ORES
Научно-практический электронный журнал
Оригинальные исследования (ОРИС), *ISSN 2222-0402*
www.ores.su
original-research.ru

УДК 004.42

ПРОЕКТИРОВАНИЕ ПРОГРАММНОЙ ПЛАТФОРМЫ ДЛЯ БЫСТРОГО ПРОТОТИПИРОВАНИЯ ИГРОВЫХ МЕХАНИК В ИНТЕРАКТИВНЫХ КАТАЛОГАХ И ИНСТАЛЛЯЦИЯХ

Попов Максим Алексеевич,
Федеральное государственное бюджетное образовательное учреждение высшего образования «МИРЭА – Российский технологический университет» (Программная инженерия), Москва, Россия
Студент кафедры игровой индустрии
E-mail: holtes10@gmail.com

Аннотация

В статье рассматривается проектирование программной платформы для быстрого прототипирования игровых механик, предназначенных для использования в интерактивных каталогах и инсталляциях. Платформа обеспечивает загрузку пользовательских ассетов (изображений, видео, аудио, 3D-моделей), их автоматическое сопоставление с параметризуемыми пресетами игровых механик, автоматическую сборку и публикацию WebGL-приложений. Проведен анализ существующих аналогов (PandaSuite, Intuiface, Verge3D, PlayCanvas, GDevelop), сформулированы функциональные и нефункциональные требования. Описаны два варианта архитектуры системы, построены функциональные модели в нотации IDEF0 и модели взаимодействия в нотации C4. Представлена логическая модель базы данных, обоснован выбор технологического стека (Unity/C# для клиента, Python для сервера, OpenAI GPT для интеллектуального анализа контента). Результаты работы могут быть использованы при разработке инструментов для автоматизации создания интерактивного контента.

Ключевые слова: игровые механики, интерактивные инсталляции, WebGL, прототипирование, Unity, low-code, no-code, пресеты механик, большие языковые модели, интерактивные каталоги.

DESIGNING A SOFTWARE PLATFORM FOR RAPID PROTOTYPING OF GAME MECHANICS IN INTERACTIVE CATALOGS AND INSTALLATIONS

Popov Maksim Alekseevich,
MIREA – Russian Technological University (Software engineering), Moscow, Russia
Student, Department of Game industry
E-mail: holtes10@gmail.com

ABSTRACT

The article considers the design of a software platform for rapid prototyping of game mechanics intended for use in interactive catalogs and installations. The platform provides uploading of user assets (images, video, audio, 3D models), their automatic mapping to parameterized game mechanics presets, automated building and publishing of WebGL applications. An analysis of existing analogs (PandaSuite, Intuiface, Verge3D, PlayCanvas, GDevelop) was conducted, and functional and non-functional requirements were formulated. Two architectural variants are described, functional models in IDEF0 notation and interaction models in C4 notation are presented. A logical database model is described, and the choice of the technology stack (Unity/C# for the client, Python for the server, OpenAI GPT for intelligent content analysis) is justified. The results can be used in developing tools for automating the creation of interactive content.

Keywords: game mechanics, interactive installations, WebGL, prototyping, Unity, low-code, no-code, mechanics presets, large language models, interactive catalogs.

АКТУАЛЬНОСТЬ

Современные интерактивные каталоги и инсталляции активно используются в музеях, на выставках, в коммерческих презентационных пространствах и образовательных учреждениях. Одним из ключевых элементов таких систем являются игровые механики, обеспечивающие вовлечение пользователя и эффективное взаимодействие с представленным контентом. Однако процесс создания интерактивных механик для подобных систем остается трудоемким и требует значительных временных и финансовых затрат.

Традиционный подход к разработке предполагает, что каждая интерактивная механика создается отдельно и интегрируется в программное обеспечение конкретного стенда. Это приводит к длительным циклам разработки, высокой стоимости адаптации и ограничению возможностей быстрого обновления контента. Существующие на рынке решения, такие как PandaSuite [1], Intuiface [2], Verge3D [3], PlayCanvas [4] и GDevelop [5], ориентированы на создание интерактивных презентаций, веб-игр или 3D-визуализаций, но ни одно из них не предоставляет комплексного инструмента для быстрого прототипирования игровых механик на основе пользовательских ассетов с их последующим развертыванием в интерактивных инсталляциях посредством WebGL-технологий [6, 13]. Кроме того, ни один из аналогов не предоставляет возможности автоматического маппинга ассетов пользователя к механикам в зависимости от контекста, что ускоряет разработку в десятки раз.

Актуальность данной работы определяется необходимостью создания программной платформы, позволяющей существенно сократить время разработки интерактивных игровых механик, обеспечить их независимость от конкретного аппаратного обеспечения стендов и предоставить low-code/no-code инструменты [14, 15] для пользователей без глубоких технических знаний.

ЦЕЛЬ ИССЛЕДОВАНИЯ

Целью исследования является проектирование программной платформы для быстрого прототипирования игровых механик, предназначенных для использования в интерактивных каталогах и инсталляциях. Платформа должна обеспечивать загрузку пользовательских ассетов, их автоматическое сопоставление с параметризуемыми пресетами игровых механик, автоматическую сборку, оптимизацию и публикацию WebGL-приложений.

Для достижения поставленной цели были сформулированы следующие задачи: провести анализ существующих аналогов и выявить их преимущества и недостатки; сформулировать функциональные и нефункциональные требования к платформе; разработать архитектуру системы с описанием вариантов алгоритмов работы; построить функциональные модели в нотации IDEF0 [7] и модели взаимодействия в нотации C4 [8]; спроектировать логическую модель базы данных; обосновать выбор технологического стека.

МАТЕРИАЛЫ И МЕТОДЫ ИССЛЕДОВАНИЯ

В ходе исследования были использованы следующие методы: сравнительный анализ существующих программных решений для создания интерактивного контента; метод функционального моделирования IDEF0 для описания алгоритмов работы платформы; метод архитектурного моделирования C4 для описания взаимодействия компонентов системы; метод проектирования реляционных баз данных с использованием ER-диаграмм [9].

В качестве объектов сравнительного анализа были рассмотрены пять платформ: PandaSuite, Intuiface, Verge3D, PlayCanvas и GDevelop. Анализ проводился по критериям поддержки WebGL, необходимости программирования, стоимости и основной области применения.

Проектирование архитектуры платформы выполнялось с учетом требований к масштабируемости, производительности и удобству использования. Клиентская часть проектировалась на базе движка Unity [10] с языком C#, серверная часть – на базе Python. Для интеллектуального анализа контента и автоматизации маппинга ассетов была предусмотрена интеграция генеративной языковой модели (LLM) [11].

