Мультиагентная система в парадигме AIOps: уровневая архитектура анализа системной телеметрии

Д. И. Пушкарев

Санкт-Петербургский государственный электротехнический университет «ЛЭТИ» им. В.И. Ульянова (Ленина)

dipushkarev@stud.etu.ru

Аннотация. **Рассматривается интеграция больших языковых моделей и мультиагентных систем в парадигме AIOps (Artificial intelligence for IT Operations, искусственный интеллект для ИТ-операций) для автоматизации процесса анализа системной телеметрии и выявления причин сбоев. Представлена декомпозированная архитектура системы анализа, включающая этапы от сбора базовых метрик до интеллектуального вывода с использованием сети взаимодействующих ИИ-агентов. Анализируются методы локализации первопричин с помощью цепочек рассуждений, применение технологии генерации с дополнительным поиском (Retrieval-Augmented Generation, RAG) для снижения риска галлюцинаций, а также протоколы межагентного взаимодействия и направления контроля и безопасности. Акцент сделан на преодолении ограничений одиночных моделей за счет распределения задач между специализированными агентами, что способствует повышению надежности диагностики и переходу к автономному самовосстановлению ИТ-инфраструктур.**

Ключевые слова: системная телеметрия; AIOps; большие языковые модели; мультиагентные системы; модель OSI; Agent-OSI; AgentOps

I. Введение

Современные ИТ-инфраструктуры характеризуются высокой степенью распределённости, многообразием микросервисных архитектур и экспоненциальным ростом объёмов генерируемой телеметрии — информационных сообщений и результатов измерений различных метрик. По данным обзора ACM Computing Surveys [1], охватившего 183 исследования за период 2020–2024 гг., парадигма AIOps (Artificial Intelligence for IT Operations) становится ключевым направлением автоматизации операционных задач в мониторинге ИТ-инфраструктуры, включая обнаружение разнообразных аномалий в ИТ-инфраструктуре, локализацию первопричин сбоев и автоматическое восстановление.

Системная телеметрия представляет собой совокупность разнородных данных, генерируемых компонентами ИТ-инфраструктуры. К основным типам телеметрии относятся: текстовые записи о событиях системы, метрики (числовые временные ряды, характеризующие состояние ресурсов), трейсы (распределённые цепочки вызовов между сервисами) и предупреждения (уведомления о превышении пороговых значений). Данные типы информации существенно различаются по формату, структуре и семантике: логи являются неструктурированными текстовыми потоками, метрики представлены в виде числовых рядов, а трейсы формируют направленные ациклические графы вызовов. Подобная неоднородность и разнообразие телеметрических данных значительно усложняют их совместную обработку и анализ, что обусловливает необходимость специализированных подходов к каждому типу данных. Существующие методы AIOps основаны на раздельном использовании моделей машинного обучения, каждая из которых нацелена на отдельную задачу: классификацию записей телеметрии, пороговое отслеживание метрик или выявление зависимостей в распределённых цепочках вызовов. Такое разделение ведёт к потере смыслового контекста между этапами диагностики. Большие языковые модели (Large Language Model, LLM) расширили возможности обработки неструктурированной телеметрии, но одиночная модель или одноагентная система подвержена генерации ошибочных выводов и стеснена объёмом контекстного окна — количеством данных, принимаемых за одно обращение [2, 3].

Мультиагентная архитектура позволяет разбить задачу анализа телеметрии на частные подзадачи, закреплённые за отдельными агентами, что позволит увеличить точность и скорость в 1,5 раза по доле успешно устранённых сбоев [4] за счёт трёх факторов: чёткой функциональной специализации агентов, архитектурной расширяемости и упрощённого контроля безопасности.

Вместе с тем возникает вопрос о системном проектировании подобных архитектур. В частности, требуется определить принципы взаимодействия агентов, механизмы координации их совместной работы и стратегии обеспечения отказоустойчивости при масштабировании системы.

В области компьютерных сетей эталонная модель OSI (Open Systems Interconnection) успешно обеспечила стандартизацию за счёт уровневой декомпозиции сетевого взаимодействия. В 2025–2026 гг. в ряде исследований предлагается использовать аналогичный подход к представлению архитектуры ИИ-агентов: работа Agent-OSI [5] описывает шестиуровневый стек протоколов для децентрализованного взаимодействия агентов, а концепция «Layer 8» [6] рассматривает агентный ИИ как надстройку над традиционной моделью OSI.

Целью настоящего исследования является разработка многоуровневой архитектуры

мультиагентной системы анализа телеметрии, основанной на структуре модели OSI, где каждый агент отвечает за анализ и классификацию телеметрических данных на своём уровне, передавая контекст другому агенту.

