Сомов Никита Ильич
студент
РТУ МИРЭА

ПАТТЕРНЫ МНОГОАГЕНТНОГО ВЗАИМОДЕЙСТВИЯ В ИНФОРМАЦИОННЫХ СИСТЕМАХ: ЦЕНТРАЛИЗОВАННАЯ И ДЕЦЕНТРАЛИЗОВАННАЯ КООРДИНАЦИЯ

Ключевые слова: мультиагентная система, координация агентов, паттерны взаимодействия, централизованная координация, децентрализованная координация, оркестрация, Supervisor, Pipeline, Blackboard, Stigmergy, эмерджентность, Peer-to-peer, большие языковые модели.

Аннотация

В статье исследуются паттерны многоагентного взаимодействия в информационных системах с фокусом на двух классах координации – централизованной и децентрализованной. Приведены формальное определение агента и мультиагентной системы, ключевые свойства агента (автономность, реактивность, проактивность, социальность) и базовое понятие координации как механизма обеспечения согласованного поведения в условиях отсутствия разделяемого состояния. Представлен обзор современных паттернов: для класса централизованной координации детально описаны механизмы работы Orchestrator/Supervisor, Pipeline и Planner-Executor; для класса децентрализованной координации – паттерны Peer-to-peer, Blackboard и Stigmergy. Проведён сравнительный анализ двух классов: выделены преимущества централизованных решений (предсказуемость, прозрачность, удобство отладки) и их ограничения, связанные с наличием единой точки отказа и ограниченной масштабируемостью, а также достоинства децентрализованных паттернов (горизонтальная масштабируемость, устойчивость к частичным отказам) и присущая им

сложность верификации поведения. Показано, что выбор между классами определяется компромиссом между управляемостью системы и её масштабируемостью. Сделан вывод о перспективах развития гибридных архитектур, сочетающих преимущества обоих классов.

Введение

Рост сложности современных информационных систем и широкое распространение технологий больших языковых моделей обусловили стремительное развитие мультиагентного подхода к автоматизации интеллектуальных задач. Под мультиагентной системой понимается совокупность автономных программных агентов, взаимодействующих посредством обмена сообщениями или через общую среду и совместно решающих задачи, превосходящие возможности отдельного компонента. Необходимость в таком подходе продиктована практическими требованиями: от автоматизации анализа документации и поддержки принятия решений до оркестрации сложных бизнес-процессов в распределённых корпоративных системах. При этом проектирование МАС осуществляется в условиях фундаментального противоречия между управляемостью системы и её масштабируемостью, что исключает универсальное архитектурное решение и требует осознанного выбора паттерна координации под конкретные требования.

Исторически развитие подходов к координации агентов прошло путь от теоретических моделей к инженерным реализациям. Первые формальные модели МАС, разработанные в 1980–90-х годах, опирались преимущественно на децентрализованные механизмы – доски объявлений (Blackboard) и стигмергические модели, вдохновлённые биологией роевого интеллекта. С развитием корпоративных распределённых систем на первый план вышли централизованные архитектуры с явным оркестратором, обеспечивающие предсказуемость и трассируемость исполнения. Новый виток интереса к обоим классам паттернов наступил с появлением LLM-агентов:

оркестраторные паттерны Supervisor и Planner-Executor стали стандартом в таких фреймворках, как LangGraph и CrewAI, тогда как исследования в области масштабируемых систем вновь обратились к децентрализованным подходам.

Общепринятая классификация разделяет паттерны координации МАС на два принципиальных класса. Первый – централизованные паттерны, в которых управление сосредоточено в едином агенте-оркестраторе, декомпозирующем задачи и направляющем работу специализированных исполнителей. Характерными представителями этого класса являются паттерны Orchestrator/Supervisor, Pipeline и Planner-Executor; они обеспечивают высокую предсказуемость ценой уязвимости к отказу управляющего узла. Второй класс – децентрализованные паттерны, в которых согласованное поведение системы возникает из локальных взаимодействий агентов без единого управляющего элемента. Ключевыми представителями здесь выступают паттерны Peer-to-peer, Blackboard и Stigmergy, обеспечивающие горизонтальную масштабируемость и устойчивость к частичным отказам ценой усложнения верификации поведения.