РЕЗУЛЬТАТЫ И ИХ ОБСУЖДЕНИЕ

Анализ существующих аналогов. Проведенный анализ пяти платформ-аналогов показал, что ни одна из них не предоставляет комплексного решения для быстрого прототипирования игровых механик с интеграцией в интерактивные инсталляции. PandaSuite и Intuiface ориентированы на создание интерактивных презентаций и инсталляций, но имеют ограниченные возможности для реализации игровых механик. Verge3D обеспечивает создание 3D-визуализаций, однако требует владения инструментами моделирования и не содержит готовых шаблонов. PlayCanvas представляет собой мощный облачный движок, но требует программирования на JavaScript. GDevelop позволяет создавать игры без программирования, однако не предназначен для интеграции в системы интерактивных инсталляций.

Таблица 1 – Сравнение аналогов

Решение	Поддержка WebGL	Необходимость программирования	Стоимость	Основная область применения
PandaSuite	Да	Нет	Высокая	Интерактивные презентации
Intuiface	Да	Нет	Средняя	Интерактивные инсталляции
Verge3D	Да	Частично	Низкая	3D-визуализации
PlayCanvas	Да	Да	Средняя	Веб-игры
GDevelop	Да	Нет	Бесплатно	2D-игры

Функциональные и нефункциональные требования. На основе анализа аналогов были сформулированы требования к платформе. К основным функциональным требованиям относятся: поддержка загрузки и валидации мультимедийных ассетов (PNG,

JPEG, MP4, WAV, glTF/GLB, FBX, OBJ); наличие библиотеки параметризуемых пресетов игровых механик; автоматическое сопоставление ассетов с элементами пресетов; автоматическая сборка, оптимизация и публикация WebGL-билдов; интеграция LLM для ускорения маппинга и генерации метаданных; предоставление API и вебхуков для взаимодействия с системами управления стендами; поддержка оффлайн-развертывания.

Нефункциональные требования включают обеспечение целевого FPS не менее 30 для оптимизированных билдов, кроссбраузерную совместимость (Chromium, Firefox, Safari), аутентификацию и разграничение прав доступа, а также интуитивный интерфейс с пошаговыми сценариями для пользователей разного уровня подготовки.

Архитектура платформы. Были рассмотрены два варианта архитектуры. Первый вариант предполагает централизованную обработку на сервере: клиент загружает ассеты и конфигурацию, после чего сервер выполняет валидацию, оптимизацию, сборку Unity → экспорт WebGL → развертывание и возвращает URL или пакет для встраивания. Второй вариант реализует гибридный подход, при котором клиент выполняет предварительную оптимизацию и локальную эмуляцию, а сервер остается точкой авторитета для финальных билдов.

Для описания алгоритмов работы платформы были построены функциональные модели в нотации IDEF0 (Рисунки 1-7), включающие контекстную диаграмму и декомпозиции подпроцессов: анализ входных данных, формирование конфигурации проекта, генерация WebGL-сборки. Для второго варианта декомпозиция блока анализа входных данных отражает распределение операций между клиентом и сервером.

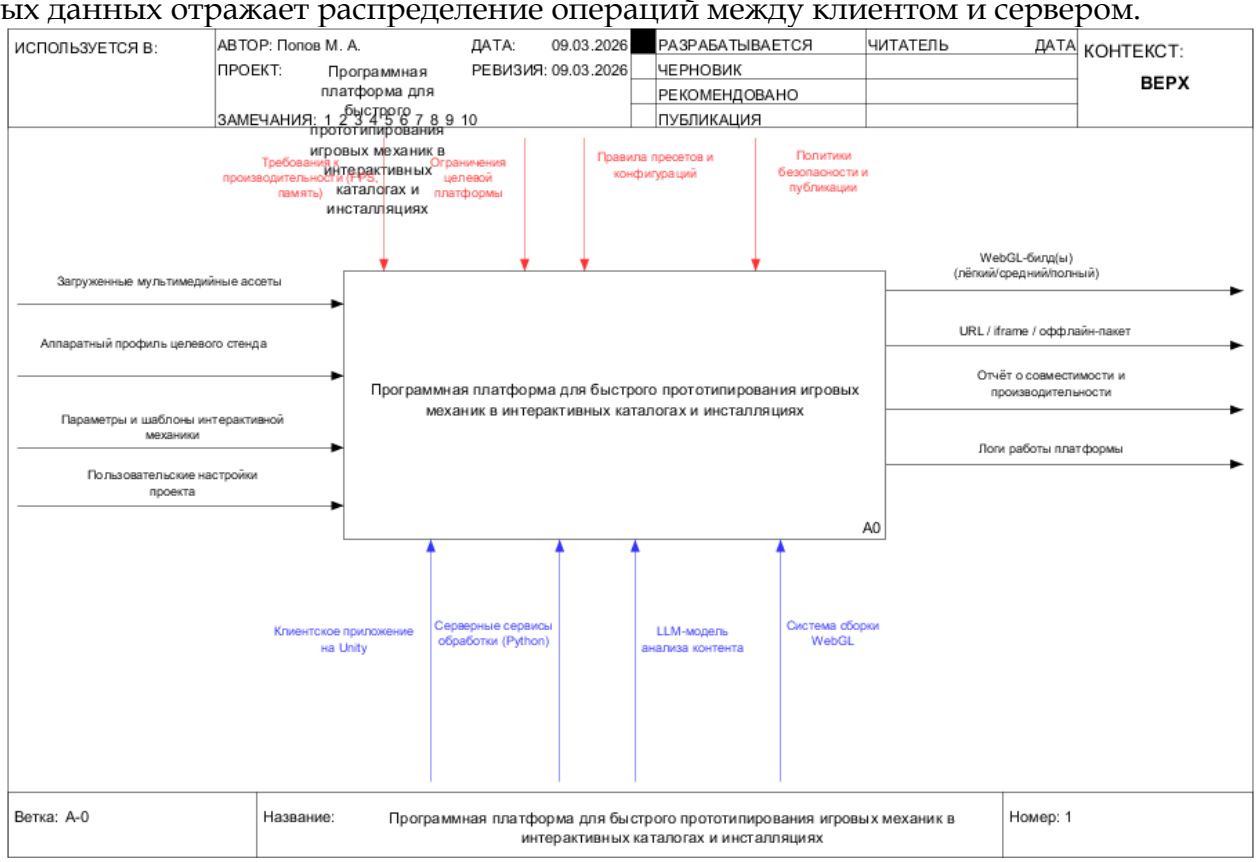

Рисунок 1 – Контекстная диаграмма первого варианта работы программного модуля платформы [разработано автором]

