МИНИСТЕРСТВО ЦИФРОВОГО РАЗВИТИЯ, СВЯЗИИ МАССОВЫХ КОММУНИКАЦИЙ РОССИЙСКОЙ ФЕДЕРАЦИИ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ "САНКТ-ПЕТЕРБУРГСКИЙ

ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ ТЕЛЕКОММУНИКАЦИЙ ИМ. ПРОФ. М. А. БОНЧ-БРУЕВИЧА"

Факультет инфокоммуникационных сетей и систем

Кафедра сетей связи и передачи данных

ЛАБОРАТОРНАЯ РАБОТА №2

по дисциплине «Процессы жизненного цикла программного обеспечения»

Тестирование программного обеспечения

Выполнил:

студент 4 курса

ИКПИ-14

Хохлов Тихон Владимирович

Санкт-Петербург

2025

**Цель работы**

Изучить классификацию видов тестирования и применить её для анализа выбранного программного проекта. В ходе работы необходимо определить применимые уровни, типы, методы и виды тестирования, выявить характерные дефекты и подготовить отчет, который поможет оценить качество проекта и спланировать процесс тестирования.

Критерии к тестируемому проекту взяты из ГОСТ 19.101-77 («Общие требования к программной документации»), ГОСТ 34.602-89 («Техническое задание на создание автоматизированной системы»), ГОСТ 19.104-78 («Основные надписи»), ГОСТ 19.105-78 («Общие требования к построению, изложению и оформлению документов»), ГОСТ 34.603-92 («Испытания и приемка автоматизированных систем»).

**Описание программного проекта**

Техническое задание студенческого проекта «КОМПЬЮТЕРНАЯ ИГРА «БОЯРСКИЙ ТУРНИР»» по дисциплине «Процессы жизненного цикла программного обеспечения» студентов группы ИКПИ-11 Крылова А.В., Дунаева В.Е., ИКПИ-14 Хохлова Т.В.

Проект представляет многопользовательскую игру в жанре RTS. Основная цель приложения — заполнить нишу на рынке мобильных игр после ухода Supercell из России, предоставив альтернативу Clash Royale, используя знакомые элементы геймплея и тематику.

В качестве ключевых функций можно выделить следующие:

1. Сражения (Бой 1 на 1)

2. Сбор эликсира

3. Призыв юнитов

4. Использование специальных способностей

5. Цикличная смена карт

6. Переопределение колод юнитов

Программа разработана на языке Rust в интегрированной среде разработки NeoVIM в операционной системе Arch linux. Также использовались следующие технологии и библиотеки:

* Bevy - это набор библиотек для разработки игр на языке Rust, включает в себя такие модули, как ECS, рендер-граф на основе WebGPU, инджектор зависимостей, сцены, ассет-менеджер, ввод/вывод и прочее

**Описание элементов классификации, применимых к программному проекту**

**Уровни тестирования:**

* Компонентное - тестирование программы на уровне отдельно взятых модулей, функций или классов. Цель – выявление локализованных в модуле ошибок в реализации алгоритмов, а также определение степени готовности системы
* Системное тестирование проверяет как функциональные, так и нефункциональные требования к системе в целом.

**Типы тестирования:**

* Функциональный тип (что система делает) описывает такие подвиды как :
  + 1. функциональное тестирование;
    2. тестирование возможности взаимодействия (включает тестирование совместимости и интеграционное тестирование);
    3. тестирование безопасности.
* Нефункциональный тип (как система это делает) описывает такие подвиды как :
  + 1. нагрузочное тестирование;
    2. стресс-тестирование;
    3. тестирование удобства пользования;
    4. тестирование надежности и т. д.

**Методы тестирования:**

* Статический метод : тестирует такие объекты, как программный код, документацию, ресурсы проекта, все то, что может быть проверено без непосредственного выполнения программного кода, т. е. запуска программной системы
* Динамический метод : тестирует функциональность программной системы и режимы ее эксплуатаци

**Виды тестирования:**

* По объекту
  + Функциональное : Проверяется соответствие требованиям. Оценка производится в соответствии с ожидаемыми и полученными результатами
  + Юзабилити : Тестирование удобства пользования приложением определяет, соответствует ли приложение потребностям целевой аудитории и отвечает ли оно требованиям пользователя
  + UI/GUI : Тестирование на соответствие стандартам графических интерфейсов, тестирование с различными разрешениями экрана, тестирование в ограниченных условиях, и т.д.
  + Совместимость : Проверяется функционирование на различных операционных системах, платформах разработки, в различных браузерах.
  + Документация : Тестируется на этапе разработки требований к программному продукту после создания функциональных спецификаций. Помогает избежать логических дефектов и ненужных изменений в продукте до начала его фактической разработки.
* По степени автоматизации
  + Ручное тестирование : это процесс поиска ошибок в программе без использования специальных ПО, силами человека. Тестировщик имитирует реальные действия пользователя и старается охватить максимум функций продукта и найти ошибки
* По признаку позитивности
  + Позитивное тестирование : это тестирование на данных или сценариях, которые соответствуют нормальному поведению системы.
  + Негативное тестирование : это тестирование на данных или сценариях, которые соответствуют нештатному поведению тестируемой системы – различные сообщения об ошибках, исключительные ситуации, запредельные состояния
* По времени проведения
  + Приемочное тестирование : формальный процесс тестирования, который проверяет соответствие системы требованиям и проводится с целью определения, удовлетворяет ли система приемочным критериям; вынесения решения заказчиком или другим уполномоченным лицом о приемке приложения.
  + Регрессионное тестирование : это вид тестирования, направленный на проверку изменений, сделанных в приложении или окружающей среде, для подтверждения того факта, что существующая ранее функциональность работает, как и прежде

