DOI: 10.58168/PPSPM2025_170-176
УДК 004.9

АРХИТЕКТУРНЫЕ ПАТТЕРНЫ ДЛЯ БЕССЕРВЕРНЫХ ПРИЛОЖЕНИЙ: СРАВНИТЕЛЬНЫЙ АНАЛИЗ ПОДХОДОВ И ПРАКТИЧЕСКИЕ АСПЕКТЫ РЕАЛИЗАЦИИ

А.М. Тюнина, В.В. Кондусова, Д.С. Кукуева, Сахаров С.Л.
ФГБОУ ВО «Воронежский государственный лесотехнический университет имени Г.Ф. Морозова»

В статье проводится комплексный анализ архитектурных паттернов для бессерверных (Serverless) приложений в контексте современных облачных платформ. Исследование охватывает три ключевых паттерна: Strangler Fig для постепенной миграции монолитных систем, Event-Driven для асинхронной обработки событий и Saga для управления распределенными транзакциями. Представлен детальный сравнительный анализ паттернов по критериям сложности реализации, типу взаимодействия, областям применения и потенциальным рискам. Особое внимание уделено практическим аспектам реализации бессерверной архитектуры, включая проблемы холодного старта, ограничения времени выполнения функций и вопросы мониторинга распределенных систем. На основе анализа современных FaaS-платформ (AWS Lambda, Azure Functions, Google Cloud Functions) сформулированы рекомендации по выбору оптимального архитектурного подхода в зависимости от бизнес-требований и характеристик нагрузки. Результаты исследования демонстрируют, что правильный выбор архитектурного паттерна позволяет существенно повысить масштабируемость, отказоустойчивость и экономическую эффективность облачных приложений.

Ключевые слова: бессерверная архитектура, FaaS, Serverless, облачные вычисления, архитектурные паттерны, Strangler Fig, Event-Driven, Saga, AWS Lambda, Azure Functions, микросервисы, холодный старт.

ARCHITECTURAL PATTERNS FOR SERVERLESS APPLICATIONS: A COMPARATIVE ANALYSIS OF APPROACHES AND PRACTICAL ASPECTS OF IMPLEMENTATION

A.M. Tyunina, V.V. Kondusova, D.S. Kukueva, Sakharov S.L.
Voronezh State University of Forestry and Technologies named after G.F. Morozov

The article provides a comprehensive analysis of architectural patterns for serverless applications in the context of modern cloud platforms. The study covers three key patterns: Strangler Fig for gradual migration of monolithic systems, Event-Driven for asynchronous event processing, and Saga for distributed transaction management. A detailed comparative analysis of the patterns is presented according to the criteria of complexity of implementation, type of interaction, areas of application and potential risks. Particular attention is paid to the practical aspects of implementing a serverless architecture, including cold start issues, function execution time constraints, and distributed system monitoring issues. Based on the analysis of modern FaaS platforms (AWS Lambda, Azure Functions, Google Cloud Functions), recommendations have been formulated for choosing the optimal architectural approach depending on business requirements and load characteristics. The results of the study demonstrate that choosing the right architectural pattern can significantly increase the scalability, fault tolerance, and cost-effectiveness of cloud applications.

Keywords: serverless architecture, FaaS, Serverless, cloud computing, architectural patterns, Strangler Fig, Event-Driven, Saga, AWS Lambda, Azure Functions, microservices, cold start.

Бессерверная архитектура стала одним из ключевых трендов в разработке облачных приложений, предлагая принципиально новый подход к управлению вычислительными ресурсами. Однако переход к Serverless-парадигме сопровождается значительными архитектурными вызовами, связанными с распределенным характером выполнения кода, ограничениями платформ и необходимостью переосмысления традиционных подходов к проектированию систем. Традиционные монолитные архитектуры и даже микросервисные подходы демонстрируют ограниченную эффективность в условиях FaaS-модели, где функции выполняются эфемерно и масштабируются независимо. Проблема усугубляется отсутствием унифицированных методик проектирования бессерверных систем и недостатком практического опыта у разработчиков. Особую сложность представляют задачи обеспечения согласованности данных в распределенных транзакциях, управления состоянием между вызовами функций и оптимизации производительности в условиях холодного старта. В связи с этим возникает необходимость в систематическом анализе архитектурных паттернов, специально адаптированных для бессерверной среды, и выработке критериев их выбора для различных классов приложений. Актуальность исследования обусловлена растущим распространением FaaS-платформ и потребностью в методических основах проектирования эффективных бессерверных систем.

Бессерверные технологии — один из ключевых трендов современной архитектуры приложений. Подход Serverless позволяет разработчикам сосредоточиться на бизнес-логике, а вопросы масштабирования, инфраструктуры и эксплуатации отдать облачным платформам.