II. Уровневая архитектура мультиагентной системы анализа телеметрии на основе модели OSI

Предлагаемая архитектура мультиагентной системы анализа телеметрии построена по принципу уровневой декомпозиции, аналогичному эталонной модели OSI. Каждый уровень инкапсулирует определённую функциональность и предоставляет интерфейсы для смежных уровней, что обеспечивает модульность, заменяемость компонентов и возможность параллельной разработки. Это позволяет эффективно обогащать контекст на каждом уровне обработки данных, передаваемых между агентами. Такая архитектура особенно результативна при долгосрочной эксплуатации: по мере накопления исторических данных временных рядов повышается точность первоначального обучения и последующего дообучения моделей, а при появлении новых типов данных система легко расширяется без перестройки существующих компонентов. Уровневая модель является наиболее подходящей для решения данной задачи, т.к. каждый уровень будет отвечать за работу со своим типом данных и операция, а также дополнительно валидироваться последующим, уровневая модель представлена на рис. 1.

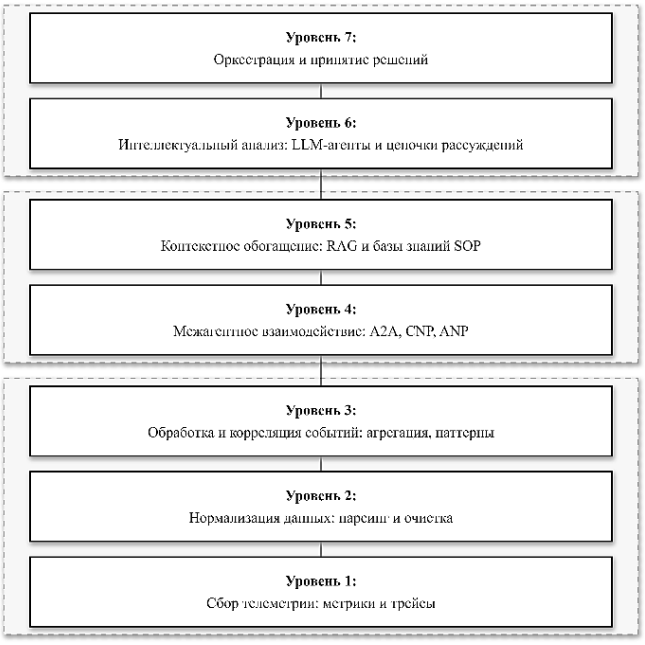

Рис. 1. Предлагаемая многоуровневая архитектура

Каждый уровень модели — агент, работающий с отдельным типом данных, ранее обученный на данных этого типа. В табл. I представлено описание соответствия уровня моделей и их функции.

Уровень 1 (Сбор телеметрии) является фундаментом архитектуры и обеспечивает унифицированный приём данных из гетерогенных источников. По аналогии с физическим уровнем OSI, данный уровень абстрагирует транспортные механизмы и предоставляет верхним уровням единообразный поток необработанных данных. Стандарт OpenTelemetry [7] выступает в качестве основного протокола данного уровня, обеспечивая сбор трёх типов телеметрии: системных метрик (числовые временные ряды), текстовых сообщений системы и трейсов (распределённых цепочек вызовов) в формате, необходимом для работы классифицирующих агентов, расположенных на вышестоящих уровнях архитектуры.

ТАБЛИЦА I. Описание уровней архитектуры Artificial Intelligence for IT Operations

Уровень OSI	Уровень системы	Функции
L7 Application	Оркестрация и принятие решений	Координация агентов, выдача рекомендаций
L6 Presentation	Интеллектуальный анализ	CoT-рассуждения, классификация причин
L5 Session	Контекстное обогащение	Извлечение знаний, сопоставление с историей
L4 Transport	Межагентное взаимодействие	Маршрутизация задач, протоколы обмена
L3 Network	Обработка и корреляция событий	Агрегация данных, выявление паттернов
L2 Data Link	Нормализация данных	Парсинг, структури-рование, дедупликация
L1 Physical	Сбор телеметрии	Приём логов, метрик, трейсов, алертов

Уровень 2 (Нормализация данных) выполняет структурирование и очистку сырых телеметрических данных. Аналогично канальному уровню OSI, обеспечивающему надёжную передачу кадров, данный уровень гарантирует целостность и единообразие представления данных для последующей обработки. На 2 уровне осуществляются парсинг неструктурированных логов, а также приведение временных меток информационных сообщений к единой временной шкале к единой временной шкале.

