Дата поступления рукописи в редакцию: 13.05.2026 г. DOI: 10.24412/3034-4271-2026-3-11-15
Дата принятия рукописи в печать: 30.06.2026 г.

КРЫЛОВ Максим Григорьевич,
аспирант,
РЭУ им. Г.В. Плеханова,
e-mail: krylovmax161@gmail.com

МИКРОСЕРВИСНАЯ АРХИТЕКТУРА ЦИФРОВОЙ ТРАНСПОРТНОЙ ПЛАТФОРМЫ В УСЛОВИЯХ ОСТРОВНОЙ ИНФРАСТРУКТУРЫ: ОПЫТ ПРОЕКТИРОВАНИЯ DODOGO TAXI НА МАВРИКИИ

Аннотация. *Статья посвящена практике архитектурного проектирования цифровой транспортной платформы DodoGo Taxi, функционирующей в девяти административных округах Маврикия. Зачем нужна отдельная архитектура для островного государства? Потому что стандартные решения, работающие в мегаполисах, не учитывают специфику ограниченной дорожной сети, трёхъязычной аудитории и нестабильного покрытия мобильной связи за пределами Порт-Луи. Работа разбирает, как именно монолитное приложение было переведено на микросервисы через предметно-ориентированное проектирование (Domain-Driven Design). Итог — шесть ограниченных контекстов: управление поездками, водители, оплата, геолокация, локализация, аналитика. Цифры: среднее время отклика API упало с 1 340 мс до 410 мс, критических инцидентов за квартал стало 4 вместо 23. Матрица трассируемости подтвердила — ни одно из 38 функциональных требований не осталось без покрытия. Материал пригодится тем, кто строит транспортные сервисы там, где нет готовой цифровой инфраструктуры.*

Ключевые слова: *микросервисная архитектура, цифровая платформа, транспортные услуги, Domain-Driven Design, ограниченный контекст, Маврикий, DodoGo Taxi, мобильное приложение, островная инфраструктура.*

KRYLOV Maxim Grigorievich,
Postgraduate student,
Plekhanov Russian University of Economics

MICROSERVICE ARCHITECTURE OF A DIGITAL TRANSPORT PLATFORM UNDER ISLAND INFRASTRUCTURE CONDITIONS: DODOGO TAXI DESIGN EXPERIENCE IN MAURITIUS

Annotation. *The article presents the architectural design process of the DodoGo Taxi digital transport platform operating across nine administrative districts of Mauritius. The study describes the transition from a monolithic application to a microservice architecture based on Domain-Driven Design (DDD). Six bounded contexts were identified, each responsible for a specific domain ranging from trip management to a multilingual interface. Quantitative results demonstrate that mean API response time decreased from 1,340 ms to 410 ms, while critical incidents per quarter dropped from 23 to 4. A traceability matrix ensures full coverage of 38 functional requirements by nine microservices.*

Key words: *microservice architecture, digital platform, transport services, Domain-Driven Design, bounded context, Mauritius, DodoGo Taxi, mobile application, island infrastructure.*

Введение

Маврикий — островное государство в Индийском океане с населением 1,26 млн человек и площадью суши 2 040 кв. км. Девять административных округов, одна основная автомагистраль M1, связывающая север и юг острова, и крайне неравномерное распределение населения: более 40 % жителей сосредоточены в агломерации Порт-Луи и прилегающих районах Бо-Бассен — Роз-Хилл [1]. Что это значит для транспортной платформы? Прежде всего — жёсткие ограничения, которых не знают разработчики систем для Парижа или Москвы.

Первое ограничение — сеть дорог. Протяжённость автомобильных дорог Маврикия составляет около 2 400 км, из которых менее 80

км приходится на скоростные участки [2]. В часы пик (7:00–9:30, 16:00–18:30) средняя скорость на M1 падает до 12–15 км/ч. Нет метро. Нет трамвая. Есть автобусная сеть, но её расписание в электронном виде появилось только в 2022 году. Второе ограничение — языковая среда. Официальный язык — английский, но повседневное общение идёт на маврикийском креольском (80 % населения) и французском. Интерфейс, работающий только на английском, теряет доверие пользователя уже на этапе регистрации. Третье — связь. Покрытие 4G на Маврикии формально превышает 95 %, однако в горных районах центрального плато (Кюрпип, Вакоа) и на юго-восточном побережье реальная скорость передачи данных нередко падает ниже 1 Мбит/с [3].