Несмотря на значительный объём публикаций, посвящённых отдельным паттернам и фреймворкам, наблюдается дефицит систематизированных обзоров, одновременно охватывающих оба класса координации, прослеживающих эволюцию их ключевых идей и формулирующих критерии рационального выбора паттерна в зависимости от требований целевой системы. В существующих работах анализ, как правило, ограничивается либо одним классом, либо конкретным фреймворком, что затрудняет формирование целостного взгляда на проблематику координации в мультиагентных системах.

Целью настоящей статьи является систематический анализ паттернов многоагентного взаимодействия в информационных системах в разрезе двух классов координации – централизованной и децентрализованной.

Определения и основные понятия

Под агентом в контексте информационных систем понимается программная сущность, способная автономно воспринимать состояние своей среды, принимать решения на основе внутреннего состояния и целей, а также исполнять действия, изменяющие эту среду. Теоретики выделяют четыре ключевых свойства агента:

— *Автономность* – агент функционирует без непосредственного вмешательства оператора и самостоятельно контролирует своё поведение.

— *Реактивность* – способность своевременно реагировать на изменения в среде: входящие события, обновление данных, сообщения от других агентов.

— *Проактивность* – целеориентированное поведение: агент не только отвечает на внешние стимулы, но и самостоятельно инициирует действия для достижения заданной цели.

— *Социальность* – способность взаимодействовать с другими агентами: кооперироваться, вести переговоры или конкурировать за ресурсы.[1]

Мультиагентная система (МАС) представляет собой совокупность взаимодействующих агентов, совместно решающих задачи, которые превышают возможности или компетентность отдельного агента. В отличие от монолитных систем, где вся логика сосредоточена в единой точке, МАС допускает декомпозицию сложной задачи на независимые подзадачи, исполняемые специализированными агентами параллельно или последовательно. Это обеспечивает масштабируемость, модульность и возможность независимого развёртывания компонентов.

Центральной проблемой проектирования МАС является координация – механизм, обеспечивающий согласованное поведение агентов в условиях отсутствия разделяемой памяти и единого глобального состояния. Без явной координации агенты вступают в конфликты за общие ресурсы, формируют противоречивые планы или дублируют выполнение одних и тех же подзадач. Существующие подходы к координации делятся на два принципиальных

класса: централизованные, при которых управление сосредоточено в одном узле системы, и децентрализованные, при которых согласованное поведение возникает из локальных взаимодействий агентов без единого управляющего элемента.[2]

Централизованные паттерны координации

В централизованных архитектурах выделяется один управляющий агент, который обладает привилегированным знанием о состоянии всей системы, принимает решения о распределении задач и контролирует их исполнение. Паттерны этого класса отличаются предсказуемостью поведения и удобством отладки, однако привносят в систему единую точку отказа. Ключевые представители класса:

Orchestrator / Supervisor – наиболее распространённый паттерн. Выделенный агент-оркестратор получает задачу верхнего уровня, декомпозирует её на подзадачи и определяет, какой специализированный агент компетентен для решения каждой из них. Далее оркестратор последовательно или параллельно выдаёт агентам инструкции, получает их ответы и синтезирует промежуточные результаты. При возникновении ошибки он принимает решение о повторном вызове агента, переключении на альтернативного исполнителя или эскалации задачи на уровень выше. Ключевое преимущество паттерна – прозрачность: вся логика управления сосредоточена в одном месте, поведение системы легко верифицируется. Платой служит уязвимость к отказу оркестратора, который парализует систему вне зависимости от работоспособности подчинённых агентов.[3]

Pipeline – агенты выстраиваются в детерминированную цепочку, где результат работы каждого предыдущего агента служит входными данными для следующего. Маршрутизация статична: порядок вызовов фиксируется на этапе проектирования и не изменяется в процессе исполнения. Это делает паттерн предсказуемым и простым в реализации, однако лишает систему

гибкости – невозможно динамически изменить порядок шагов или пропустить ненужный этап в зависимости от промежуточных результатов.

