

[Open in app](#)

Denis Korolev

106 Followers · About [Follow](#)

Monolith vs SOA vs MSA



Denis Korolev Feb 17, 2018 · 4 min read

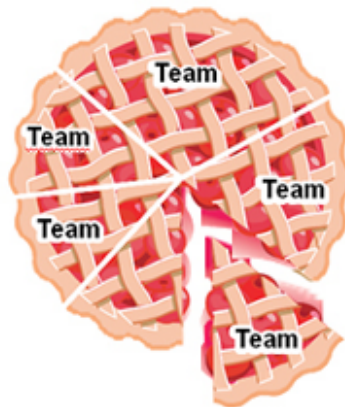
1990s and earlier

Pre-SOA (monolithic)



2000s

Traditional SOA



2010s

Microservices



Abstract

- There is no universal, modern or better architecture;
- Each architecture has its own pros and cons;
- The choice of architecture is not a matter of taste;

You need to consider:

- the speed of the project development;
- load speed;
- requirements for scaling and its appearance;
- requirements for a technological stack;
- financial and economic justification.

The proverbial CAP

- Consistency (English consistency of data) - in all nodes of a distributed system at one point in time, all data are the same and do not contradict each other;
- Availability (англ. доступность) — на любой запрос к любому узлу распределённой системы приходит ожидаемый ответ, но нет гарантии, что все ответы в один момент времени будут совпадать;
- Partition tolerance (англ. устойчивость к разделению) — разделение узлов распределённой системы на несколько изолированных сегментов не приводит к неожиданной работе системы.

Architecture/Feature	Consistency	Availability	Partition tolerance
Monolith	✓	-	-
SOA	-	✓	✓
MSA	-	✓	✓

Яркие свойства архитектуры

Важно: минусы могут варьироваться от конечного архитектурного решения.

Масштабирование

Вертикальное

Самое понятное масштабирование, когда в жертву демона Server Application приносят молодое девственное железо.

Самый простой пример:

- Сервер, на котором работает серверное приложение — один;
- Нагрузка на сервер возросла выше 75–80% (среднее, максимальное);
- Пиковая нагрузка близка к 100%;
- Железо уже не обеспечивает технические требования по скорости отклика.

В этом случае покупают более мощное железо, расширяют канал связи.

Суть: увеличивается качество железа, но не количество.

У этого подхода есть явное ограничение сверху: когда самого мощного железа будет не хватать — проекту наступит медленная смерть.

Горизонтальное

Используется уже на состоявшихся проектах со взрослой командой разработчиков и эксплуатации.

Пример:


- Сервис, который занимается обработкой видео (стриминг);
- При увеличении числа форматов и разрешения видео необходимо увеличить число параллельно обрабатываемых файлов (поток);

В этом случае можно увеличить число параллельно обрабатываемых потоков (несколько процессов за счёт ОС, несколько серверов, которые выполняют одновременную конвертацию).

Суть: увеличивается количество, но не качество.

У этого подхода есть много неявных ограничений, популярные из них:

- Ограничение по каналам связи;
- Лавинная нагрузка.



Допустимость вида масштабирования от архитектуры

Ресурсоёмкость

Виды ресурсов

- Память (RAM);
- Ресурсы процессора (CPU);
- Место на диске (HDD/SSD);
- Коммуникация (LAN/CPU/RAM).

Чувствительность к ресурсам



Зависимость архитектуры от архитектуры (примеры из практики, могут быть исключения); S — small, M — Middle, L — Large.

Процесс разработки

Monolith

Плюсы:

- Реализация в коде простая и понятная;
- Малое количество ролей при разработке и эксплуатации;
- Можно брать разработчиков со средней квалификацией;
- Легко обеспечивать транзакционную целостность данных;
- Не требует высоких знаний у группы эксплуатации.

Минусы:

- Единый технологический стек;
- Большие временные и человеческие затраты при модифицировании и рефакторинге: изменения одну части кода потребуются изменить везде в связанных частях;
- Перезапуск всего приложения при появлении новой функциональности (есть исключение на разных технологических стеках).

SOA

Плюсы:

- Позволяет осуществлять горизонтальное масштабирование;
- Требования к технологическому стеку ниже, чем у монолита: каждый сервис может работать на своём стеке;
- Есть готовые решения для организации SOA от Microsoft, Oracle, Google, IBM, часто всего это коммерческие продукты, как следствие, доступность для изучения низкая, практиковаться никто не даст, нужна лицензиями подтверждённая квалификация.

Минусы:

- Высокий порог входа: нужны знания в SOAP, CORBA, REST, RPC, AMQ или подобном;
- Требуется умение создавать, организовывать и поддерживать шины данных на разных технологических стеках;
- Согласованность данных снижается;
- Сложнее обеспечивать транзакционную целостность;
- Требуются специалисты по тестированию с навыками куда выше, чем на монолитных решениях.

MSA

Плюсы:

- Обеспечивает ещё большую гибкость в масштабировании;
- Высокая стеко-независимость;
- Позволяет почти безшовно менять монолитную архитектуру, параллельно меняя технологический стек;
- Декомпозиция задач похожа на декомпозицию микросервисов;
- Легко тестировать работу микросервиса (одного, а не комплекса);
- Допустимо перезапускать только того, что изменилось;
- Допустима поддержка нескольких версий одновременно.

Минусы:

- Стеко-независимость приводит к хаосу внутри проекта;
- Нужно обеспечивать обратную совместимость для всех сервисов;
- Нужен навык в оркестрировании сервисов;
- Нужен навык в развёртывании микросервисов;
- Нужен навык в управлении распределёнными;
- Высокая сложность организации транзакций;
- Высокая сложность в разработке устойчивой к сетевым ошибкам приложений;
- Требуются специалисты по тестирования с очень высокими навыками.

Связанность

Monolith

- Любые изменения в коде зависимости требуют изменения потребителей кода;

- Для ведения проекта, как правило, требуется знания во всех сопутствующих областях кода;
- Невозможно сменить технологический стек;
- Даже при модульной разработке невозможно разным модулям использовать стороннюю зависимость разных версий.

SOA

- Картина аналогична монолиту;
- Для ведения проекта дополнительно требуются знания в стандартах/форматах для обмена данными.

MSA

- Низкая кодовая связанность между микросервисов (при правильной организации кода).

Дирижирование

Этой проблемы не существует на Monolith и на SOA в силу специфики организации приложения.

У MSA появились новые сложности:

- Управление распределёнными конфигурациями (их стало много) — Consul;
- Deployment and load management (so as not to run by hand) - Kubernetes.

Have questions?

You can discuss here - <https://www.facebook.com/groups/deb.beer/>

[About](#) [Help](#) [Legal](#)

Get the Medium app