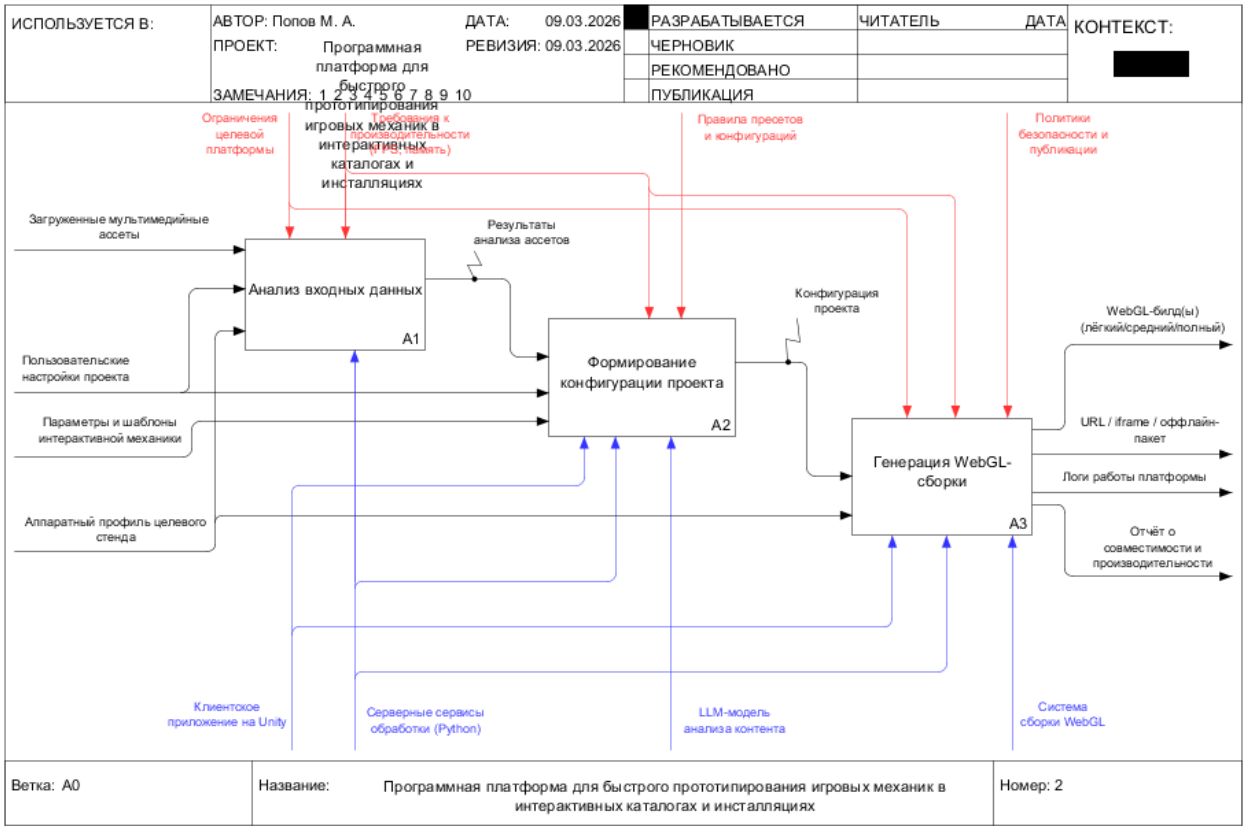

Рисунок 2 – Декомпозиция контекстной диаграммы первого варианта работы программного модуля платформы [разработано автором]

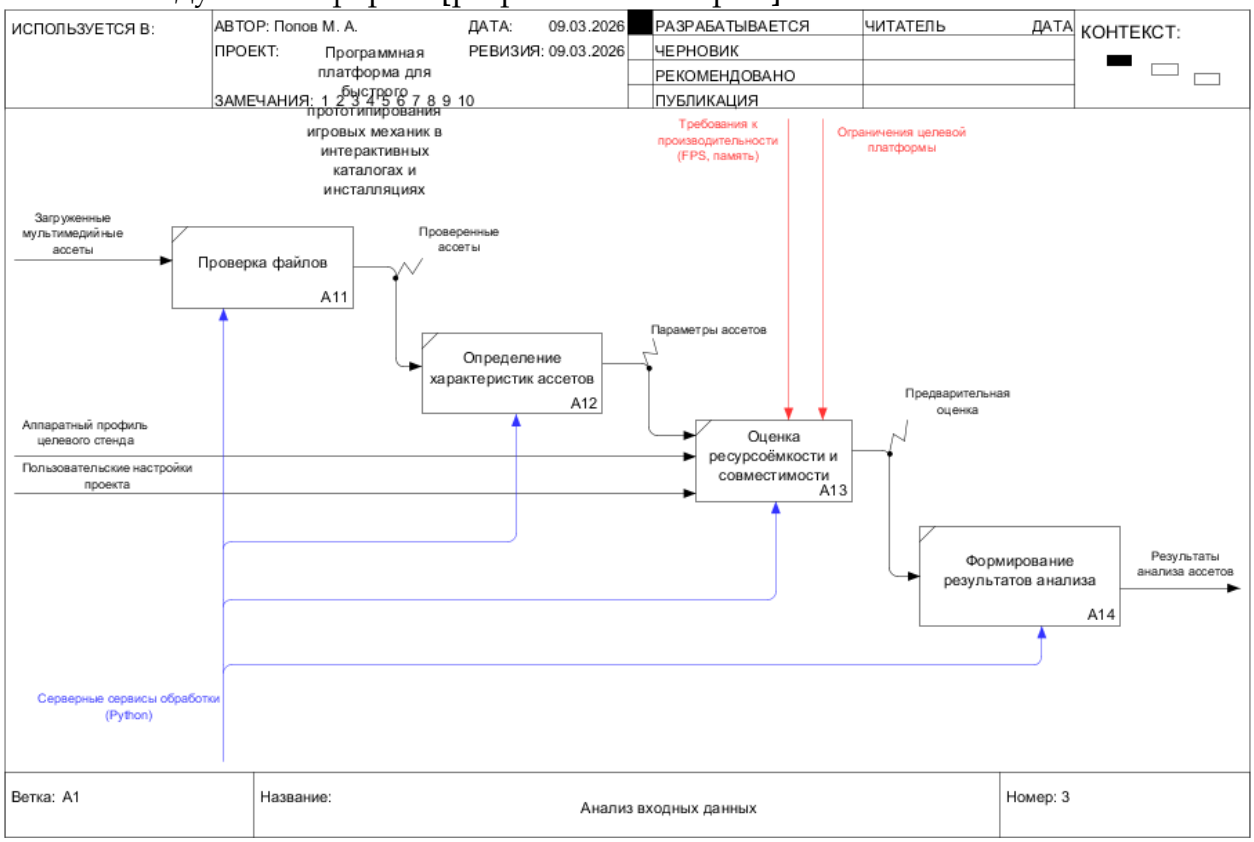

Рисунок 3 – Декомпозиция подпроцесса «Анализ входных данных» первого варианта работы программного модуля платформы [разработано автором]