Уровень 3 (Обработка и корреляция событий) агрегирует нормализованные данные и выявляет пространственно-временные зависимости между событиями из различных источников. По аналогии с сетевым уровнем OSI, осуществляющим маршрутизацию пакетов, данный уровень определяет «маршруты» распространения аномалий через компоненты инфраструктуры, используя графовые алгоритмы и механизмы комплексной обработки событий (Complex Event Processing, CEP).

Уровень 4 (Межагентное взаимодействие) определяет протоколы и механизмы коммуникации между агентами, аналогично транспортному уровню OSI, обеспечивающему надёжную передачу сообщений. Данный уровень управляет маршрутизацией задач между агентами, контролем доставки результатов и балансировкой нагрузки. Среди протоколов межагентного взаимодействия выделяются: Contract Net Protocol (CNP) для конкурентного распределения задач, протокол Agent-to-Agent (A2A) и Agent Network Protocol (ANP) [5, 8].

Уровень 5 (Контекстное обогащение) реализует механизм Retrieval-Augmented Generation (RAG), обеспечивая доступ агентов к внешним базам знаний. По аналогии с сеансовым уровнем OSI, управляющим контекстом соединений, данный уровень поддерживает контекст диагностического сеанса, извлекая релевантную информацию из баз стандартных операционных процедур (Standard Operating Procedure,

SOP) и истории инцидентов. Как показано в работе [3], без RAG-компонента показатель F1-score падает до 0.295 на датасете облачной платформы, что подтверждает критическую роль контекстного обогащения в агентных системах.

Уровень 6 (Интеллектуальный анализ) содержит ядро системы — LLM-агентов, выполняющих рассуждения посредством цепочек мыслей (Chain-of-Thought, CoT). Аналогично уровню представления OSI, данный уровень преобразует структурированные данные в семантическое представление, пригодное для принятия решений. Система Flow-of-Action [2] демонстрирует, что управление рассуждениями через SOP-процедуры существенно снижает вероятность галлюцинаций.

Уровень 7 (Оркестрация и принятие решений) координирует работу всех агентов и формирует итоговые рекомендации. По аналогии с прикладным уровнем OSI, данный уровень предоставляет интерфейс конечному пользователю (инженеру по обеспечению надёжности — Site Reliability Engineer, SRE) и управляет жизненным циклом инцидента — от его обнаружения до рекомендации по устранению или автоматического восстановления.

III. Анализ и построение архитектуры мультиагентной системы

На основе описанной уровневой модели предлагается компонентная архитектура мультиагентной системы, представленная на рис. 2.

Система состоит из центрального оркестратора и четырёх специализированных агентов, взаимодействующих через протоколы.

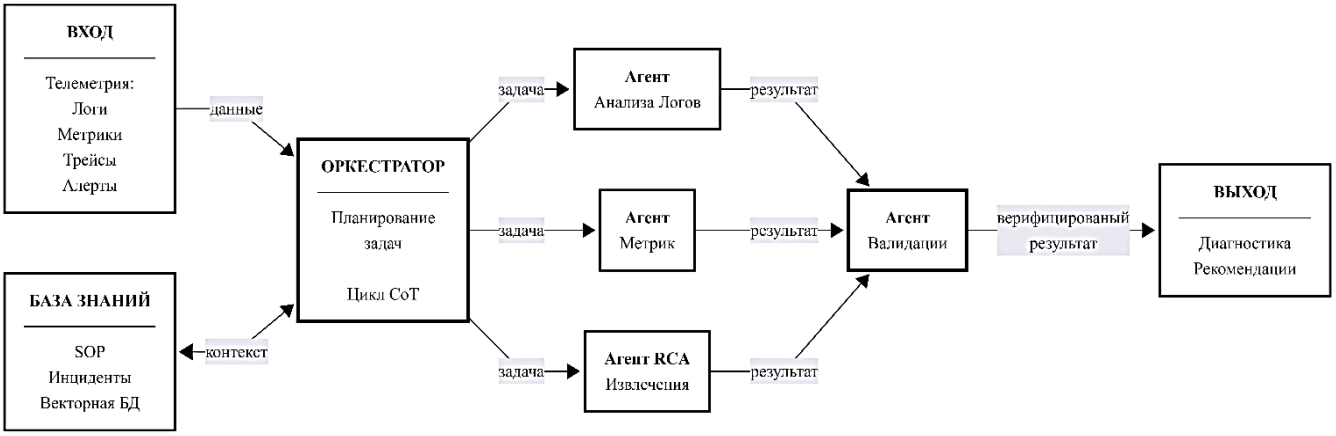

Рис. 2. Архитектура мультиагентной системы AIOps с оркестратором и специализированными агентами