Planner-Executor – паттерн, отделяющий планирование от исполнения на уровне архитектуры. Агент-планировщик строит подробный план решения задачи с явными зависимостями между шагами и передаёт его набору агентов-исполнителей, специализированных под конкретные типы операций. В современных системах на основе больших языковых моделей планировщик, как правило, реализуется на базе более мощной модели, способной к многошаговому рассуждению, тогда как исполнители могут использовать облегчённые модели или детерминированные инструменты. Разделение ролей позволяет оптимизировать стоимость вычислений и упрощает замену отдельных исполнителей без изменения логики планирования.[4]

Децентрализованные паттерны координации

Децентрализованные подходы устраняют зависимость от единого управляющего агента: каждый участник системы обладает автономией и взаимодействует с ограниченным числом соседей или с общей средой. Глобальная согласованность в таких системах возникает как эмерджентный эффект множества локальных взаимодействий, а не как результат централизованного управления. Ключевые представители класса:

Peer-to-peer (Mesh) – агенты взаимодействуют напрямую, без посредника-оркестратора. Каждый агент может инициировать коммуникацию с любым другим участником сети, передавая задачи или запрашивая результаты через механизм передачи управления. Отсутствие единого управляющего узла делает архитектуру устойчивой к отказам: при выходе из строя одного участника остальные продолжают функционировать, а система деградирует плавно. Вместе с тем распределённость логики управления существенно затрудняет отладку и мониторинг.[5]

Blackboard – координация организуется не через прямые сообщения между агентами, а через общее хранилище данных. Агенты независимо

читают актуальное состояние среды с «доски», выполняют свою работу и записывают результаты обратно. Другие агенты реагируют на появление нужных им данных и запускают собственные задачи. Таким образом, взаимодействие оказывается косвенным: агенты координируются через данные, а не через явные вызовы. Это упрощает добавление новых агентов – достаточно описать, какие данные агент потребляет и какие производит. Уязвимость паттерна состоит в том, что доска становится узким местом по производительности при высокой нагрузке.[6]

Stigmergy – наиболее радикальное выражение децентрализации, заимствованное из биологии роевого интеллекта. Агенты не обмениваются сообщениями напрямую: они оставляют в среде сигналы, аналогичные феромонным следам муравьёв, которые затухают со временем. Другие агенты воспринимают эти сигналы и корректируют своё поведение, не зная ничего о намерениях их источника. В инженерных реализациях сигналы представляются в виде типизированных записей с параметрами интенсивности и скорости затухания. Ключевым преимуществом стигмергии является отсутствие накладных расходов на маршрутизацию: добавление новых агентов не требует изменений в логике существующих участников. Недостатком служит непредсказуемость поведения системы в целом, которое приходится верифицировать через симуляции.[7]

Сравнительный анализ и выбор подхода

Выбор между централизованной и децентрализованной координацией определяется прежде всего требованиями конкретного приложения к надёжности, управляемости и масштабируемости.

Централизованные паттерны обладают следующими эксплуатационными характеристиками:

— высокая предсказуемость и прозрачность: логика принятия решений сосредоточена в оркестраторе, состояние системы легко воспроизводимо и поддаётся аудиту;

— простота отладки и верификации поведения;

— ограниченная масштабируемость: при росте числа агентов оркестратор превращается в вычислительное и коммуникационное узкое место;

— единая точка отказа: выход оркестратора из строя нарушает работу всей системы.

Децентрализованные паттерны, напротив, характеризуются следующим:

— горизонтальная масштабируемость: система линейно расширяется за счёт добавления новых агентов без изменения существующей логики;

— устойчивость к частичным отказам: система продолжает функционировать при выходе из строя отдельных участников, деградируя постепенно;

— сложность верификации поведения: эмерджентные паттерны взаимодействия трудно предусмотреть на этапе проектирования;

— непредсказуемость в нештатных ситуациях.[8]

Таким образом, централизованные подходы являются предпочтительным выбором для систем, где критична детерминированность исполнения и возможность пошаговой отладки, – например, в корпоративных информационных системах с требованиями к трассируемости решений. Децентрализованные паттерны уместны в задачах с высокой степенью параллелизма и слабосвязанными подзадачами, где глобальный координатор был бы не только излишен, но и контрпродуктивен.