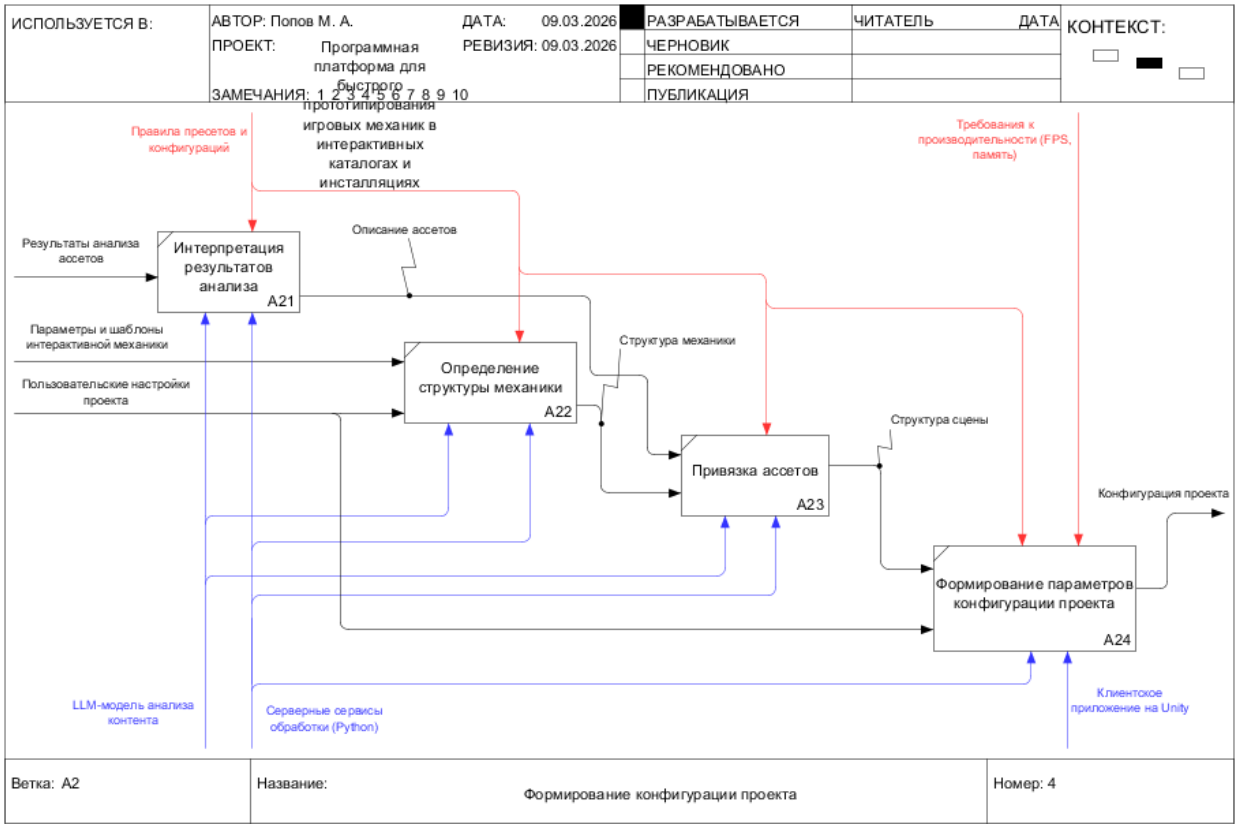

Рисунок 4 – Декомпозиция подпроцесса «Формирование конфигурации проекта» программного модуля платформы [разработано автором]

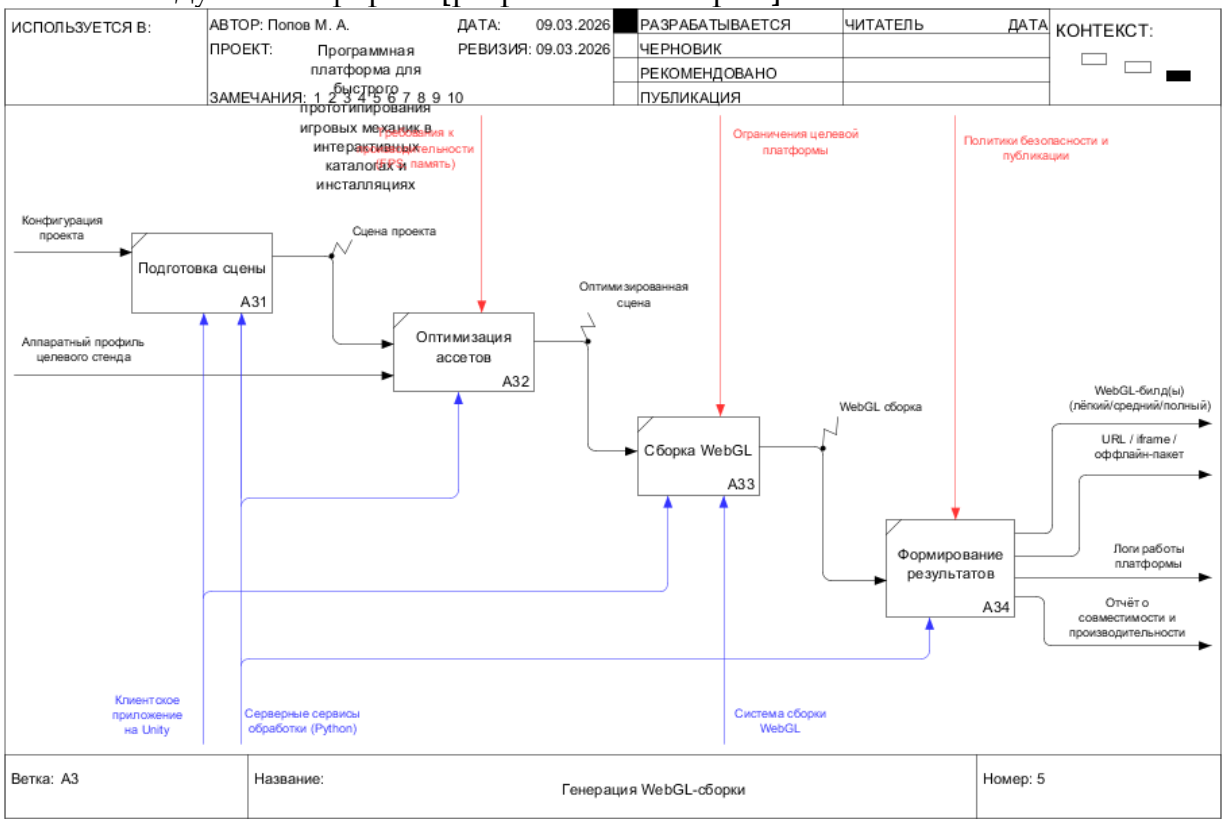

Рисунок 5 – Декомпозиция подпроцесса «Генерация WebGL-сборки» программного модуля платформы [разработано автором]

