Миграция крупных программных проектов с использованием генеративного ИИ на современные кроссплатформенные технологии

А. А. Беляев*

Исследуются LLM-миграции двух C#-систем: EnergyCalc с Xamarin.Forms на Uno Platform и HouseBuilder с Xamarin.Forms и WPF/OpenTK на Godot. Схема оценки разделяет переиспользование ядра, создание хоста, участие LLM и сценарный паритет. В EnergyCalc сохранено 40–55% кода и пройдено 260 из 300 сценариев за 200 человеко-часов; в HouseBuilder переиспользовано 70–80% продуктового кода и пройдено 450 из 600 сценариев за 330 человеко-часов.

Ключевые слова: большие языковые модели, ИИ-агенты, миграция ПО, C#, Uno Platform, Godot.

1. Введение

При миграции системы требуется сохранить предметное ядро, заменив платформенный слой. Агент формирует изменения и использует сборку и тесты как обратную связь. Работы рассматривают взаимодействие с LLM [2] и крупномасштабные изменения [3], но не сопоставляют C#-миграции между UI/графическими платформами с разными границами ядра. В отличие от стандартного реинжиниринга [1], новизна работы — раздельный учёт ядра, нового хоста, участия LLM и паритета сценариев.

2. Методика

Строки кода, разметки, сцен и шейдеров подсчитаны в снимках от 05.10.2026. Сохранённым считался код библиотек и файлов с малыми адаптациями; диапазоны получены автоматическим сравнением и экспертной классификацией. Паритет — доля полностью пройденных равновесных

* *Беляев Артем Александрович* — Старший преподаватель, Кыргызско-Российский Славянский университет имени первого Президента Российской Федерации Б. Н. Ельцина, e-mail: a.a.beliaev@krsu.kg, ORCID: 0009-0004-2071-952X.

Beliaev Artem — Senior Lecturer, Kyrgyz-Russian Slavic University named after the First President of the Russian Federation B. N. Yeltsin.

Опубликовано на условиях лицензии CC-BY 4.0
Published under CC-BY 4.0

сценариев. Итерация — цикл “задача — изменение — сборка/тест — анализ”, иногда без полного теста; 30% циклов завершались автономно, 45% требовали подтверждения, 25% — исправления. Использовался Cursor с Grok 4.5/4.6; Claude Opus 5.x — для анализа и документации. Агенту были доступны репозиторий, терминал, сборка, тесты, приложение, скриншоты и Git без `push`. Токены не учитывались. Человеко-часы включают только труд участников: задачи и промпты 20%, контролируемое ожидание до 10%, анализ 20%, исправления 30%, тестирование 20%.

3. Миграция EnergyCalc на Uno Platform

Клиент EnergyCalc на Xamarin.Forms содержал 129 тыс. строк; решение на Uno Platform 6 [4] и .NET 9 — 119 тыс. Сохранены модель плана, расчётные библиотеки и API-клиент; 40–55% кода перенесено с сохранением логики. Разработчик/BA и QA за 1,5 месяца затратили 200 человеко-часов. Пройдено 260 из 300 сценариев (86,7%). Две подписки Cursor стоили 120 долл. при использовании около 80% лимитов.

4. Миграция HouseBuilder на Godot

В HouseBuilder Xamarin.Forms- и WPF/OpenTK-хосты заменены хостом Godot 4.7 .NET [5]. Доменный слой содержит 229 797 строк, исходный UI — 121 058, новый хост — 23 838. Контур 250–280 тыс. строк включает сохраняемые библиотеки и хост. Переиспользовано 70–80% продуктовых строк, но 5–15% UI/render; сходство переработанного GUI не превышает 55%. Back-разработчик, QA и BA за два месяца затратили 330 человеко-часов. Пройдено 450 из 600 сценариев (75%). Три подписки Cursor стоили 240 долл. при использовании около 80% лимитов.