В основе этого подхода лежит модель FaaS (Function-as-a-Service), где каждая функция выполняется по запросу, не требуя выделенного сервера. Это подход, при котором разработчики фокусируются на логике, а управление инфраструктурой полностью берёт на себя облачный провайдер. В контексте

FaaS (Function-as-a-Service), таких как AWS Lambda, Azure Functions и Google Cloud Functions, особенно актуальны определённые архитектурные паттерны, облегчающие построение масштабируемых, адаптивных и раздельно разворачиваемых систем. [1]

Таблица 1 – Сравнение паттернов

Критерий	Strangler Fig	Event-driven	Saga
Главная цель	Миграция монолита	Асинхронные распределённые процессы	Распределённая согласованность
Тип взаимодействия	Синхронное API	Асинхронные события	Асинхронные/оркестрируемые шаги
Сложность	Средняя	Высокая	Высокая
Хорошо подходит для:	Legacy-to-serverless перехода	Потоков данных, IoT, очередей	Бизнес-процессов, транзакций
Риски	Дублирование функционала	Сложно отслеживать цепочки	Трудно писать компенсации

Strangler Fig — это паттерн постепенной замены частей монолитной системы без остановки всей платформы. Идея заключается в том, чтобы «обвить» существующее приложение новыми компонентами, перенаправлять к ним часть трафика и со временем полностью заменить устаревшие элементы. В мире FaaS данный паттерн используется для миграции логики монолита в отдельные функции. API Gateway или аналогичный маршрутизатор управляет тем, какие запросы обрабатываются старой системой, а какие — новой

бессерверной логикой. Strangler Fig — отличный выбор, если ваша основная задача — постепенная миграция на Serverless, а не создание системы с нуля.

Событийно-ориентированный подход — наиболее органическая архитектура для бессерверного стека. Здесь функции запускаются в ответ на события: приход сообщения в очередь, изменение данных в хранилище, HTTP-запрос, таймер или IoT-сигнал. FaaS идеально масштабируется под непредсказуемые нагрузки, а событийно-ориентированная архитектура создаёт слабосвязанные компоненты. Каждая функция отвечает за свой маленький шаг — и вызывается лишь тогда, когда это необходимо. Saga — оптимальный выбор, когда Serverless должен поддерживать сложные транзакционные сценарии без потери согласованности.

Saga — это механизм управления распределёнными транзакциями, который разделяет одну большую транзакцию на последовательность локальных операций. Каждая операция выполняется в рамках одного сервиса или функции и является автономной. Если какой-то этап не удаётся выполнить успешно, запускаются компенсирующие действия, которые возвращают систему в согласованное состояние.

Вместо строгой последовательности ACID Saga гарантирует согласованность данных в течение определённого времени через корректно организованную реакцию на события.

Saga — один из ключевых паттернов, делающих распределённые бессерверные приложения надёжными и согласованными. Она позволяет компенсировать невозможность запуска ACID-транзакций в FaaS, разложив процесс на независимые шаги с чётко определённой логикой отката. В сочетании с событийной архитектурой и оркестраторами Saga формирует надёжный фундамент для сложных облачных систем: e-commerce, платежей, биллинга, логистики и всех областей, где важна целостность данных.

Несмотря на преимущества serverless-подхода, он приносит и ряд характерных проблем, которые необходимо учитывать при проектировании системы.

Холодный старт — задержка при первом вызове функции после периода бездействия. Он возникает из-за необходимости загрузить контейнер, зависимости и инициализировать окружение.

Большинство FaaS-платформ ограничивает продолжительность вызова функции — до нескольких минут. Это усложняет выполнение длительных задач, особенно связанных с обработкой больших данных. Архитектура

бессерверных приложений — это не просто набор FaaS- функций, а комплексный подход, требующий глубокого понимания распределённых систем. Паттерны Strangler Fig, Event-Driven и Saga представляют собой фундаментальные строительные блоки Serverless- архитектуры. При этом разработчикам нужно помнить о характерных сложностях бессерверных систем: холодных стартах, ограничениях времени выполнения и проблемах с мониторингом. Эти особенности требуют переосмысления привычных подходов и адаптации к требованиям облачного мира. [2]

В конечном счёте Serverless — это не только технологический тренд, но и новый взгляд на проектирование архитектур, где ценность создаётся небольшими и гибкими компонентами, а масштабирование и инфраструктура становятся заботой облака.