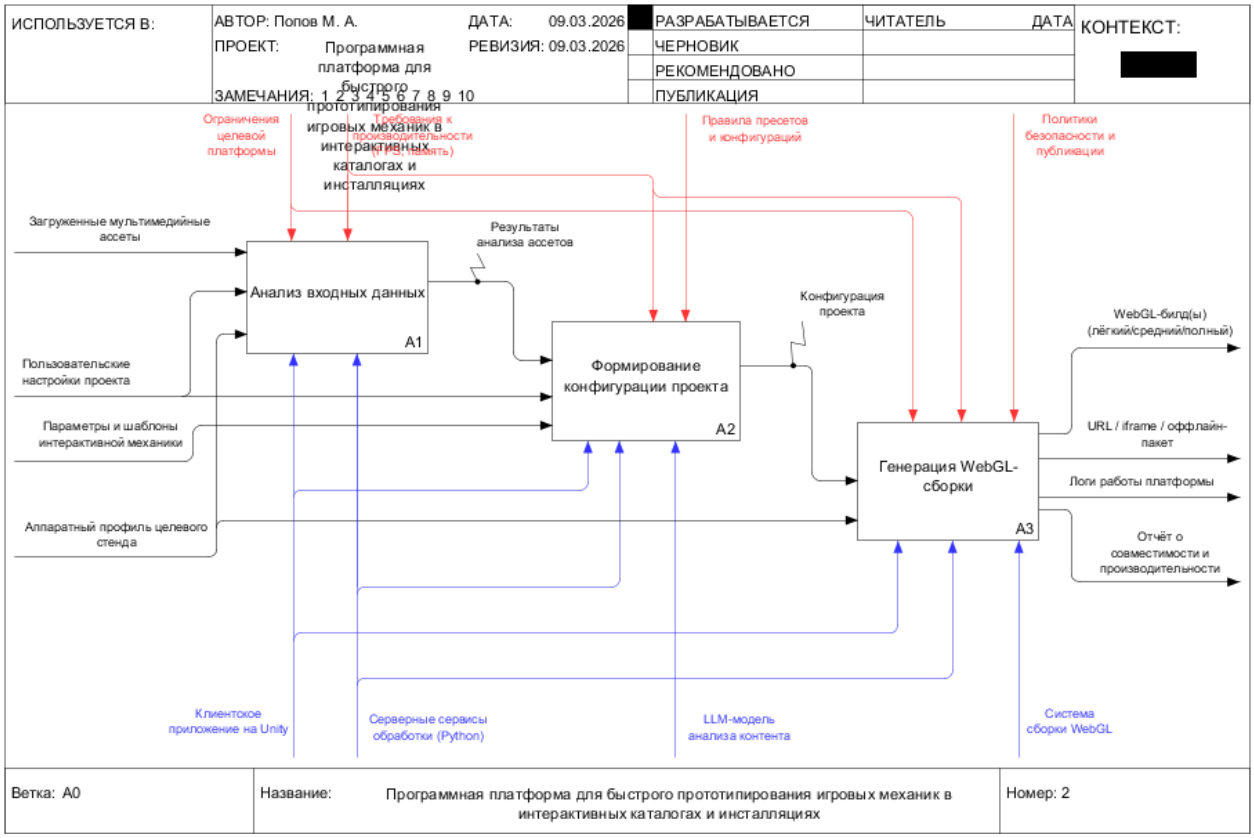

Рисунок 6 – Декомпозиция контекстной диаграммы второго варианта работы программного модуля платформы [разработано автором]

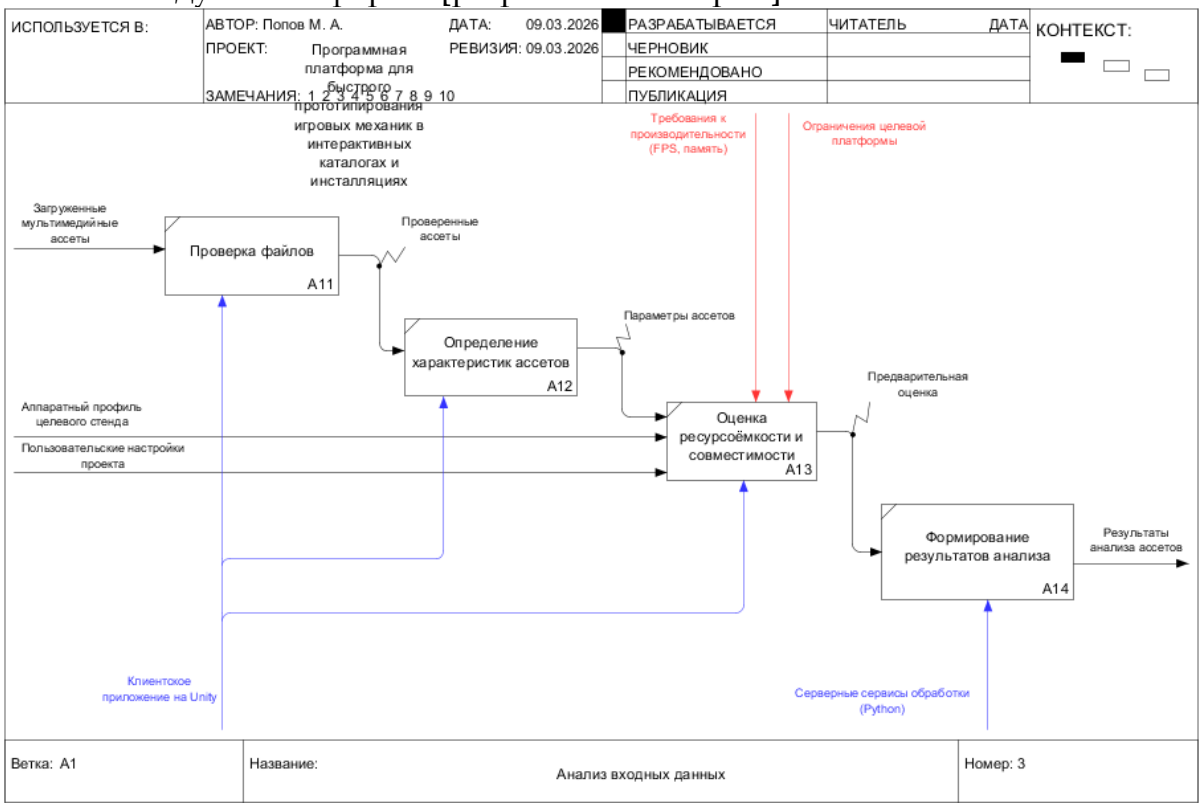

Рисунок 7 – Декомпозиция подпроцесса «Анализ входных данных» второго варианта работы программного модуля платформы [разработано автором]

