирования бизнес-приложений.

Список используемой литературы и используемых источников

1. 1С-Битрикс: Управление сайтом. Настройки [Электронный ресурс]. URL: https://dev.1c-bitrix.ru/user_help/settings/index.php (дата обращения: 10.04.2024).

2. Автоматизированное тестирование: что это? Краткое учебное пособие. [Электронный ресурс]. URL: https://logrocon.ru/news/automation_testing (дата обращения: 10.04.2024).

3. Архитектура интерпрайз-приложений может быть другой [Электронный ресурс]. URL: https://habr.com/ru/articles/520858/ (дата обращения: 10.04.2024).

4. Архитектура платформы 1С: Предприятие (версия 8.3.22). Автоматизированное тестирование [Электронный ресурс]. URL: https://v8.1c.ru/platforma/avtomatizirovannoe-testirovanie (дата обращения: 10.04.2024).

5. Виноградова М. В., Белоусова В.И. Унифицированный процесс разработки программного обеспечения : учебное пособие. М: МГТУ им. Н.Э. Баумана, 2015. 82 с. URL: https://www.iprbookshop.ru/134990.html (дата обращения: 10.04.2024).

6. Емельянова Т. В., Кольчатов А.М., Зюзина Н.Ю. Моделирование баз данных : учебное пособие. Саратов : Ай Пи Эр Медиа, 2018. 62 с. URL: https://www.iprbookshop.ru/74560.html (дата обращения: 16.04.2024).

7. Зубкова Т. М. Технология разработки программного обеспечения : учебное пособие. Оренбург : Оренбургский государственный университет, ЭБС АСВ, 2017. 469 с. URL: https://www.iprbookshop.ru/78846.html (дата обращения: 16.04.2024).

8. Коцюба И.Ю., Чунаев А.В., Шиков А.Н. Методы оценки и измерения характеристик информационных систем. Учебное пособие. СПб: Университет ИТМО, 2015. 264 с.

9. Леоненков А. В. Объектно-ориентированный анализ и проектирование с использованием UML и IBM Rational Rose : учебное пособие. М. : ИНТУИТ, Ай Пи Ар Медиа, 2020. 317 с. [Электронный ресурс]. URL: https://www.iprbookshop.ru/97554.html (дата обращения: 10.02.2024).

10. Моделирование данных: обзор Электронный ресурс]. URL: https://habr.com/ru/articles/556790/ дата обращения: 10.04.2024).

11. Подходы к разработке ПО: как правильно выбрать методологию разработки программного обеспечения [Электронный ресурс]. URL: https://issoft.by/blog/podkhody-k-razrabotke-po-kak-pravilno/ (дата обращения: 10.04.2024).

12. Разработка бизнес-приложений [Электронный ресурс]. URL: https://appmaster.io/ru/blog/razrabotka-biznes-prilozhenii (дата обращения: 16.04.2024).

13. Рындин Н. А. Технологии разработки клиентских WEB-приложений на языке JavaScript : учебное пособие. Воронеж : Воронежский государственный технический университет, ЭБС АСВ, 2020. 54 с. URL: https://www.iprbookshop.ru/108188.html (дата обращения: 16.04.2024).

14. Система управления контентом WordPress [Электронный ресурс]. URL: https://wordpress.com/ru/ (дата обращения: 10.04.2024).

15. Современные ERP платформы: Обзор. [Электронный ресурс]. URL: https://www.clouderp.ru/tags/erp_platformy (дата обращения: 10.04.2024).

16. Ступина М. В. Введение в веб-разработку на языке PHP : учебное пособие / Ростов-на-Дону : Донской государственный технический университет, 2022. 95 с. URL: https://www.iprbookshop.ru/130402.html (дата обращения: 16.04.2024).

17. Технические проблемы при установке и использовании плагина WordPress [Электронный ресурс]. URL: https://editor-help.setka.io/hc/ru/ (дата обращения: 10.04.2024).

18. Учебное пособие по PHP Ajax с примером [Электронный ресурс]. URL: https://www.guru99.com/ru/php-ajax.html (дата обращения: 10.04.2024).

19. Функциональное тестирование ПО [Электронный ресурс]. URL: https://gb.ru/blog/funktsionalnoe-testirovanie-po/ (дата обращения: 10.04.2024).

20. Четырнадцать плагинов WordPress CRM, чтобы зарядить ваш бизнес в 2022 году [Электронный ресурс]. URL:https://offlinecrm.ru/14-plaginov-wordpress-crm-chtoby-zaryadit-vash-biznes-v-2022-godu/ (дата обращения: 10.04.2024).