5. Сравнение результатов

HouseBuilder сохранил ядро, но переработал GUI, тогда как Uno ближе к Xamarin.Forms. До работ команды оценили ручное достижение 80% функциональности при полной занятости. Это не контрольный эксперимент: состав и загрузка команд различались, поэтому коэффициент ускорения не вычислялся.

Таблица 1: Сравнительные показатели миграции

Показатель	EnergyCalc: Xamarin.Forms → Uno	HouseBuilder: Xamarin.Forms, WPF/OpenTK → Godot
Код	129 тыс. → 119 тыс.	контур 250–280 тыс.; хост 24 тыс.
Сохранено	40–55%	70–80% продукта; 5–15% UI/render
Команда; срок	2 человека; 1,5 мес.	3 человека; 2 мес.
Трудоёмкость	200 чел.-ч	330 чел.-ч
Сценарии	260/300 (86,7%)	450/600 (75%)
Подписки Cursor	120 долл.	240 долл.
Ручная оценка до работ	4 чел. × 6 мес.	6 чел. × 6–7 мес.

6. Ограничения и заключение

Исследование ограничено двумя проектами без контрольной ручной миграции и учёта токенов. Оно описывает практический результат, но не доказывает количественную эффективность LLM. Для такого вывода требуются одинаковые подвыборки задач, журнал времени и токенов и параллельное ручное выполнение. Полученные данные подтверждают применимость процесса с выделением ядра и обратной связью от сборки и тестов.

Список литературы

[1] E. J. Chikofsky, J. H. Cross II, “Reverse engineering and design recovery: A taxonomy”, *IEEE Software*, **7**:1 (1990), 13–17. DOI: 10.1109/52.43044.

[2] I. M, B. Omidvar-Tehrani, A. Anubhai, “Evaluating human–AI partnership for LLM-based code migration”, *CHI EA ’24*, 2024, 1–8. DOI: 10.1145/3613905.3650896.

[3] C. Ziftci et al., “Migrating code at scale with LLMs at Google”, *Companion Proceedings of the 33rd ACM International Conference on the Foundations of Software Engineering (FSE Companion ’25)*, ACM, New York, 2025, 162–173. DOI: 10.1145/3696630.3728542.

[4] Uno Platform, *Uno Platform documentation*, accessed 07.10.2026, https://platform.uno/docs/.

[5] Godot Engine, *Godot Engine documentation*, accessed 07.10.2026, https://docs.godotengine.org/ .

Статья поступила 8 октября 2026 г.

Generative AI-Assisted Migration of Large Software Projects to Modern Cross-Platform Technologies

A. Beliaev

Two LLM-assisted C# migrations are studied: EnergyCalc from Xamarin.Forms to Uno Platform and HouseBuilder from Xamarin.Forms and WPF/OpenTK to Godot. The assessment separates core reuse, host creation, LLM participation, and scenario parity. EnergyCalc retained 40–55% of code and passed 260 of 300 scenarios in 200 person-hours. HouseBuilder reused 70–80% of product code and passed 450 of 600 scenarios in 330 person-hours.

Keywords: large language models, AI agents, software migration, C#, Uno Platform, Godot.

References

[1] E. J. Chikofsky, J. H. Cross II, "Reverse engineering and design recovery: A taxonomy", *IEEE Software*, **7**:1 (1990), 13–17. DOI: 10.1109/52.43044.

[2] I. M, B. Omidvar-Tehrani, A. Anubhai, "Evaluating human–AI partnership for LLM-based code migration", *CHI EA '24*, 2024, 1–8. DOI: 10.1145/3613905.3650896.

[3] C. Ziftci et al., "Migrating code at scale with LLMs at Google", *Companion Proceedings of the 33rd ACM International Conference on the Foundations of Software Engineering (FSE Companion '25)*, ACM, New York, 2025, 162–173. DOI: 10.1145/3696630.3728542.

[4] Uno Platform, *Uno Platform documentation*, accessed 07.10.2026, https://platform.uno/docs/.

[5] Godot Engine, *Godot Engine documentation*, accessed 07.10.2026, https://docs.godotengine.org/.

Received on October 8, 2026