Функции компонентов системы:

- оркестратор осуществляет декомпозицию входящей задачи и распределяет подзадачи между агентами с использованием цикла Chain-of-Thought (CoT);
- агент анализа логов специализируется на парсинге и семантическом анализе текстовых записей;
- агент метрик обрабатывает числовые временные ряды и выявляет статистические аномалии;
- агент валидации верифицирует гипотезы других агентов, снижая число ложноположительных срабатываний;
- агент RAG/извлечения обеспечивает доступ к базе знаний, включающей SOP-процедуры, историю инцидентов и векторную базу данных.

Подобный декомпозиционный подход позволяет преодолеть ограничения одноагентных систем, что подтверждается результатами эмпирических исследований. В тестах OpenRCA [9] (ICLR 2025), охватывающих 335 реальных сбоев и 68 ГБ телеметрии, лучшая одноагентная модель устранила лишь 11.34% инцидентов, что свидетельствует о необходимости внедрения мультиагентного подхода. Например, мультиагентная система для анализа первопричин (Multi-Agent Framework for Root Cause Analysis, MA-RCA) продемонстрировала критичность роли каждого компонента мультиагентной архитектуры и показала, что исключение валидирующего агента вызывает существенное увеличение доли ложноположительных срабатываний, тогда как удаление агента извлечения приводит к падению метрики F1 до значения 0,144 на датасете энергетической инфраструктуры [3].

Вопрос адаптации эталонной модели OSI к проектированию агентных систем находит отражение в ряде современных исследований. В работе [5] предложен шестиуровневый стек протоколов для децентрализованных сетей агентов, охватывающий защищённый обмен сообщениями, децентрализованную идентификацию, тарификацию, верифицируемое исполнение, семантическую интероперабельность и оркестрацию. Концепция «Layer 8» [6] трактует агентный ИИ как дополнительный уровень над классической семиуровневой моделью, обосновывая это принципиальным различием: прикладной уровень реализует детерминированные операции, тогда как ИИ-агенты осуществляют автономное недетерминированное принятие решений с адаптацией поведения на основе контекста. Эта граница отделяет классическую сетевую архитектуру от агентной.

В разработанной архитектуре уровневый подход OSI применяется не напрямую, а как методологическая рамка проектирования. Аналогия с OSI позволяет:

- обеспечить plug-and-play замену компонентов (например, замену LLM-модели на уровне 6 без изменения протоколов уровня 4);
- формализовать интерфейсы между уровнями для обеспечения интероперабельности;

- применить опыт обеспечения безопасности сетевых архитектур к контролю агентных систем, включая защиту от манипуляции телеметрией [10].

В табл. II представлено сравнение предлагаемой уровневой модели Artificial intelligence for IT Operations и модели OSI по нескольким критериям.

ТАБЛИЦА II. СРАВНЕНИЕ МОДЕЛИ OSI И УРОВНЕВОЙ МОДЕЛИ ARTIFICIAL INTELLIGENCE FOR IT OPERATIONS

Критерий	**Модель OSI (сетевая)**	**Уровневая модель AIOps**
Число уровней	7 фиксированных	7 (расширяемо до 8+ с учётом Layer 8)
Тип данных	Детерминированные пакеты	Мультимодальная телеметрия + семантика
Взаимодействие	Последовательная инкапсуляция	Иерархическое + горизонтальное (A2A)
Детерминизм	Полностью детерминированное	Недетерминированное (LLM-рассуждения)

Ключевым отличием архитектуры AIOps является совмещение вертикальной декомпозиции с горизонтальным взаимодействием. В системах STRATUS [4] и MA-RCA [3] агенты организованы в линейный конвейер или звёздообразную топологию, тогда как в представленной многоуровневой модели уровни 3 и 6 обмениваются данными напрямую через уровень 4, аналогично туннелированию в сетевых протоколах.

