УДК 004.4'2

ИССЛЕДОВАНИЕ И РАЗРАБОТКА МЕТОДА ПО АВТОМАТИЗАЦИИ СОЗДАНИЯ СИСТЕМ ПОДДЕРЖКИ МОДИФИКАЦИЙ ДЛЯ ВИДЕОИГР

Тугушев Д.Р. (Университет ИТМО)
Научный руководитель – преподаватель Школы разработки видеоигр Берзон Д.И.
(Университет ИТМО)

Введение. Пользовательский контент, создаваемый игроками, а особенно модификации к видеоиграм, пользуются большой популярностью и актуальны как для игроков, которые и создают их, тем самым выражая свой интерес к игре, так и для остальных игроков, которые продолжают взаимодействовать с игровым продуктом и дальше. Но больше всего выгоды получают сами разработчики видеоигры, включая: продленный жизненный цикл продукта и повышенный интерес игроков [1], обратная связь в виде идей по улучшению игры и даже их реализации, а также формирование сообщества специалистов, откуда можно набирать персонал для дальнейшей разработки уже не модификаций, а самой исходной игры [2].

Однако в случае, если разработчики предоставят возможность создавать модификации (или не будут юридически преследовать игроков согласно EULA), потребуются значительные ресурсы на разработку инструмента по запуску сторонних модификаций во время игрового процесса, что отсрочит выход продукта на рынок. Если же этого не произойдет, сами игроки будут проводить обратную разработку игры и изменять её. Но это потребует огромных, никем не оплачиваемых усилий [3], которые будут утрачены при обновлении игры, которая будет защищена снова. А взаимодействие разработчиков модификаций будет затруднено отсутствием единого сообщества со схожими взглядами на процессы модификации игры [4].

Таким образом, достаточно актуальной является проблема высокой сложности создания систем поддержки модификаций для видеоигр. Наличие метода для автоматизированной генерации таких систем для разных видеоигр позволило бы затрачивать меньше ресурсов на их разработку, тестирование и поддержку, при этом не допуская как юридических нарушений, так и трения между игроками как членами сообществ, ведь первичным ресурсом для модификации окажется, фактически, игра и её разработчики, а не сторонние наработки.

Основная часть. Для того, чтобы игра взаимодействовала с модификациями к ней, необходимо наличие: интерфейса для обращения к игровым сущностям (для добавления новых их типов, изменения существующих либо создания совершенно новых) и системы событий, позволяющей отслеживать происходящее в игре и реагировать на это необходимым модификации образом. Важно отметить, что подход к изменению игры, предотвращающий конфликты с использованием лишь одной модификации, напрямую меняющей игру, можно сравнить с системами плагинов любого другого ПО, работающих с функциями обратного вызова для реализации шаблона “наблюдатель”. Также, для поддержки модификаций чаще всего используют скриптовые языки, которые удобны своей кросс-платформенностью и низким порогом вхождения, делая разработку модификаций более доступной для игроков.

Однако, полная автоматизация генерации кода с помощью LLM невозможна, поскольку результаты генерации обычно обладают низким качеством и требуют доработки, что сводит на нет пользу от их использования. Частичная же автоматизация достигается выделением методов и оценки влияния их изменения (с представлением кода в виде графа, с помощью AST [5]) совместно с разработчиком, позволяя ему полностью контролировать процесс создания системы для конкретной игры. Это достигается пересечением двух методологий разработки: предмет-ориентированной (DDD) для моделирования предметной области, и модель-управляемой (MDE) для автоматизированного преобразования требований к системе в исходный код с помощью абстрактных моделей (предметной области) [6].

Сама же система, которая позволит функционировать всем генерируемым системам поддержки модификаций для разных видеоигр, построена на предложенной исследователями

архитектуре RAMP [7], предоставляющей одну и ту же систему для использования другими. Она реализована на одном из скриптовых языков для обеспечения интероперабельности с играми, модификациями и их поддержкой. Управление событиями основано на реактивности, как варианта эволюции шаблонов обмена сообщениями, с применением шаблона DCRC [8].

Выводы. Проведен сравнительный анализ существующих систем для поддержки в создании модификаций для видеоигр, а также способов их внедрения, в особенности вариантов управления событиями для исполнения стороннего кода во время работы самой игры. Исследованы возможности по автоматизации генерации подобных систем на основе анализа исходного кода видеоигры, подразумевающей активное привлечение разработчиков. Сформирована архитектура предлагаемого решения на основе изученных способов модификации видеоигр, а также возможности автоматизации их создания с учетом выделенных требований к системам поддержки модификаций.

В результате, разработан метод по автоматизации создания систем поддержки модификаций для различных видеоигр, что означает интероперабельность метода и его независимость от технологий (языка программирования) для реализации видеоигр. Более того, затронуты проблемы, связанные с реализацией данного метода, а также оценена его эффективность в решении поставленной перед работой цели.

Полученные результаты не только обобщают знания о поддержке модификаций в видеоиграх, но и также, исследуя автоматизацию создания таких игровых систем, открывают возможности по дальнейшему упрощению их создания и, как следствие, распространению в индустрии для роста вовлеченности игроков как потребителей.

Список использованных источников:

1. Lee D., Lin D., Bezemer C., Hassan A. Building the perfect game – an empirical study of game modifications // Empirical Software Engineering. – 2020. – V. 25 – C. 2485-2518.
2. Miller W.G. Mod the World: How Entrepreneurs Learn from Video Game “Modding” Communities // The Invisible Hand in Virtual Worlds – The Economic Order of Video Games / Под ред. McCaffrey M. – Cambridge University Press. – 2021. – C. 183-210.
3. Hawranke T. Intrinsic Research – a Practice-Based Approach to Computer Game Modding // Playful Participatory Practices: Theoretical and Methodological Reflections / Под ред. P. Abend, B. Beil, V. Ossa – Springer VS Wiesbaden. – 2020. – C. 31-53.
4. Watson N. Re-Crafting Games: The inner life of Minecraft modding : дис. … Doctor of Philosophy (Communication) / Watson N. – Montreal, 2019 – 276 C.
5. Das D., Mathews N.S., Mathai A., Tamilselvam S., Sedamaki K., Chimalakonda S., Kumar A. COMEX: A Tool for Generating Customized Source Code Representations // In Proceedings of the 38th IEEE/ACM International Conference on Automated Software Engineering (ASE '23). – 2024. – C. 2054-2057.
6. Comparing Domain-Driven Design with Model-Driven Engineering [Электронный ресурс]. – URL: https://modeling-languages.com/comparing-domain-driven-design-model-driven-engineering (дата обращения 07.09.2024).
7. Bühler F., Barzen J., Harzenetter L., Leymann F., Wundrack P. Combining the Best of Two Worlds: Microservices and Micro Frontends as Basis for a New Plugin Architecture. // Service-Oriented Computing. SummerSOC 2022. Communications in Computer and Information Science – V. 1603 – 2022. – C. 3-23.
8. Marum O., Paulo J., Cunningham H., Jones J., Liu Y. Following the Writer’s Path to the Dynamically Coalescing Reactive Chains Design Pattern // Algorithms 2024. – 2024.