

Е. Д. Востокова¹✉, Н. С. Головачев¹

Разработка подсистемы управления сценариями и процессами вовлечения пользователей в игровом приложении «BlushBound»

¹Сибирский государственный университет геосистем и технологий,
г. Новосибирск, Российская Федерация
e-mail: liza.vost@mail.ru

Аннотация. В статье представлены результаты разработки подсистемы управления сценариями и процессами вовлечения пользователей в игровом приложении «BlushBound». Выполнен анализ предметной области современных игровых систем с процедурной генерацией уровней, вероятностными механизмами получения наград и интеграцией интерактивного повествования. Разработка выполнена на платформе Unity с использованием языка программирования C#. Спроектирована модульная архитектура, обеспечивающая централизованное управление состояниями игры, генерацию уровней, поведение персонажей и противников, а также динамическую активацию сценарных событий. Проведено тестирование и выполнено экономическое обоснование проекта.

Ключевые слова: игровое приложение, Unity, C#, подсистема управления сценариями, процедурная генерация, вовлечение пользователей

E. D. Vostokova¹✉, N. S. Golovachev¹

Development of a scenario and user engagement management subsystem in the BlushBound game application

¹Siberian State University of Geosystems and Technologies, Novosibirsk, Russian Federation
e-mail: liza.vost@mail.ru

Abstract. The article presents the development results of a scenario and user engagement management subsystem for the BlushBound game application. The subject area of modern game systems with procedural level generation, probabilistic reward mechanics, and interactive storytelling integration is analyzed. The subsystem was implemented using the Unity platform and C#. A modular architecture was designed to provide centralized game state management, level generation, character and enemy control, and dynamic story activation. Testing and economic feasibility analysis were conducted.

Keywords: game application, Unity, C#, scenario management subsystem, procedural generation, user engagement

Введение

Современная игровая индустрия характеризуется высокой конкуренцией и быстрым развитием технологий. Пользователь ожидает не только качественную игровую механику, но и продуманную систему прогрессии, нарративное сопровождение и устойчивую мотивацию к продолжению игры.

Игровые проекты все чаще объединяют процедурную генерацию уровней, систему случайных наград и элементы интерактивного повествования. Однако в

ряде решений данные компоненты реализуются отдельно, что снижает уровень вовлеченности и удержания аудитории.

Актуальность работы обусловлена необходимостью создания подсистемы, объединяющей генерацию уровней, систему наград и сценарные события в единую архитектуру.

Цель работы – разработка подсистемы управления сценариями и процессами вовлечения пользователей в игровом приложении «BlushBound».

Исследование предметной области

Процедурная генерация уровней позволяет формировать уникальные игровые пространства при каждом прохождении. Алгоритмы генерации основаны на заданных правилах построения локаций и размещения объектов [1]. Данный подход повышает реиграбельность и снижает затраты на создание статического контента.

Вовлечение пользователей обеспечивается сочетанием прогрессии, системы наград и эмоционального контента [2]. Вероятностная модель получения персонажей стимулирует повторные игровые сессии, создавая эффект ожидания результата. При этом важен баланс вероятностей и постепенное усложнение игрового процесса.

Интеграция нарратива в игровой цикл усиливает эмоциональную связь пользователя с продуктом. Согласно подходу MDA (Mechanics–Dynamics–Aesthetics), игровые механики должны быть согласованы с динамикой и эмоциональным восприятием [3].

Анализ существующих решений показал, что редко реализуется централизованное управление всеми перечисленными механизмами. В проекте «BlushBound» предложена архитектура, объединяющая данные элементы.

Методы и технологии

Разработка выполнена в среде Unity с использованием языка C# [1, 4]. Архитектура построена по модульному принципу с применением принципов объектно-ориентированного проектирования и паттернов программирования [5, 6].

Основные модули системы:

- менеджер состояния игры;
- модуль генерации уровней;
- модуль управления персонажами;
- модуль искусственного интеллекта противников;
- модуль сценарного управления;
- система наград.

Хранение конфигурационных данных реализовано с помощью ScriptableObject [1], что позволяет отделить параметры игровых объектов от логики выполнения.

Поведение персонажа построено на основе конечного автомата состояний: ожидание, перемещение, атака, применение способности, получение урона.

Противники реализуют алгоритм обнаружения игрока по радиусу и переходят в режим преследования и атаки. Для боссов предусмотрена фазовая модель поведения: при снижении здоровья ниже порогового значения изменяются параметры атаки и скорости.

Сценарные события представлены структурированными объектами, содержащими текст, условия активации и ссылки на последующие события. Активация происходит через систему триггеров, связанных с прогрессом пользователя.

