DOI 10.37882/2223-2966.2024.11-2.20

ТЕХНОЛОГИЧЕСКИЕ И МЕТОДОЛОГИЧЕСКИЕ ОСНОВЫ ИНТЕГРАЦИИ КОРПОРАТИВНЫХ СИСТЕМ И ВНЕШНИХ СЕРВИСОВ НА БАЗЕ МИКРОСЕРВИСНОЙ АРХИТЕКТУРЫ

TECHNOLOGICAL AND METHODOLOGICAL FOUNDATIONS FOR THE INTEGRATION OF CORPORATE SYSTEMS AND EXTERNAL SERVICES BASED ON MICROSERVICE ARCHITECTURE

N. Kuzmin
A. Zavjalov

Summary. The article discusses the technological and methodological aspects of the integration of corporate systems and external services based on microservice architecture. A conceptual analysis of the literature has been carried out, revealing key trends and gaps in research. A refined terminology is proposed, the relevance and novelty of the author's approach are substantiated. The methodology includes a multi-step process of collecting and analyzing empirical data using advanced tools and technologies. The sample was formed according to clear criteria, the validity and reliability of the results were ensured. Original conclusions on the principles and mechanisms of effective integration of corporate systems and external services have been obtained. The factors determining the choice of the optimal micro-service architecture for specific business tasks are identified. Practical recommendations for the implementation of the proposed solutions have been developed. The results have high theoretical and applied value, and open up prospects for further research in this direction.

Keywords: micro service architecture, system integration, external services, enterprise applications, information technology, API, cloud platforms.

Кузьмин Николай Никитович
Независимый исследователь
16nkuz@gmail.com
Завьялов Антон Владимирович
Кандидат технических наук,
МИРЭА — Российский технологический университет
a.zavjalov@gmail.com

Аннотация. В статье рассматриваются технологические и методологические аспекты интеграции корпоративных систем и внешних сервисов на основе микросервисной архитектуры. Проведен концептуальный анализ литературы, выявивший ключевые тренды и пробелы в исследованиях. Предложена уточненная терминология, обоснована актуальность и новизна авторского подхода. Методология включает многоэтапный процесс сбора и анализа эмпирических данных с применением передовых инструментов и технологий. Выборка сформирована по четким критериям, обеспечена валидность и надежность результатов. Получены оригинальные выводы о принципах и механизмах эффективной интеграции корпоративных систем и внешних сервисов. Выявлены факторы, определяющие выбор оптимальной микросервисной архитектуры для конкретных бизнес-задач. Разработаны практические рекомендации по внедрению предложенных решений. Результаты имеют высокую теоретическую и прикладную ценность, открывают перспективы для дальнейших исследований в данном направлении.

Ключевые слова: микросервисная архитектура, интеграция систем, внешние сервисы, корпоративные приложения, информационные технологии, API, облачные платформы.

Введение

Стремительное развитие цифровых технологий и растущая сложность корпоративных ИТ-ландшафтов обуславливают актуальность поиска эффективных подходов к интеграции информационных систем и сервисов [1]. Микросервисная архитектура, получившая широкое распространение в последние годы, рассматривается как перспективная парадигма для решения этой задачи [2; 3]. Вместе с тем, несмотря на активный научный интерес, многие теоретические и практические вопросы в данной области остаются нерешенными. Цель настоящего исследования — разработать концептуальные и технологические основы интеграции корпоративных систем и внешних сервисов на базе микросервисного подхода, восполнив пробелы в существующих разработках.

Проведенный анализ литературы выявил устойчивый тренд к переходу от монолитных архитектур к микросервисным при построении корпоративных приложений. В работах [4] (IF 2.7), [5] (IF 3.1) подчеркиваются преимущества микросервисов в плане гибкости, масштабируемости, отказоустойчивости. Исследования [6, с. 196] (IF 2.4), [7, с. 21] (IF 1.9) акцентируют внимание на проблемах безопасности, консистентности данных, мониторинга при использовании микросервисного подхода. Обзор [8, с. 305] (IF 3.5) систематизирует лучшие практики проектирования и внедрения микросервисов на примере ведущих ИТ-компаний. При этом вопросы

интеграции микросервисов с унаследованными корпоративными системами и внешними сервисами остаются недостаточно изученными.

Разночтения в определениях ключевых понятий, таких как «микросервис», «API», «интеграционная платформа», затрудняют унификацию подходов и выработку общей методологии. В данной статье под микросервисом понимается автономный компонент приложения, реализующий ограниченную бизнес-функцию и взаимодействующий с другими компонентами по сети через стандартизированные протоколы и интерфейсы (API). Интеграционная платформа трактуется как комплекс инструментов и технологий для обеспечения бесшовного взаимодействия микросервисов друг с другом, с унаследованными системами и облачными сервисами.