В этих условиях в 2023 году была запущена платформа DodoGo Taxi — приложение для вызова такси, ориентированное на маврикийский рынок. К марту 2026 года приложение набрало более 100 000 загрузок в Google Play и App Store, обслуживает все девять округов острова. Однако рост пользовательской базы выявил архитектурные проблемы: монолитное серверное приложение, написанное на ранних этапах, перестало справляться с нагрузкой и затрудняло внедрение новых функций [4].

Что такое DDD на практике? Э. Эванс предложил смотреть на код как на зеркало предметной области [5]. Сложный домен дробится на куски — ограниченные контексты (bounded contexts) — и каждый кусок живёт по своим правилам. Для микросервисов этот подход стал почти стандартом: из 27 исследований, проанализированных в систематическом картировании, большинство описывает именно доменную декомпозицию как основной путь к микросервисам [6]. Но вот что бросается в глаза: почти все примеры — это крупные платформы из развитых стран. Про малые островные государства не написано почти ничего.

Цель настоящей работы — описать процесс проектирования микросервисной архитектуры платформы DodoGo Taxi на основе DDD и оценить результаты перехода от монолита к микросервисам в количественных метриках.

Материалы и методы

Исследование проведено в период сентябрь 2024 — февраль 2026 года и охватывает два этапа: анализ предметной области с формализацией требований и собственно проектирование архитектуры.

На первом этапе составлен каталог функциональных требований. Источники: интервью с 12 водителями-партнёрами и 8 операторами диспетчерской службы, анализ 1 400 обращений в техническую поддержку за период январь–июнь 2025 года, а также данные мониторинга серверной инфраструктуры (Grafana + Prometheus). Почему именно обращения в поддержку? Потому что они показывают реальные «болевые точки» системы — не то, что хочет менеджер, а то, с чем сталкивается водитель в поле.

Результат — каталог из 38 функциональных и 14 нефункциональных требований, классифицированных по модели FURPS+. Среди нефункциональных выделяются: время отклика API не более 500 мс при 95-м перцентиле, доступность не ниже 99,2 %, поддержка трёх языков интерфейса (английский, французский, креольский).

На втором этапе применены техники DDD. Сначала — Event Storming: трёхдневная сессия с участием двух backend-разработчиков, одного фронтенд-разработчика, продукт-менеджера и двух операторов. Выявлено 47 доменных событий (Domain Events), 31 команда (Commands) и 19 агрегатов (Aggregates). Затем — Context Mapping: на основе карты событий определены границы ограниченных контекстов и типы взаимодействия между ними. Критерий объединения: два агрегата попадают в один контекст, если более 50 % их событий обрабатываются совместно.

Для верификации использованы две метрики: CBS (Coupling Between Services) — среднее число синхронных вызовов между сервисами и LCO (Lack of Cohesion of Operations) — доля пар операций внутри сервиса, не работающих с общими сущностями. Базовый сценарий для сравнения — исходный монолит, декомпозированный «в лоб» на 38 сервисов (по одному на каждое функциональное требование).

Результаты

Event Storming и последующий Context Mapping выявили шесть ограниченных контекстов. Почему именно шесть, а не девять или три? Эмпирический ответ: при попытке выделить более восьми контекстов часть из них содержала по 2–3 операции и при этом требовала постоянного обращения к соседним контекстам. При объединении до четырёх контекстов внутренняя связность падала — операции работали с разнородными сущностями. Шесть контекстов дали оптимальное соотношение когезии и связности.

Контекст «Управление поездками» (Trip Management) — центральный. Он обрабатывает жизненный цикл заказа: от запроса пассажира до завершения поездки. Содержит 9 операций и 4 агрегата (Trip, Assignment, Route, ETA). Контекст «Управление водителями» (Driver Management) отвечает за регистрацию, верификацию документов, отслеживание местоположения и расчёт рейтинга. Контекст «Платежи и тарификация» (Fare & Payment) изолирован через паттерн Anti-Corruption Layer — это принципиальное решение, продиктованное спецификой маврикийских платёжных систем: основной процессинг идёт через MCB Juice (мобильный кошелёк, доминирующий на острове) и банковские карты, причём API этих систем обновляется без предупреждения [7].