Архитектура и программные компоненты подсистемы

Для обеспечения масштабируемости, удобства тестирования и дальнейшего расширения функционала подсистема управления сценариями и процессами вовлечения реализована на основе модульной архитектуры. Взаимодействие между компонентами построено по событийно-управляемой модели, что минимизирует жесткие зависимости и упрощает отладку игровых процессов.

В состав подсистемы входят ключевые программные компоненты.

Менеджер состояния игры (GameManager), отвечающий за инициализацию сцен, загрузку уровней, сохранение и загрузку прогресса, а также синхронизацию работы всех модулей. Обеспечивает корректный переход между игровыми состояниями и обрабатывает глобальные события.

Модуль процедурной генерации уровней (LevelGenerator), формирующий структуру локации на основе заданных параметров (биом, текущая сложность, этап сюжета), размещает противников, интерактивные объекты и триггерные зоны. Генерация ограничивается правилами проходимости и сбалансированности, что исключает формирование непроходимых или избыточно сложных конфигураций.

Модуль управления персонажами и ИИ противников реализующий логику поведения на основе конечных автоматов состояний. Для игрового персонажа обрабатываются ввод пользователя, перемещение, расчет зон поражения и активация способностей. Для противников реализованы алгоритмы обнаружения цели, преследования, атаки и перезарядки. Боссовые сущности поддерживают фазовую модель поведения: при снижении здоровья ниже порогового значения изменяются параметры урона, скорости и набор доступных атак.

Сценарный движок (ScenarioEngine), хранящий структуру диалогов, событий и условий их активации. Интегрируется с игровым циклом через систему триггеров: сюжетные фрагменты запускаются при завершении уровня, победе над ключевыми противниками или получении персонажей определенной редкости. Обеспечивает приостановку игрового процесса на время отображения диалоговых окон и сохранение ветвления истории.

Система наград и прогрессии (RewardSystem), управляющий выдачей внутриигровых ресурсов, рассчитывает вероятности получения персонажей разной редкости, реализует механизмы гарантированного дропа (pity system) и отслеживает выполнение условий разблокировки новых локаций и контента.

Конфигурационные параметры игровых объектов (характеристики, редкость, условия активации сценариев, диапазоны генерации) отделены от программной

логики и хранятся в структуре ScriptableObject платформы Unity. Такой подход позволяет настраивать баланс, добавлять новый контент и корректировать сценарные ветвления без изменения исходного кода, что существенно ускоряет итерации разработки.

Взаимосвязь основных компонентов подсистемы и потоки данных между ними представлены на структурной схеме (рис. 1).