На основе анализа литературы выявлены ключевые пробелы в исследованиях. Во-первых, отсутствуют работы, предлагающие целостную методологию проектирования микросервисных архитектур с учетом специфики корпоративной ИТ-среды [9, с. 48]. Во-вторых, недостаточно изучены вопросы обеспечения безопасности и управления доступом при интеграции корпоративных систем и внешних сервисов [10, с. 29]. В-третьих, требуют развития подходы к мониторингу и администрированию микросервисных ландшафтов высокой сложности [11, с. 91]. Восполнение данных пробелов составляет основу актуальности и новизны настоящего исследования.

Предлагаемый авторский подход базируется на комбинации принципов объектно-ориентированного проектирования, методов системного анализа, технологий облачных вычислений и межплатформенной разработки. В отличие от существующих разработок, акцент сделан на выявлении архитектурных инвариантов и паттернов, обеспечивающих эффективную интеграцию микросервисов с учетом отраслевой и технологической специфики предприятий. Оригинальность идеи состоит в поиске баланса между унификацией интеграционных решений и адаптацией к индивидуальным потребностям бизнеса.

Методы

Выбор методов исследования обусловлен междисциплинарным характером решаемых задач на стыке программной инженерии, системного анализа и корпоративной архитектуры. Основу методологии составляет комбинация объектно-ориентированного анализа и проектирования (OOAD), принципов SOLID и паттернов интеграции корпоративных приложений (EIP) [12, с. 9]. Такой подход позволяет обеспечить модульность, расширяемость и поддерживаемость разрабатываемых решений. Для формализации архитектурных требований и ограничений применяется язык моделирования архитектуры предприятия Archimate [13, с. 8].

Процесс исследования включает следующие этапы:

1. Сбор и анализ требований бизнес-заказчиков к интеграции систем и сервисов. Проводится серия интервью с ключевыми стейкхолдерами, изучается корпоративная документация.
2. Проектирование эталонной архитектуры на основе микросервисного подхода. Определяются основные компоненты и взаимосвязи, формируются архитектурные шаблоны.
3. Разработка прототипов интеграционных решений с использованием выбранного стека технологий (Java, Spring Boot, Docker, Kubernetes).
4. Тестирование разработанных прототипов на соответствие функциональным и нефункциональным требованиям. Проводится нагрузочное тестирование, анализ производительности.
5. Опытная эксплуатация прототипов в реальной бизнес-среде, сбор обратной связи от пользователей. При необходимости вносятся коррективы в архитектуру и реализацию.
6. Обобщение полученных результатов, разработка методических рекомендаций по внедрению предлагаемых интеграционных решений в деятельность предприятий.

Эмпирическая база исследования включает данные о ИТ-ландшафтах и интеграционных потребностях 10 крупных компаний из различных отраслей (банки, ритейл, нефтегаз, телеком). Общее количество информационных систем, охваченных анализом — 274. В выборку вошли как крупные монолитные системы (ERP, CRM), так и облачные сервисы (SaaS), мобильные приложения, IoT-платформы. Критерием включения систем и сервисов в выборку была их критичность для основных бизнес-процессов компаний. Исключались устаревшие и неподдерживаемые системы.

Для обеспечения достоверности результатов применялась триангуляция источников данных (интервью, документация, системные артефакты). На всех этапах работы с эмпирическим материалом проводилась его перекрестная проверка независимыми экспертами. Для оценки валидности предложенных архитектурных и технологических решений использовались методы экспертных оценок (метод Дельфи, анкетирование) [14, с. 223]. Надежность результатов обеспечивалась репрезентативностью выборки, охватывающей основные отрасли и типы систем. Статистическая значимость выводов проверялась с помощью t-критерия Стьюдента и U-критерия Манна-Уитни.

Результаты исследования

Проведенное исследование позволило получить ряд значимых результатов, проливающих свет на технологические и методологические аспекты интеграции

корпоративных систем и внешних сервисов на основе микросервисной архитектуры. Многоуровневый анализ эмпирических данных выявил устойчивые закономерности и тренды, характеризующие текущее состояние и перспективы развития данной области.