Модели взаимодействия в нотации C4. На диаграмме контекста (Рисунок 8) определены основные взаимодействия платформы с пользователями (администраторы, редакторы) и внешними системами (веб-браузер, системы управления стендами, сервис хостинга, LLM). Диаграмма контейнеров (Рисунок 9) детализирует внутреннюю структуру

платформы: клиентское приложение Unity, API-сервер на Python, модули анализа ассетов, генерации конфигурации, сборки WebGL и публикации.

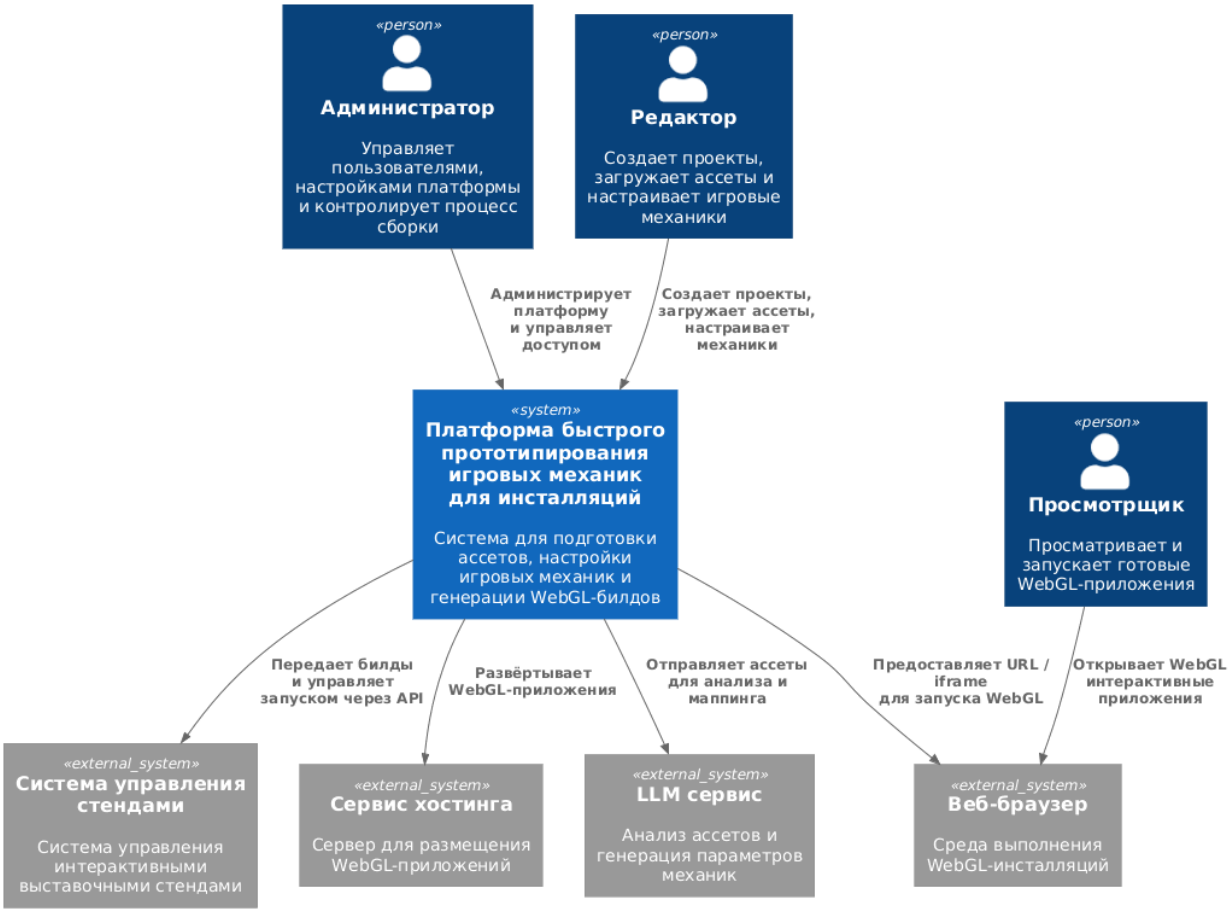

Рисунок 8 – Диаграмма контекста программной платформы [разработано автором]