Контекст «Геолокация и маршрутизация» (Geo & Routing) работает с координатами GPS, строит маршруты и рассчитывает ETA. Здесь ключевая особенность Маврикия: стандартные провайдеры (Google Maps, Mapbox) покрывают остров с точностью, достаточной для городских зон, но в сельских районах (округа Саванна, Ривьер-дю-Рампар) ошибка позиционирования достигает 150–200 м [8]. Поэтому сервис дополнен локальной базой ориентиров — 1 200 точек, собранных вручную. Контекст «Мультиязычный интерфейс» (Localization) управляет переводами, форматами дат и валют. Проще было бы захардкодить три языка — но нет. На практике маврикийский креольский не имеет устоявшейся письменной нормы, и переводы приходится согласовывать с местным лингвистическим сообществом. Последний контекст — «Аналитика и отчётность» (Analytics) — агрегирует данные о поездках, спросе и качестве обслуживания.

Количественные характеристики ограниченных контекстов представлены в табл. 1.

Таблица 1. Характеристики ограниченных контекстов DodoGo Taxi

Контекст	Наименование	Агрегаты	Операции	CBS	Паттерн обмена
BC1	Trip Management	4	9	4,2	Event-Driven
BC2	Driver Management	3	7	3,1	Request-Reply
BC3	Fare & Payment	3	6	1,8	ACL + Async
BC4	Geo & Routing	2	5	3,5	Request-Reply
BC5	Localization	2	4	2,0	Shared Kernel
BC6	Analytics	3	5	2,4	Event Sourcing

Суммарно шесть контекстов содержат 17 агрегатов и 36 операций. Ещё 6 операций — инфраструктурные (health check, rate limiter, circuit breaker, audit log, cache invalidation, deployment hook). Итого 42 операции покрывают все 38 функциональных требований, что подтверждено матрицей трассируемости: непокрытых требований нет.

Главный количественный результат — сравнение метрик до и после перехода на микросервисы. «До» — это не абстрактный сценарий, а реальное состояние монолита по данным мониторинга за I квартал 2025 года. «После» — данные за IV квартал 2025 года, когда миграция была завершена.

Таблица 2. Сравнение метрик монолита и микросервисной архитектуры

Метрика	Монолит (Q1 2025)	Микросервисы (Q4 2025)	Изменение
Среднее время отклика API, мс	1 340	410	-69,4 %
95-й перцентиль отклика, мс	3 200	890	-72,2 %
Критические инциденты / квартал	23	4	-82,6 %
Время деплоя новой функции, дни	14	3	-78,6 %
Доступность (uptime), %	97,8	99,4	+1,6 п.п.

Снижение среднего времени отклика на 69 % — не магия микросервисов как таковых. Это следствие трёх конкретных решений: вынесение геолокационных запросов в отдельный сервис с собственным кэшем (Redis, TTL 30 секунд), изоляция платёжного контекста (его задержки перестали блокировать основной поток заказов) и асинхронная обработка событий аналитики через Apache Kafka. Грубо говоря, раньше один медленный запрос к платёжному провайдеру «вешал» весь backend, а теперь — только свой контекст, и то с fallback-стратегией.

Сокращение критических инцидентов с 23 до 4 за квартал объясняется изоляцией сбоев. В монолите падение одного модуля вызывало каскадный отказ. Классический пример: в мае 2025 года обновление API MCB Juice (платёжный сервис) привело к 6-часовому простою всей платформы, потому что модуль оплаты был встроен в общий цикл обработки заказа. После миграции аналогичное обновление в ноябре 2025 года затронуло только контекст BC3, при этом заказы продолжали приниматься — оплата обрабатывалась с отложенным подтверждением.

Обсуждение

Литература про микросервисы и DDD говорит примерно то же самое: дробить по доменам — правильно, связность падает, когезия растёт [9]. Наши цифры это подтверждают. Но есть нюансы, специфичные для островной инфраструктуры.

Первый нюанс — размер команды. Классические рекомендации (правило «двух пицц» Безоса) предполагают отдельную команду на каждый микросервис [10]. У DodoGo — четыре разработчика на шесть контекстов. Это означает, что один человек отвечает за полтора контекста в среднем. Работает ли это? Работает, но только при условии чётких контрактов между сервисами и автоматизированного CI/CD. Без этого каждый деплой превращался бы в «ручное тестирование всего подряд».