21. Enterprise Application Testing. [Электронный ресурс]. URL: https://www.360logica.com/blog/enterprise-application-testing (дата обращения: 10.04.2024).

22. Hamilton T. Test Environment for Software Testing. [Электронный ресурс]. URL: https://www.guru99.com/test-environment-software-testing.html (дата обращения: 10.04.2024).

23. MySQL Workbench: Visual tool for database architects [Электронный ресурс]. URL: https://www.mysql.com/products/workbench/ (дата обращения: 10.04.2024).

24. Oracle Testing as a Servicehttps [Электронный ресурс]. URL: https://www.oracle.com/technetwork/oem/cloud-mgmt/ds-oracletesting-as-a-service-1905796.pdf (дата обращения: 10.04.2024).

25. Scrum: методология гибкой разработки [Электронный ресурс]. URL: https://gb.ru/blog/skrum/ (дата обращения: 10.04.2024).

26. Software Requirements [Электронный ресурс]. URL: http://beepvolume.com/oop/2020/software-requirements/ (дата обращения: 10.02.2024).

27. Testing your products for PHP version compatibility [Электронный ресурс]. URL: https://learn.wordpress.org/tutorial/testing-your-products-for-php-version-compatibility/ (дата обращения: 10.04.2024).

28. Unified Modeling Language (UML) Diagrams [Электронный ресурс]. URL: https://www.geeksforgeeks.org/unified-modeling-language-uml-introduction/ (дата обращения: 10.04.2024).

29. Waterfall Methodology: A Comprehensive Guide [Электронный ресурс]. URL: https://www.atlassian.com/agile/project-management/waterfall-methodology (дата обращения: 10.04.2024).

30. What is Business Application Software? [Электронный ресурс]. URL: https://www.syntacticsinc.com/news-articles-cat/business-app-software (дата обращения: 10.04.2024).

Приложение А

Фрагмент кода ПО для автоматизации тестирования бизнес-приложений

```
<?php
namespace OptimizedCode;
class CodeChecker {
	private static $singleton = null;
	public static function get() {
		if ( ! self::$singleton ) {
			self::$singleton = new self();
			self::$singleton->setup();
		}
		return self::$singleton;
	}
	public function loadScripts( $page = '' ) {
		if ( 'tools_page_optimized-code' !== $page ) {
			return;
		}
		$assetsFile = dirname( dirname( __FILE__ ) )
 . '/build/scan.asset.php';
		if ( ! file_exists( $assetsFile ) ) {
			add_action(
				'admin_notices',
				function() {
					?>
					<div class="notice notice-error is-dismissible">
						<p><?php echo wp_kses_post( __( 'optimized-code'
) ); ?></p>
					</div>
					<?php
```

```
        }
                        );
                        return;
                }
                $jsAssets = require_once $assetsFile;
                $cssFile = '../build/scan.css';
                $jsFile  = '../build/scan.js';
                wp_enqueue_style(
                        'code-checker',
                        plugins_url( $cssFile, __FILE__ ),
                        array(),
                        $jsAssets['version']
                );
        wp_register_script(
                        'code-checker',
plugins_url( $jsFile, __FILE__ ),
$jsAssets['dependencies'],
                        $jsAssets['version'],
                        true );
                wp_localize_script(
                        'code-checker',
                        'checkerItems',
                        array(
                                'plugins' => $this->getPlugins(),
                                'themes'  => $this->getThemes(),
                        )
                );
                wp_enqueue_script( 'code-checker' );
        }
        public function setup() {
add_action( 'admin_menu', array( $this, 'createMenu' ) );
        add_action( 'admin_enqueue_scripts', array( $this, 'loadScripts' ) );
        }
```

```
        public function getPlugins() {
                require_once ABSPATH . 'wp-admin/includes/plugin.php';
                $plugins = get_plugins();
                $excludedPlugins = apply_filters( 'phpcompat_excluded_plugins', array( 'Code Checker', 'Hello Dolly' ) );
                $plugins = array_filter( $plugins, function ( $pluginData ) use ( $excludedPlugins ) {
                                return ! in_array( $pluginData['Name'], $excludedPlugins, true );
                        }
                );
                if ( empty( $plugins ) ) {
                        return array();
                }
                $activePlugins = get_option( 'active_plugins' );
                if ( is_multisite() ) {
                   $activePlugins = array_merge( $activePlugins, array_keys( (array) get_site_option( 'active_sitewide_plugins', array() ) ) );
                }
        }
}
?>
```