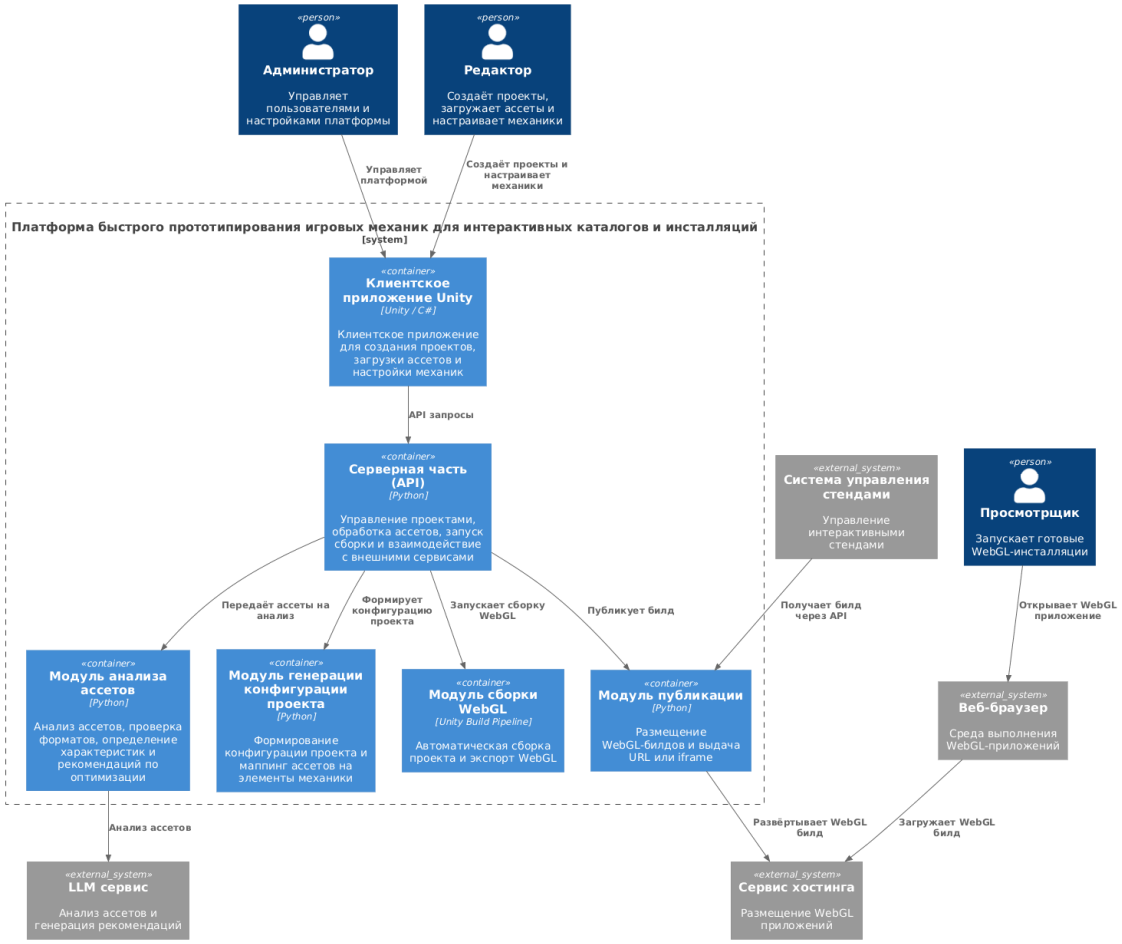

Рисунок 9 – Диаграмма контейнеров для разрабатываемой платформы [разработано автором]

Логическая модель базы данных. Спроектирована реляционная база данных, нормализованная до третьей нормальной формы (Рисунок 10). Основные сущности: Users (пользователи с ролями admin/editor), Projects (проекты с привязкой к аппаратным профилям), Hardware_profiles (профили целевых стендов), Presets (пресеты игровых механик), Assets (метаданные ассетов), Configurations (конфигурации проектов), Build_jobs (задания сборки), Build_artifacts (результаты сборки), llm_requests (журнал обращений к LLM), webhooks (внешние интеграции). Для хранения гибких структур используются поля формата JSON, бинарные данные хранятся во внешнем файловом хранилище.

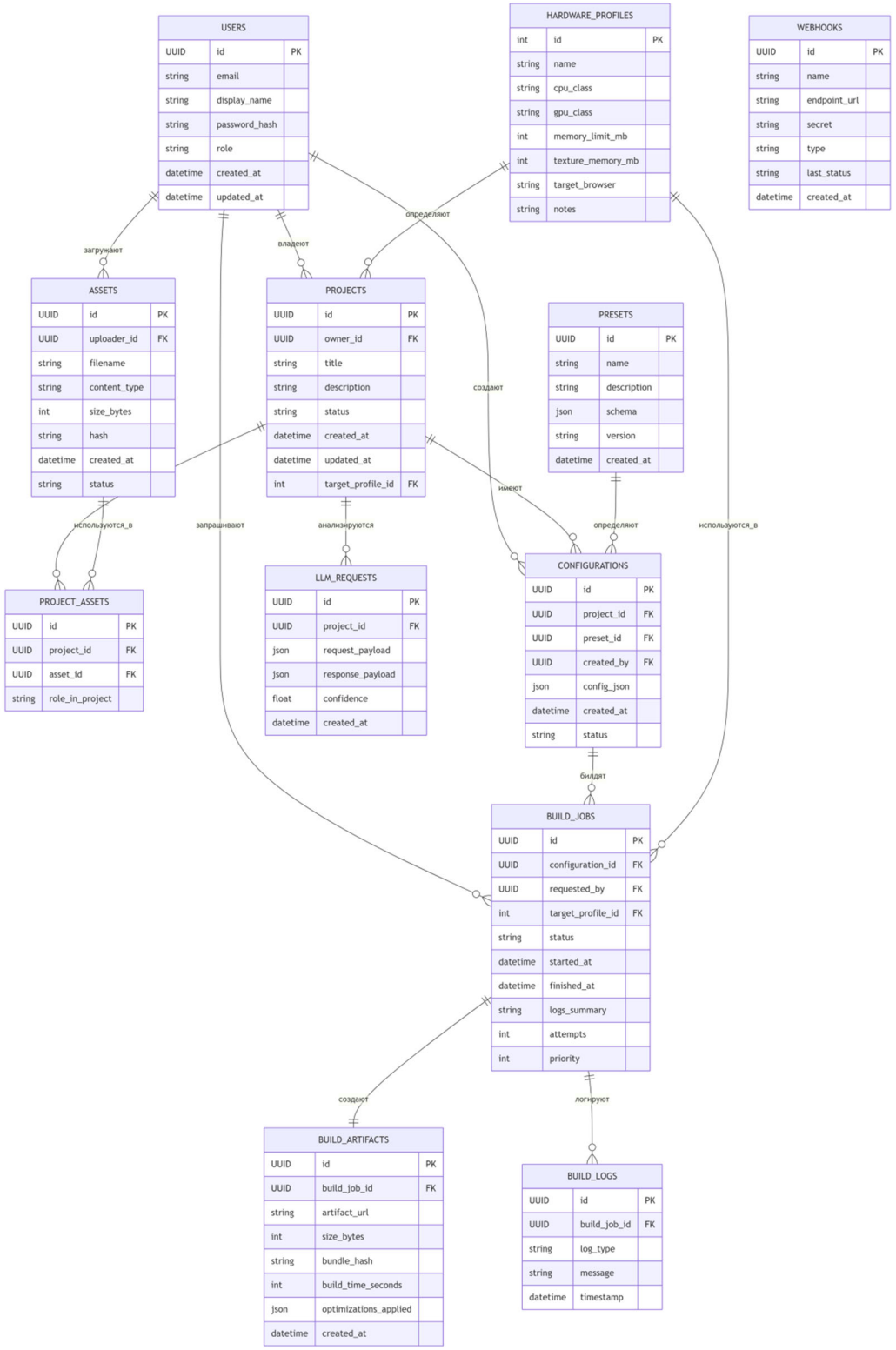

Рисунок 10 – ER-диаграмма платформы [разработано автором]

Выбор технологического стека. Для клиентской части выбран движок Unity с языком C#, обеспечивающий визуальный интерфейс для загрузки и настройки ассетов, предварительное тестирование механик и кроссплатформенную совместимость. Серверная часть реализуется на Python, что позволяет использовать обширную экосистему библиотек для обработки мультимедиа (Pillow, OpenCV, pygltflib) и интеграции LLM. В качестве генеративной модели выбрана OpenAI GPT [12], обеспечивающая интеллектуальный анализ загруженных ассетов и автоматическое сопоставление с параметрами пресетов. WebGL выбран в качестве формата сборки как стандарт для запуска интерактивного 3D-контента в браузере.