**Примеры дефектов, характерных для каждого вида тестирования.**

**Уровни тестирования:**

* **Компонентное тестирование**:
  + Тестирование отдельных модулей игры, таких как:
    - Модуль сбора ресурсов
    - Модуль призыва юнитов.
    - Модуль сражений.
  + Пример дефекта: Некорректное отображение количества ресурсов после их сбора.
* **Интеграционное тестирование**:
  + Тестирование взаимодействия между модулями, например:
    - Взаимодействие модуля сбора ресурсов и модуля призыва юнитов.
    - Взаимодействие модуля сражений и модуля защиты башен.
  + Пример дефекта: Юниты не появляются на поле боя после призыва, если ресурсы не были корректно собраны.
* **Системное тестирование**:
  + Тестирование игры в целом, включая:
    - Проверка всех функций игры в комплексе.
    - Проверка работы игры на разных платформах (мобильные устройства, ПК).
  + Пример дефекта: Игра зависает при одновременном использовании нескольких функций.

**Типы тестирования:**

* **Функциональное тестирование:**
  + Проверка выполнения функций игры, таких как:
    - Сбор ресурсов.
    - Призыв юнитов.
    - Сражения.
  + Пример дефекта: Функция сбора ресурсов не работает, если игрок одновременно пытается призвать юнита.
* **Нефункциональное тестирование:**
  + Проверка характеристик, не связанных с функциональностью, таких как:
    - Производительность (например, время загрузки игрового мира).
    - Надежность (например, устойчивость к сбоям).
    - Юзабилити (удобство интерфейса).
  + Пример дефекта: Игра тормозит при большом количестве юнитов на поле боя.

**Методы тестирования:**

* **Статическое тестирование:**
  + Проверка документации и кода без запуска программы, например:
    - Анализ технического задания на соответствие требованиям.
    - Проверка исходного кода на наличие ошибок.
  + Пример дефекта: В ТЗ отсутствует описание некоторых функций игры.
* **Динамическое тестирование:**
  + Тестирование игры в процессе выполнения, например:
    - Проверка работы функций в реальном времени.
    - Проверка взаимодействия игроков в многопользовательском режиме.
  + Пример дефекта: Игра вылетает при одновременном подключении большого числа игроков.

**Виды тестирования:**

* По объекту
  + **Нагрузочное тестирование:**
* Проверка производительности игры под нагрузкой, например:
  + Одновременное подключение большого числа игроков.
  + Использование большого количества юнитов на поле боя.
* Пример дефекта: Игра зависает при одновременном подключении 1000 игроков.
  + **Юзабилити-тестирование:**
* Проверка удобства интерфейса, например:
  + Удобство управления юнитами.
  + Удобство навигации по меню.
* Пример дефекта: Игроки не могут быстро найти кнопку для призыва юнитов.
  + **Тестирование безопасности:**
* Проверка защиты данных игроков, например:
  + Защита от взлома аккаунтов.
  + Защита от читерства.
* Пример дефекта: Данные игроков могут быть украдены из-за уязвимости в системе авторизации.
  + **Регрессионное тестирование:**
* Проверка, что новые изменения не сломали существующие функции, например:
  + После добавления новой карты проверить, что старые функции работают корректно.
* Пример дефекта: после обновления игры перестала работать функция сбора ресурсов.
  + **Документация :**
    - * Проверить документацию на полноту и непротиворечивость

**Выводы по работе**

В ходе работы изучена классификация видов тестирования и применена к анализу приложения многопользовательской игры в жанре RTS. Определены применимые уровни, типы и методы тестирования, такие как компонентное, системное, функциональное и нефункциональное тестирование, а также методы белого и черного ящика. Выявлены примеры дефектов.

Работа подтвердила важность комплексного подхода к тестированию для повышения качества программного продукта. Полученные навыки помогут в дальнейшей работе с контролем качества ПО.

**Список использованных источников**

1. Пачин А.В. Лаборатоный практикум по дисциплине “Процессы жизненного цикла программного обеспечения”
2. Смирнов К.А Лекции по дисциплине “Процессы жизненного цикла программного обеспечения”
3. Отчет по курсовой работе по предмету «Разработка Java-приложений управления телекоммуникациями» студента группы ИКПИ-11 Крылова А.В.