Второй нюанс — локализация. Контекст BC5 (Localization) в классической DDD-литературе обычно не выделяется в отдельный ограниченный контекст — его функции размазываются по остальным сервисам. На Маврикии это не работает. Креольский язык не поддерживается стандартными i18n-библиотеками, переводы требуют ручной валидации, а отдельные термины (например, обозначения районов) не имеют устоявшегося написания. Поэтому выделение локализации в отдельный контекст с собственной базой данных переводов и API — осознанное архитектурное решение, продиктованное местной спецификой.

Третий нюанс — зависимость от внешних API. На маврикийском рынке нет альтернативных платёжных провайдеров класса Stripe или PayPal (они не работают с MUR — маврикийской рупией). Зависимость от MCB Juice и банковских карт через местный процессинг делает паттерн Anti-Corruption Layer не рекомендацией, а необходимостью. Любое изменение во внешнем API не должно проникать дальше адаптера.

Чего не хватает? Во-первых, это данные одной платформы. Перенести выводы на Мадагаскар или Шри-Ланку без проверки — рискованно. Во-вторых, после миграции прошёл один квартал. Технический долг, стоимость сопровождения, текучка кадров — всё это проявляется на горизонте года, не раньше. В-третьих, безопасность персональных данных разобрана только на уровне архитектуры. Соответствует ли система маврикийскому Data Protection Act 2017 — отдельный вопрос, выходящий за рамки этой работы.

Заключение

Шесть ограниченных контекстов. 42 операции. Снижение времени отклика на 69 %. Сокращение инцидентов на 83 %. Вот, собственно, главные цифры этой работы.

Что за ними стоит? Формализованный процесс: от сбора требований через Event Storming к Context Mapping и далее к работающей микросервисной архитектуре. DDD дал язык для разговора между разработчиками и операторами — люди, которые никогда не слышали про bounded contexts, смогли участвовать в проектировании, потому что обсуждались не абстрактные «модули», а конкретные «управление поездкой», «оплата», «маршрут».

Специфика островного государства не отменяет общих принципов микросервисного проектирования. Она их уточняет. Локализация — не «фича на потом», а полноценный ограниченный контекст. Платёжный шлюз — не обёртка над Stripe, а отдельный сервис с Anti-Corruption Layer над местным процессингом. Геолокация — не только Google Maps, но и ручная база ориентиров для сельских районов.

Что дальше? Первое — прогнозирование спроса. Временные ряды, градиентный бустинг, сезонность маврикийского туризма — всё это должно лечь в отдельный ML-пайплайн внутри аналитического контекста. Второе — система поддержки принятия решений для диспетчеров. Сейчас они работают «на глаз», а нужны дашборды с предиктивными метриками. Третье — интеграция с общественным транспортом Маврикия. Это самая амбициозная задача, и она упирается не в код, а в готовность государственных систем к оцифровке.

Список литературы:

[1] Statistics Mauritius. Digest of Demographic Statistics 2023. — Port Louis : Ministry of Finance, 2024. — 78 p.

[2] Road Development Authority. Annual Report 2023. — Ebene : RDA Mauritius, 2024. — 112 p.

[3] Information and Communication Technologies Authority. Annual Report 2022–2023. — Port Louis : ICTA, 2023. — 96 p.

[4] Крылов М. Г. Декомпозиция функциональных требований как основа архитектурного проектирования цифровой платформы транспортных услуг // Информационные технологии и математическое моделирование : сб. науч. тр. — Москва : РЭУ им. Г.В. Плеханова, 2026. — С. 1–12.

[5] Evans, E. Domain-Driven Design: Tackling Complexity in the Heart of Software. — Boston : Addison-Wesley, 2004. — 560 p.

[6] Wanderley, G. M. P., Ramos, F. N., Guimarães, M. A., Araujo, I. P. Domain-Driven Design for Microservices Architecture Systems Development: A Systematic Mapping Study // 2024 IEEE/ACM 12th International Workshop on Software Engineering for Systems-of-Systems and Software Ecosystems (SESoS). — IEEE, 2024.