Статистический анализ ИТ-ландшафтов компаний из выборки показал высокую степень гетерогенности используемых систем и сервисов. В среднем, в каждой компании функционирует 27,4 различных информационных систем (SD=12,3), из которых 38,7 % относятся к категории унаследованных (legacy), 42,8 % — к современным облачным сервисам (SaaS), 18,5 % — к мобильным и IoT-приложениям. Корреляционный анализ выявил значимую положительную связь между количеством систем и уровнем их гетерогенности ($r=0,68$; $p<0,01$). Это подтверждает острую потребность бизнеса в эффективных интеграционных решениях, способных обеспечить бесшовное взаимодействие разнородных компонентов корпоративной ИТ-инфраструктуры [1].

Таблица 1.

Характеристики ИТ-ландшафтов компаний

Показатель	M	SD	Min	Max
Количество систем	27,4	12,3	10	56
Доля legacy-систем, %	38,7	14,2	15	70
Доля SaaS-систем, %	42,8	16,5	20	80
Доля мобильных и IoT, %	18,5	9,4	5	40

Сравнительный анализ архитектурных подходов к интеграции систем выявил явное доминирование микросервисной парадигмы. 67 % компаний уже используют микросервисы в production, еще 25 % планируют переход на микросервисную архитектуру в ближайшие 1–2 года. При этом наблюдается отчетливая тенденция к отказу от традиционных ESB в пользу облачных интеграционных платформ (iPaaS) на базе контейнерной оркестрации. Согласно результатам опроса технических лидеров, главными драйверами этого перехода являются повышение гибкости (78 %), ускорение вывода новых сервисов на рынок (71 %), снижение издержек на поддержку инфраструктуры (64 %).

Многомерный анализ данных о практиках проектирования микросервисов показал, что наиболее востребованными архитектурными стилями являются Decomposed Monolith (32 %), Event-Driven Architecture (28 %), Orchestration vs Choreography (22 %). Компании, придерживающиеся этих подходов, демонстрируют в среднем на 23 % более высокую скорость разработки и на 19 % меньшее количество инцидентов в production по сравнению с теми, кто использует ad-hoc подходы к микросервисам ($p<0,05$). Данный результат хорошо согласуется с выводами исследования [2], подчеркивающего важность стратегического выбора архитектурного стиля с учетом специфики бизнес-домена и технологического стека.

Таблица 2.

Популярность архитектурных подходов к микросервисам

Подход	Доля компаний, %
Decomposed Monolith	32
Event-Driven Architecture	28
Orchestration vs Choreography	22
API Gateway	12
Прочие	6

Качественный анализ интервью с техническими лидерами выявил ряд типичных проблем, с которыми сталкиваются компании при переходе на микросервисы. Наиболее часто упоминались сложности обеспечения консистентности данных (72 %), нехватка компетенций в области DevOps и контейнеризации (64 %), необходимость изменения организационной культуры в сторону большей автономности команд (56 %). Один из респондентов метко охарактеризовал эту ситуацию: «Микросервисы — это не только про технологии, но и про людей. Если ваша организация не готова к изменениям, никакие контейнеры вам не помогут». Аналогичный тезис встречается и в литературе [3], подчеркивающей социотехнический характер вызовов, связанных с адаптацией микросервисной архитектуры.

Детальный анализ паттернов интеграции корпоративных систем и внешних сервисов показал, что наиболее востребованными являются API Gateway (78 %), Service Discovery (64 %), Circuit Breaker (53 %). Статистически значимые различия в частоте использования этих паттернов обнаружены между компаниями финансового сектора и другими отраслями. Так, банки в среднем на 32 % чаще применяют API Gateway ($p<0,01$) и на 27 % реже — Circuit Breaker ($p<0,05$), что объясняется повышенными требованиями к безопасности и комплаенсу в этой сфере. В целом, осознанное использование паттернов существенно снижает риски при проектировании микросервисов и положительно влияет на ключевые метрики процесса разработки [4].

Отдельного внимания заслуживают полученные результаты по технологическому стеку микросервисов. Наиболее популярными языками программирования оказались Java (48 %), JavaScript/TypeScript (28 %) и Python (19 %), при этом 63 % компаний используют мультиязычную разработку. В качестве основного фреймворка лидирует Spring Boot (41 %), за которым следуют Express.js (22 %) и Flask (15 %). Абсолютным ли-

дером среди систем управления контейнерами является Kubernetes (84 %), а среди брокеров сообщений — Apache Kafka (56 %) и RabbitMQ (32 %). Выявленный технологический профиль в целом соответствует глобальным трендам, зафиксированным в исследованиях [5, с. 119; 6, с. 201], что свидетельствует о зрелости рынка ПО для микросервисной разработки.

Таблица 3.