ВЫВОДЫ

В результате проведенного исследования спроектирована программная платформа для быстрого прототипирования игровых механик, предназначенных для интерактивных каталогов и инсталляций. Проведенный анализ пяти существующих аналогов подтвердил отсутствие на рынке комплексного решения, позволяющего создавать игровые механики на основе пользовательских ассетов и развертывать их в интерактивных инсталляциях посредством WebGL-технологий.

Сформулированы функциональные и нефункциональные требования к платформе, охватывающие загрузку и валидацию ассетов, автоматическое сопоставление с пресетами механик, оптимизацию и публикацию WebGL-билдов, интеграцию LLM, а также требования к производительности и безопасности.

Разработаны два варианта архитектуры системы: централизованный серверный pipeline и гибридный подход с локальной предобработкой. Построены функциональные модели в нотации IDEF0 и модели взаимодействия в нотации C4. Спроектирована логическая модель реляционной базы данных. Обоснован выбор технологического стека: Unity/C# для клиентской части, Python для серверной части, OpenAI GPT для интеллектуального анализа контента.

Платформа позволит существенно сократить время создания и адаптации игровых механик, обеспечивая совместимость с оборудованием различных конфигураций и предоставляя удобный low-code/no-code инструментарий для пользователей без глубоких технических знаний.

Список литературы:

1. PandaSuite [Электронный ресурс]. — 2026. — URL: https://pandasuite.com/ (дата обращения: 03.04.2026).
2. Intuiface [Электронный ресурс]. — 2026. — URL: https://www.intuiface.com/ (дата обращения: 03.04.2026).
3. Verge3D [Электронный ресурс]. — 2026. — URL: https://www.soft8soft.com/verge3d/ (дата обращения: 03.04.2026).
4. PlayCanvas [Электронный ресурс]. — 2026. — URL: https://playcanvas.com/ (дата обращения: 03.04.2026).
5. GDevelop [Электронный ресурс]. — 2026. — URL: https://gdevelop.io/ru-ru (дата обращения: 03.04.2026).
6. WebGL [Электронный ресурс]. — 2026. — URL: https://get.webgl.org/ (дата обращения: 04.04.2026).
7. IDEF0. Знакомство с нотацией и пример использования [Электронный ресурс]. — URL: https://trinion.org/blog/idef0-znakomstvo-s-notaciey-i-primer-ispolzovaniya (дата обращения: 04.04.2026).
8. C4 model for software architecture [Электронный ресурс]. — 2025. — URL: https://c4model.info/ (дата обращения: 04.04.2026).
9. What is an entity relationship diagram? [Электронный ресурс]. — 2025. — URL: https://www.ibm.com/think/topics/entity-relationship-diagram (дата обращения: 04.04.2026).
10. Unity [Электронный ресурс]. — 2026. — URL: https://unity.com/ru (дата обращения: 05.04.2026).

11. LLM [Электронный ресурс]. — 2026. — URL: https://aws.amazon.com/ru/what-is/large-language-model/ (дата обращения: 05.04.2026).
12. OpenAI GPT [Электронный ресурс]. — 2026. — URL: https://openai.com/index/openai-api/ (дата обращения: 05.04.2026).
13. Evans A., Romeo M., Bahrehmand A. [et al.] 3D graphics on the web: A survey // Computers & Graphics. — 2014. — Vol. 41. — P. 43–61. — DOI: 10.1016/j.cag.2014.02.002.
14. Sufi F. Algorithms in Low-Code-No-Code for Research Applications: A Practical Review // Algorithms. — 2023. — Vol. 16, No. 2. — P. 108. — DOI: 10.3390/a16020108.
15. Alamin M. A. A., Uddin G., Malakar S. [et al.] Developer discussion topics on the adoption and barriers of low code software development platforms // Empirical Software Engineering. — 2023. — Vol. 28, No. 1. — P. 4. — DOI: 10.1007/s10664-022-10244-0.

References:

1. PandaSuite [Online]. — 2026. — URL: https://pandasuite.com/ (accessed: 03.04.2026).
2. Intuiface [Online]. — 2026. — URL: https://www.intuiface.com/ (accessed: 03.04.2026).
3. Verge3D [Online]. — 2026. — URL: https://www.soft8soft.com/verge3d/ (accessed: 03.04.2026).
4. PlayCanvas [Online]. — 2026. — URL: https://playcanvas.com/ (accessed: 03.04.2026).
5. GDevelop [Online]. — 2026. — URL: https://gdevelop.io/ru-ru (accessed: 03.04.2026).
6. WebGL [Online]. — 2026. — URL: https://get.webgl.org/ (accessed: 04.04.2026).
7. IDEF0. Introduction to the notation and example of use [Online]. — URL: https://trinion.org/blog/idef0-znakomstvo-s-notaciey-i-primer-ispolzovaniya (accessed: 04.04.2026).
8. C4 model for software architecture [Online]. — 2025. — URL: https://c4model.info/ (accessed: 04.04.2026).
9. What is an entity relationship diagram? [Online]. — 2025. — URL: https://www.ibm.com/think/topics/entity-relationship-diagram (accessed: 04.04.2026).
10. Unity [Online]. — 2026. — URL: https://unity.com/ru (accessed: 05.04.2026).
11. LLM [Online]. — 2026. — URL: https://aws.amazon.com/ru/what-is/large-language-model/ (accessed: 05.04.2026).
12. OpenAI GPT [Online]. — 2026. — URL: https://openai.com/index/openai-api/ (accessed: 05.04.2026).
13. Evans A., Romeo M., Bahrehmand A. [et al.] 3D graphics on the web: A survey // Computers & Graphics. — 2014. — Vol. 41. — P. 43–61. — DOI: 10.1016/j.cag.2014.02.002.
14. Sufi F. Algorithms in Low-Code-No-Code for Research Applications: A Practical Review // Algorithms. — 2023. — Vol. 16, No. 2. — P. 108. — DOI: 10.3390/a16020108.
15. Alamin M. A. A., Uddin G., Malakar S. [et al.] Developer discussion topics on the adoption and barriers of low code software development platforms // Empirical Software Engineering. — 2023. — Vol. 28, No. 1. — P. 4. — DOI: 10.1007/s10664-022-10244-0.