Перспективы развития архитектурных подходов для бессерверных приложений связаны с созданием гибридных моделей, сочетающих преимущества различных паттернов и адаптирующихся к изменяющимся требованиям бизнеса. Особый потенциал демонстрируют подходы, основанные на машинном обучении для прогнозирования нагрузки и оптимизации распределения функций между различными FaaS-провайдерами. Дальнейшие исследования должны быть направлены на разработку стандартизированных интерфейсов для оркестрации сложных бизнес-процессов в бессерверной среде и создание эффективных механизмов управления состоянием распределенных транзакций. Важным направлением является также преодоление ограничений текущих FaaS-платформ, включая минимизацию задержек холодного старта и увеличение максимального времени выполнения функций для ресурсоемких задач. Учитывая растущую популярность edge-вычислений, особую актуальность приобретают исследования в области распределенных бессерверных архитектур, способных эффективно работать в условиях нестабильного сетевого соединения и ограниченных ресурсов периферийных устройств. Развитие стандартов и инструментов мониторинга бессерверных систем открывает новые возможности для создания самовосстанавливающихся архитектур, способных автоматически адаптироваться к сбоям и изменяющимся условиям работы.

Список литературы

1. Тхашоков, М. Х. Уязвимости бессерверных приложений и перспективы их применения / М. Х. Тхашоков // Управление в современных системах : сборник трудов IX Всероссийской (национальной) научно-практической конференции научных, научно-педагогических работников и аспирантов, Челябинск, 12 декабря 2019 года. – Челябинск: Южно-Уральский технологический университет, 2019. – С. 350-356. – EDN FXSZAK.

2. Актуальность бессерверных приложений и вычислений с помощью функций Azure / Д. С. Кириллов, Э. Ф. Насиров, Г. Р. Мертинс, Д. Д. Молостов // СОВРЕМЕННАЯ НАУКА: АКТУАЛЬНЫЕ ВОПРОСЫ, ДОСТИЖЕНИЯ и ИННОВАЦИИ : сборник статей XXIII Международной научно-практической конференции, Пенза, 10 января 2022 года. – Пенза: Наука и Просвещение (ИП Гуляев Г.Ю.), 2022. – С. 53-55. – EDN RXEOHF.

3. Мархакшинов, А. Л. Разработка бессерверных мобильных приложений / А. Л. Мархакшинов // Информационные системы и технологии в образовании, науке и бизнесе : Материалы всероссийской научно-практической конференции с международным участием, Улан-Удэ, 05 июля 2019 года / Научный редактор Е.Р. Урмакшинова. – Улан-Удэ: Бурятский государственный университет имени Доржи Банзарова, 2019. – С. 88-91. – DOI 10.18101/978-5-9793-1397-9-88-91. – EDN CLNLFL.

4. Нажимова, Н. А. Исследование методологии devops для разработки программного обеспечения / Н. А. Нажимова, А. А. Вдовин // Научное обозрение. Технические науки. – 2023. – № 2. – С. 44-49. – DOI 10.17513/srts.1433. – EDN GUEMHO.

5. Гордина, А. Т. Проектирование приложений Serverless-архитектуры / А. Т. Гордина, А. В. Забродин, А. Д. Хомоненко // Вестник Российского нового университета. Серия: Сложные системы: модели, анализ и управление. – 2022. – № 2. – С. 140-148. – DOI 10.18137/RNU.V9187.22.02.P.140. – EDN NTOTJS.

References

1. Tkhashokov, M. H. Vulnerabilities of serverless applications and prospects for their application / M. H. Tkhashokov // Management in modern systems : proceedings of the IX All-Russian (national) Scientific and Practical Conference of scientific, scientific and pedagogical workers and graduate students, Chelyabinsk,

December 12, 2019. Chelyabinsk: South Ural Technological University, 2019. pp. 350-356. EDN FXSZAK.

2. The relevance of serverless applications and computing using Azure functions / D. S. Kirillov, E. F. Nasirov, G. R. Mertins, D. D. Molostov // MODERN SCIENCE: CURRENT ISSUES, ACHIEVEMENTS and INNOVATIONS : collection of articles of the XXIII International Scientific and Practical Conference, Penza, January 10, 2022. Penza: Science and Education (IP Gulyaev G.Yu.), 2022. pp. 53-55. - EDN RXEOHF.

3. Markhakshinov, A. L. Development of serverless mobile applications / A. L. Markhakshinov // Information systems and technologies in education, science and business : Proceedings of the All-Russian scientific and practical conference with international participation, Ulan-Ude, July 05, 2019 / Scientific editor E.R. Urmakshinova. – Ulan-Ude: Dorzhi Banzarov Buryat State University, 2019. – pp. 88-91. – DOI 10.18101/978-5-9793-1397-9-88-91. – EDN CLNLFL.

4. Najdova, N. A. A study of devops methodology for software development / N. A. Najdova, A. A. Vdovin // Scientific Review. Technical sciences. - 2023. – No. 2. – pp. 44-49. – DOI 10.17513/srts.1433. – EDN GUEMHO.

5. Gordina, A. T. Designing Serverless architecture applications / A. T. Gordina, A.V. Zabrodin, A.D. Khomonenko // Bulletin of the Russian New University. Series: Complex systems: models, analysis and management. – 2022. – No. 2. – pp. 140-148. – DOI 10.18137/RNU.V9187.22.02.P.140. – EDN NTOTJS.