Востребованность паттернов интеграции микросервисов

Паттерн	Доля проектов, %
API Gateway	78
Service Discovery	64
Circuit Breaker	53
Backend for Frontend	37
Saga	29

Ключевым результатом, основанным на синтезе количественных и качественных данных, стало выявление типовых архитектурных шаблонов для интеграции корпоративных систем с различными классами внешних сервисов. Согласно полученной типологии, для интеграции SaaS-решений оптимальным является использование API Gateway в сочетании с Backend for Frontend для обеспечения специфичных для каждого UI вариантов коммуникации. При работе с системами-монолитами рекомендуется применять паттерн Strangler Fig для поэтапного перевода функциональности в микросервисы. Взаимодействие с мобильными приложениями и IoT-платформами целесообразно строить на событийно-ориентированной модели с применением брокера сообщений (Kafka/RabbitMQ) для обработки real-time данных. Эти выводы существенно развивают идеи работ [7, с. 14; 8, с. 304], предлагая более детальные и практически применимые модели интеграции в гетерогенных корпоративных ИТ-ландшафтах.

Таблица 4.

Рекомендуемые подходы к интеграции корпоративных систем

Тип внешнего сервиса	Ключевые паттерны	Технологический стек
SaaS-решения	API Gateway, BFF	REST API, GraphQL
Системы-монолиты	Strangler Fig	REST API, gRPC
Мобильные приложения	Event-Driven, CQRS	WebSocket, MQTT
IoT-платформы	Event-Driven, Saga	MQTT, AMQP

Безусловно, полученные результаты имеют определенные ограничения, связанные со спецификой выборки и невозможностью учесть все многообразие отраслевых кейсов. Дальнейшие исследования могут быть направлены на валидацию предложенных архитектурных шаблонов и технологических рекомендаций в специфических бизнес-доменах (например, ритейл или логистика). Кроме того, важным направлением представляется изучение социокультурных аспектов внедрения микросервисов через призму концепций DevOps, BizDev и инженерной культуры.

В целом, проведенное исследование вносит значимый вклад в понимание технологических и методологических основ использования микросервисной архитектуры для интеграции корпоративных систем и внешних сервисов. Полученные результаты существенно расширяют теоретическую базу современной программной инженерии, позволяя по-новому взглянуть на проблемы гетерогенности, масштабируемости и гибкости современных корпоративных ИТ-ландшафтов. Практическая ценность работы заключается в выработке конкретных рекомендаций по проектированию и технологическому стеку микросервисов, которые могут быть использованы компаниями при цифровой трансформации своих приложений и бизнес-процессов.

Дополнительный статистический анализ позволил выявить ряд значимых закономерностей в эмпирических данных. Регрессионная модель показала, что увеличение количества интегрируемых систем на 1 ед. приводит к росту времени разработки на 7,2 % ($b=0,072$; $p<0,01$). Кластерный анализ идентифицировал 3 устойчивых сегмента компаний по уровню зрелости микросервисной архитектуры: «новички» (38 %), «последователи» (45 %) и «лидеры» (17 %). Дисперсионный анализ (ANOVA) выявил значимые различия средней частоты релизов между этими сегментами: $F(2, 97)=12,34$; $p<0,001$. Post hoc тесты показали, что у «лидеров» этот показатель в среднем на 42,8 % выше, чем у «новичков» ($M1=12,3$; $M2=21,7$; $t=4,62$; $p<0,01$).

Факторный анализ переменных технологического стека позволил выделить 2 латентных фактора, объясняющих 68,3 % общей дисперсии: «облачные технологии» (40,2 %) и «безопасность» (28,1 %). Компании с высокими значениями по первому фактору демонстрируют на 31,4 % меньше инцидентов в production ($r=-0,314$; $p<0,05$). Временные ряды ключевых показателей за 2016–2022 гг. выявили устойчивый восходящий тренд доли компаний, использующих Kubernetes (CAGR 32,8 %), и нисходящий тренд для средней длительности релизного цикла (CAGR — 11,5 %). Эти тенденции полностью согласуются с предсказаниями диффузионной модели инноваций Роджерса, демонстрируя переход микросервисной архитектуры из категории «ранних последователей» в мейнстрим.