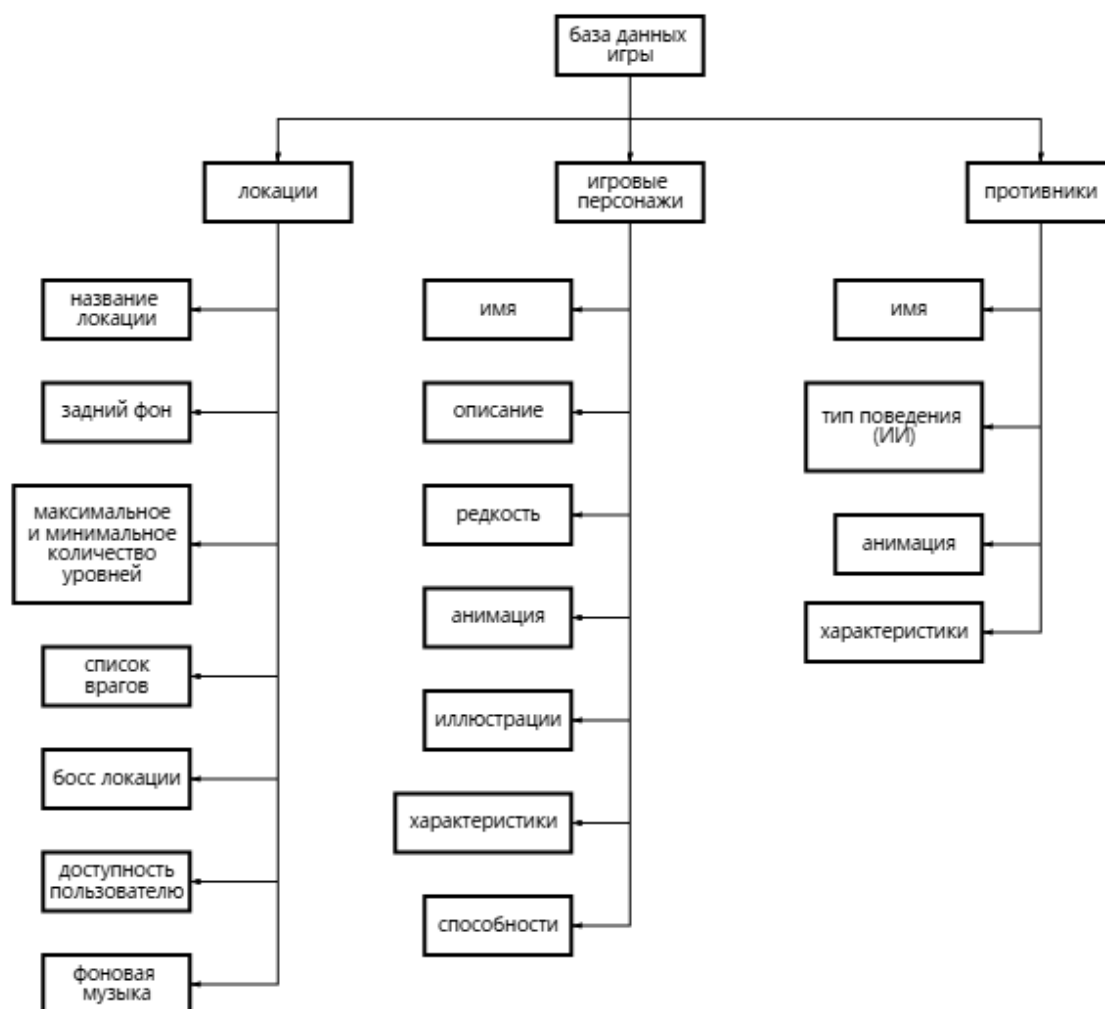


Рис. 1. Структура хранения компонентов подсистемы управления сценариями

Предложенная архитектура обеспечивает высокую степень связности игровых механик и нарративных элементов, позволяя динамически адаптировать игровой процесс под текущий прогресс пользователя и поддерживать устойчивый цикл вовлечения на протяжении всего жизненного цикла приложения.

Результаты

В ходе разработки реализованы следующие функциональные возможности:

- при загрузке сцены формируется структура комнат и размещаются противники в допустимых координатах;

- игрок выбирает персонажей различной редкости, влияющей на характеристики и стиль прохождения;
- боевая система с расчетом урона, осуществляющимся с учетом характеристик атакующего и защитных параметров цели;
- активные способности с механизмом перезарядки;
- поведение босса локации изменяется в зависимости от текущего уровня здоровья.
- сюжетные фрагменты активируются при прохождении уровней, победе над боссами и получении редких персонажей.

Проведено функциональное тестирование, включающее проверку корректности расчета урона, переходов между сценами и поведения противников. Критических ошибок не выявлено. Производительность соответствует требованиям платформы Unity [1].

Экономическое обоснование выполнено для базового сценария: 20 000 установок, конверсия 3,5 %, ARPPU 800 рублей и рекламный доход 25 рублей на пользователя. Совокупная выручка составляет 1 060 000 рублей при затратах 997 600 рублей. Точка окупаемости достигается при 18 823 установках.

Заключение

Разработана подсистема управления сценариями и процессами вовлечения пользователей для игрового приложения «BlushBound». Реализована модульная архитектура, обеспечивающая интеграцию процедурной генерации уровней, системы наград и сценарных механизмов.

Тестирование подтвердило стабильность функционирования системы. Экономические расчеты демонстрируют потенциальную самоокупаемость проекта в течение первого года эксплуатации.

Предложенные решения могут быть использованы при разработке аналогичных игровых приложений.

БИБЛИОГРАФИЧЕСКИЙ СПИСОК

1. Unity Technologies. Unity User Manual 2022.3 (LTS). – San Francisco : Unity Technologies, 2023.
2. Adams E. Fundamentals of Game Design. – 3rd ed. – New Riders, 2014.
3. Hunicke R., LeBlanc M., Zubek R. MDA: A Formal Approach to Game Design and Game Research // Proceedings of the AAAI Workshop on Challenges in Game AI. – 2004.
4. Рабинович С. Разработка 2D-игр на Unity. – Санкт-Петербург : Питер, 2022. – 448 с.
5. Ларошель Д. Паттерны проектирования игр на C#. – Москва : БХВ-Петербург, 2020. – 496 с.
6. Фримен Э., Робсон Э. Паттерны архитектуры программных систем. – Санкт-Петербург : Питер, 2019. – 656 с.

© Е. Д. Востокова, Н. С. Головачев, 2026