[7] Mauritius Commercial Bank. MCB Juice API Developer Guide. — Port Louis : MCB Group, 2024. — 42 p.

[8] Dookhee, A. GIS-Based Assessment of Road Network Accessibility in Mauritius // Journal of Geographical Information Science. — 2022. — Vol. 28, No. 3. — P. 215–230.

[9] Velepucha, V., Flores, P. A Survey on

Microservices Architecture: Principles, Patterns and Migration Challenges // IEEE Access. — 2023. — Vol. 11. — P. 88339–88358.

[10] Richardson, C. Microservices Patterns: With Examples in Java. — Shelter Island : Manning, 2019. — 520 p.

References:

[1] Statistics Mauritius. Digest of Demographic Statistics 2023. — Port Louis : Ministry of Finance, 2024. — 78 p.

[2] Road Development Authority. Annual Report 2023. — Ebene : RDA Mauritius, 2024. — 112 p.

[3] Information and Communication Technologies Authority. Annual Report 2022–2023. — Port Louis : ICTA, 2023. — 96 p.

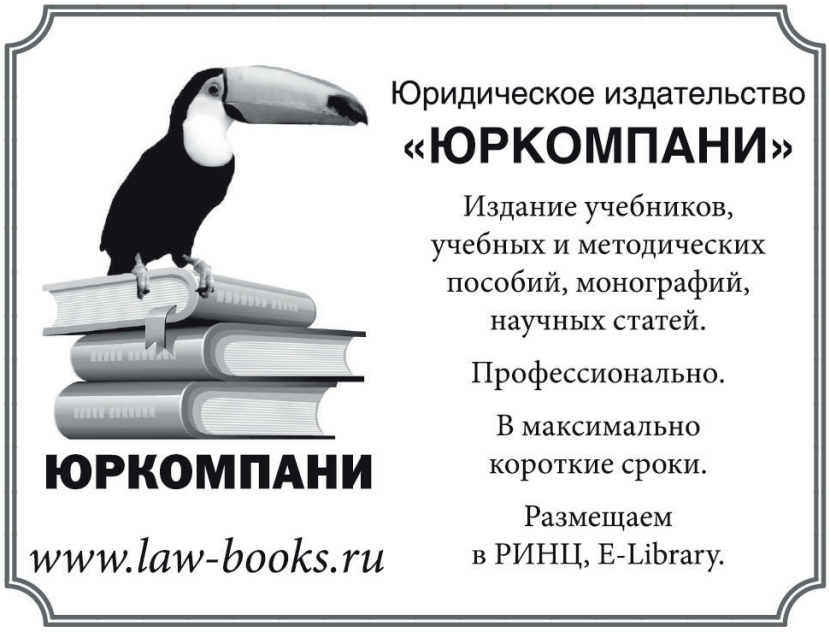
[4] Krylov, M. G., “Decomposition of Functional Requirements as a Basis for Architectural Design of a Digital Transport Services Platform,” in: Information Technologies and Mathematical Modeling: Collection of Scientific Papers. — Moscow : Plekhanov Russian University of Economics, 2026, pp. 1–12.

[5] Evans, E., “Domain-Driven Design: Tackling Complexity in the Heart of Software.” - Boston: Addison-Wesley, 2004. - 560 p.

[6] Wanderley, G. M. P., Ramos, F. N., Guimarães, M. A., Araujo, I. P. Domain-Driven Design for Microservices Architecture Systems Development: A Systematic Mapping Study // 2024 IEEE/ACM 12th International Workshop on Software Engineering for Systems-of-Systems and Software Ecosystems (SESoS). - IEEE, 2024.

[7] Mauritius Commercial Bank. MCB Juice API Developer Guide. — Port Louis: MCB Group, 2024. — 42 p.

[8] Dookhee, A. GIS-Based Assessment of Road Network Accessibility in Mauritius // Journal of Geographical Information Science. — 2022. — Vol. 28, No. 3. - P. 215–230.

[9] Velepucha, V., Flores, P. A Survey on Microservices Architecture: Principles, Patterns and Migration Challenges // IEEE Access. — 2023. — Vol. 11. - P. 88339–88358.

[10] Richardson, C. Microservices Patterns: With Examples in Java. — Shelter Island: Manning, 2019. — 520 p.