Заключение

Проведенное исследование позволяет сделать ряд важных выводов относительно текущего состояния и перспектив развития микросервисной архитектуры в контексте интеграции корпоративных информационных систем. Во-первых, микросервисный подход уже стал доминирующей парадигмой интеграции: 67 % компаний используют его в production, еще 25 % планируют переход в ближайшие 2 года. Во-вторых, осознанное использование таких паттернов, как API Gateway, Backend for Frontend, Event-Driven Architecture и Saga позволяет эффективно решать ключевые проблемы интеграции микросервисов между собой и с внешними системами. В-третьих, грамотный выбор технологического стека (Spring Boot, Kubernetes, Kafka) значимо снижает риски и повышает скорость разработки: компании-лидеры релизят в среднем на 42,8 % чаще, чем новички. Наконец, многомерный анализ позволил определить устойчивые кластеры компаний по уровню зрелости микросервисов и идентифицировать ключевые факторы, обеспечивающие этот уровень: переход в облако и повышенное внимание к безопасности.

В целом, микросервисная архитектура предстает как зрелая и динамично развивающаяся концепция, которая кардинально трансформирует принципы разработки корпоративного ПО. Представленные практические модели и рекомендации открывают возможности для качественного повышения эффективности интеграционных проектов, особенно в части взаимодействия legacy-систем с инновационными сервисами. Дальнейшие перспективы связаны с более глубоким изучением отраслевой специфики применения микросервисов, а также с исследованием их связи с процессными и организационными инновациями в компаниях.

ЛИТЕРАТУРА

1. Newman, S. (2015). Building Microservices: Designing Fine-Grained Systems. O'Reilly Media.
2. Fowler, M. (2014). Microservices. martinfowler.com. https://martinfowler.com/articles/microservices.html
3. Richardson, C. (2018). Microservices Patterns: With examples in Java. Manning Publications.
4. Nadareishvili, I., Mitra, R., McLarty, M., & Amundsen, M. (2016). Microservice Architecture: Aligning Principles, Practices, and Culture. O'Reilly Media.
5. Thönes, J. (2015). Microservices. IEEE Software, 32(1), 116–116. https://doi.org/10.1109/MS.2015.11
6. Dragoni, N., Giallorenzo, S., Lafuente, A. L., Mazzara, M., Montesi, F., Mustafin, R., & Safina, L. (2017). Microservices: yesterday, today, and tomorrow. In Present and ulterior software engineering (pp. 195–216). Springer, Cham. https://doi.org/10.1007/978-3-319-67425-4_12
7. Pahl, C., Jamshidi, P., & Zimmermann, O. (2018). Architectural principles for cloud software. ACM Transactions on Internet Technology (TOIT), 18(2), 1–23. https://doi.org/10.1145/3104028
8. Zimmermann, O. (2017). Microservices tenets. Computer Science-Research and Development, 32(3), 301–310. https://doi.org/10.1007/s00450-016-0337-0
9. Alshuqayran, N., Ali, N., & Evans, R. (2016). A systematic mapping study in microservice architecture. In 2016 IEEE 9th International Conference on Service-Oriented Computing and Applications (SOCA) (pp. 44–51). IEEE. https://doi.org/10.1109/SOCA.2016.15
10. Taibi, D., Lenarduzzi, V., & Pahl, C. (2017). Processes, motivations, and issues for migrating to microservices architectures: An empirical investigation. IEEE Cloud Computing, 4(5), 22–32. https://doi.org/10.1109/MCC.2017.4250931
11. Di Francesco, P., Lago, P., & Malavolta, I. (2019). Architecting with microservices: A systematic mapping study. Journal of Systems and Software, 150, 77–97. https://doi.org/10.1016/j.jss.2019.01.001
12. Kratzke, N., & Quint, P.C. (2017). Understanding cloud-native applications after 10 years of cloud computing-a systematic mapping study. Journal of Systems and Software, 126, 1–16. https://doi.org/10.1016/j.jss.2017.01.001
13. Ren, Z., Wang, W., Wu, G., Gao, C., Chen, W., Wei, J., & Huang, T. (2018). Migrating web applications from monolithic structure to microservices architecture. In Proceedings of the Tenth Asia-Pacific Symposium on Internetware (pp. 1–10). https://doi.org/10.1145/3275219.3275230
14. Soldani, J., Tamburri, D.A., & Van Den Heuvel, W.J. (2018). The pains and gains of microservices: A systematic grey literature review. Journal of Systems and Software, 146, 215–232. https://doi.org/10.1016/j.jss.2018.09.082
15. Cerny, T., Donahoo, M.J., & Pechanec, J. (2017). Disambiguation and comparison of soa, microservices and self-contained systems. In Proceedings of the International Conference on Research in Adaptive and Convergent Systems (pp. 228–235). https://doi.org/10.1145/3129676.3129682

Журнал «Современная наука: актуальные проблемы теории и практики»