IV. РАСПРЕДЕЛЕННАЯ ТРАССИРОВКА И АЛЬТЕРНАТИВНЫЕ ПОДХОДЫ К НАБЛЮДАЕМОСТИ

Параллельно развитию систем AIOps происходило развитие систем отслеживания и трассировки событий. Значительное распространение получил подход распределённой трассировки (distributed tracing) — сквозное отслеживание запросов через цепочку микросервисов и систем. Совместно с метриками и логами трассировка формирует три столпа наблюдаемости (observability) — способности системы предоставлять информацию о своём внутреннем состоянии на основе анализа её внешних выходных данных [11], унифицированных стандартом OpenTelemetry. Одной из подобных систем является Zipkin. Основанная на принципах Dapper, Zipkin предоставляет широкий инструментарий для работы с идентификаторами трассировки (trace ID). Однако подход distributed tracing имеет ряд ограничений:

- трассировка фиксирует лишь структуру вызовов без семантики ошибок;
- при каскадных сбоях граф может содержать тысячи действий внутри системы, увеличивая среднее время восстановления (Mean Time To Recovery, MTTR);
- графовые алгоритмы ранжирования (PageRank, random walk) не учитывают контекст логов и метрик.

Разработанная модель AIOps принципиально отличается от distributed tracing: трассировка реализует пассивную наблюдаемость, тогда как мультиагентная архитектура обеспечивает активную диагностику с верификацией гипотез. Для AIOps: на уровне 1 события трассировки являются одним из типов входных телеметрических данных наряду с информационными сообщениями, метриками и оповещениями, на уровне 3 корреляционный агент объединяет все источники; уровень 6 применяет LLM для семантической интерпретации телеметрических данных.

Предлагаемая архитектура является надстройкой над инфраструктурой наблюдаемости: трассировка остаётся источником данных на уровне 1, а агенты верхних уровней обогащают и интерпретируют её, обеспечивая переход к проактивной диагностике [4].

Вопрос безопасности является одним из ключевых для работы моделей. В работе [10] продемонстрировано, что LLM-системы подвержены целенаправленным манипуляциям с телеметрией, что создаёт риски при принятии решений и увеличение случаев ошибочных диагностических заключений. В модели AIOps уровень 2 фильтрует аномальные данные, агент валидации верифицирует рекомендации, а спецификация TNR [4] и принципы нулевого доверия обеспечивают контроль полномочий агентов.

V. ЗАКЛЮЧЕНИЕ

В статье представлена многоуровневая архитектура мультиагентной системы анализа телеметрии разработанная на основе модели OSI, обеспечивающая модульность и формализацию интерфейсов между агентами, выполняющими специализированные функции анализа, нормализации и корреляции данных на каждом уровне обработки телеметрии.

Сопоставление с distributed tracing показало комплементарность подходов: трассировка обеспечивает пассивную наблюдаемость и является источником данных для агентов, а семантическая интерпретация на верхних уровнях преодолевает ограничения изолированного анализа.

Модель OSI применима, но требует адаптаций: учёта недетерминизма LLM, горизонтальных взаимодействий и контроля галлюцинаций через RAG. Дальнейшие исследования включают валидацию на бенчмарках AIOpsLab и OpenRCA, масштабирование числа агентов и интеграцию протоколов нулевого доверия.

СПИСОК ЛИТЕРАТУРЫ

[1] Zhang L. et al. A Survey of AIOps in the Era of Large Language Models // ACM Comput. Surv. 2025. DOI: 10.1145/3746635.

[2] Pei C. et al. Flow-of-Action: SOP Enhanced LLM-Based Multi-Agent System for Root Cause Analysis // WWW Companion. 2025. DOI: 10.1145/3701716.3715225.

[3] Fu F. et al. Leveraging Multi-Agent Framework for Root Cause Analysis // Complex Intell. Syst. 2025. DOI: 10.1007/s40747-025-02096-0.

[4] Chen Y. et al. STRATUS: A Multi-agent System for Autonomous Reliability Engineering of Modern Clouds // NeurIPS. 2025.

[5] Xu W. et al. Agent-OSI: A Layered Protocol Stack Toward a Decentralized Internet of Agents. 2026. arXiv:2602.13795.

[6] Huang K. Is Agentic AI Layer 8? // Agentic AI Substack. 2026.

[7] OpenTelemetry Specification. CNCF, 2024. URL: https://opentelemetry.io/docs/specs/.

[8] Derouiche H. et al. Agentic AI Frameworks: Architectures, Protocols, and Design Challenges. 2025. arXiv:2508.10146.

[9] Xu J. et al. OpenRCA: Can Large Language Models Locate the Root Cause of Software Failures? // ICLR. 2025.

[10] Pasquini D. et al. When AIOps Become "AI Oops": Subverting LLM-driven IT Operations via Telemetry Manipulation. 2025. arXiv:2508.06394.

[11] Faseeha U. et al. Observability in Microservices: An In-Depth Exploration of Frameworks, Challenges, and Deployment Paradigms // IEEE Access. 2025. DOI: 10.1109/ACCESS.2025.3562125.