Современная инженерная практика тяготеет к гибридным архитектурам, сочетающим сильные стороны обоих классов. Типичная схема предполагает оркестратор верхнего уровня, ответственный за планирование и контроль общего прогресса, в сочетании с децентрализованным взаимодействием агентов-исполнителей нижнего уровня через Blackboard или механизм peer-to-peer передачи управления. Практический выбор конкретного сочетания паттернов должен основываться на анализе трёх факторов: критичности

предсказуемости поведения, допустимой степени деградации при отказах и требуемой производительности при росте числа агентов.[9]

Заключение

Проведённый анализ подтверждает фундаментальный компромисс между управляемостью и масштабируемостью, присущий паттернам координации мультиагентных систем двух рассмотренных классов. Централизованные паттерны (Orchestrator/Supervisor, Pipeline, Planner-Executor) благодаря сосредоточению логики управления в единой точке обеспечивают высокую предсказуемость поведения, прозрачность исполнения и удобство отладки, оставаясь оптимальным решением для корпоративных информационных систем с жёсткими требованиями к трассируемости и детерминированности. Децентрализованные паттерны (Peer-to-peer, Blackboard, Stigmergy) обеспечивают горизонтальную масштабируемость и устойчивость к частичным отказам ценой усложнения верификации поведения и непредсказуемости в нештатных сценариях. Современная тенденция направлена на сближение характеристик этих классов за счёт гибридных архитектур, в которых централизованный оркестратор верхнего уровня сочетается с децентрализованным взаимодействием агентов-исполнителей через общую среду данных. Практический выбор конкретного паттерна или их комбинации должен основываться на балансе между требуемой предсказуемостью поведения, допустимой степенью деградации при отказах и ожидаемой нагрузкой при масштабировании системы.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Wooldridge M. Intelligent Agents: Theory and Practice / M. Wooldridge, N.R. Jennings // The Knowledge Engineering Review. – Cambridge: Cambridge University Press, 1995. – Vol. 10, No. 2. – Pp. 115–152.

2. Городецкий В.И. Многоагентные системы: современное состояние исследований и перспективы применения / В.И. Городецкий // Новости искусственного интеллекта. – 1996. – № 1. – С. 3–34.

3. LangChain. Multi-agent / LangChain. – Текст: электронный // LangChain Documentation: интернет-портал. – URL: https://docs.langchain.com/oss/python/langchain/multi-agent

4. Runkle S. Choosing the Right Multi-Agent Architecture / S. Runkle. – Текст: электронный // LANGCHAIN BLOG: интернет-портал. – URL: https://www.langchain.com/blog/choosing-the-right-multi-agent-architecture

5. Yang Y., Chai H., Shao S., Song Y., Qi S., Rui R., Zhang W. AgentNet: Decentralized Evolutionary Coordination for LLM-based Multi-Agent Systems / Y. Yang, H. Chai, S. Shao, Y. Song, S. Qi, R. Rui, W. Zhang // arXiv preprint arXiv:2504.00587. – 2025. – 20 p.

6. Corkill D.D. Blackboard Systems / D.D. Corkill // AI Expert. – 1991. – Vol. 6, No. 9. – Pp. 40–47.

7. Theraulaz G. A Brief History of Stigmergy / G. Theraulaz, E. Bonabeau // Artificial Life. – 1999. – Vol. 5, No. 2. – Pp. 97–116.

8. Dorri A. Multi-Agent Systems: A Survey / A. Dorri, S.S. Kanhere, R. Jurdak // IEEE Access. – 2018. – Vol. 6. – Pp. 28573–28593.

9. Microsoft. Agentic design patterns / Microsoft. – Текст: электронный // Microsoft AutoGen Documentation: интернет-портал. – URL: https://microsoft.github.io/autogen/stable/user-guide/core-user-guide/design-patterns/intro.html