What does it take to be an architect ...

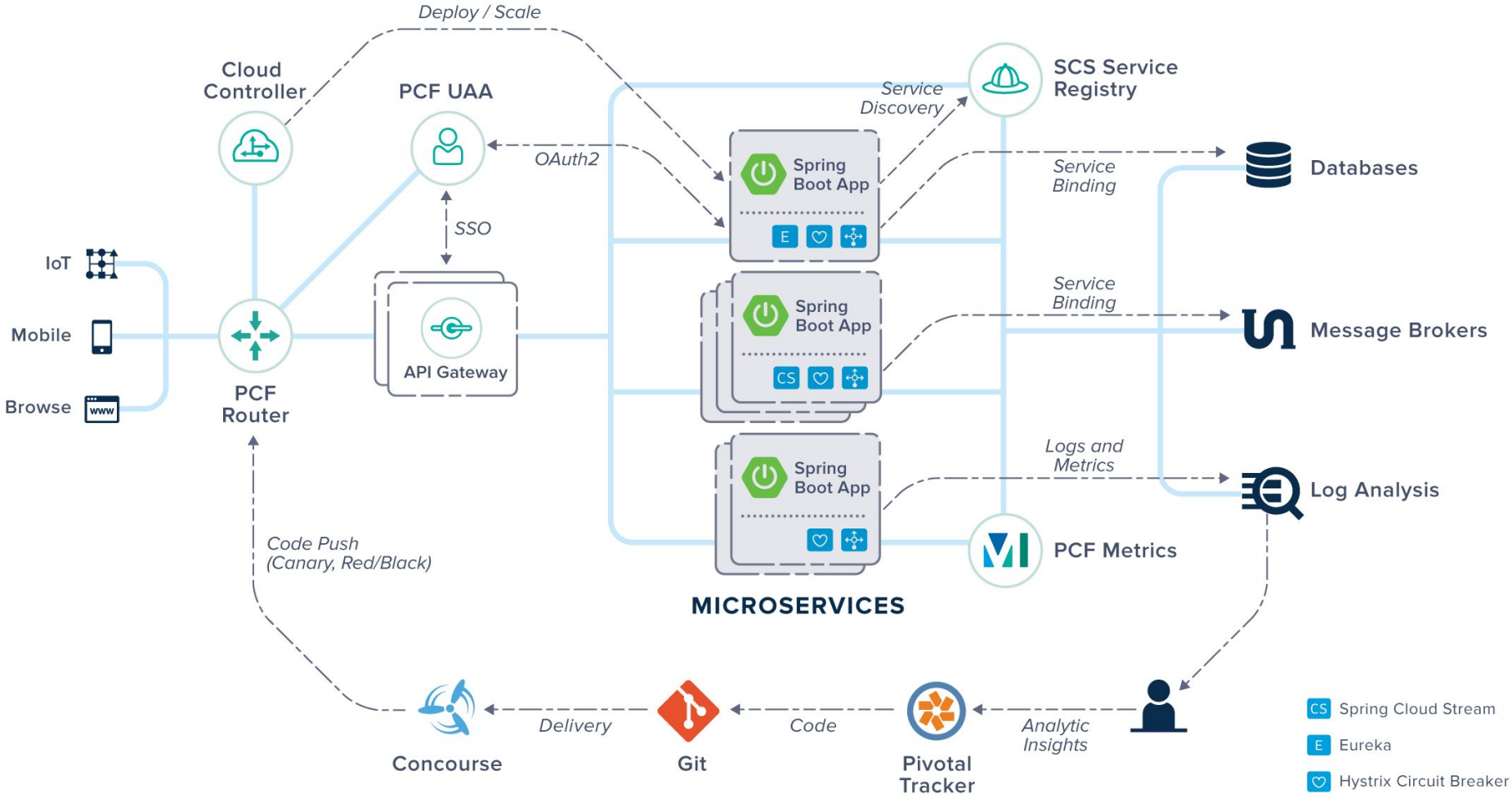
Konstantin Slisenko
Solution Architect at EPAM

twitter: @kslisenko
facebook.com/konstantin.slisenko.1

My own experience

- Big company (not a startup)
- Teams in different cities and time zones
- Enough time and money

What is architecture?



https://dzone.com/articles/spring-cloud-microservices-at-pivotal-platform

Architecture work is

Architecture work is

1. Understanding business goals

Architecture work is

1. Understanding business goals
2. Working with stakeholders

Architecture work is

1. Understanding business goals
2. Working with stakeholders
3. Identifying requirements and constraints

Architecture work is

1. Understanding business goals
2. Working with stakeholders
3. Identifying requirements and constraints
4. Making design decisions

Architecture work is

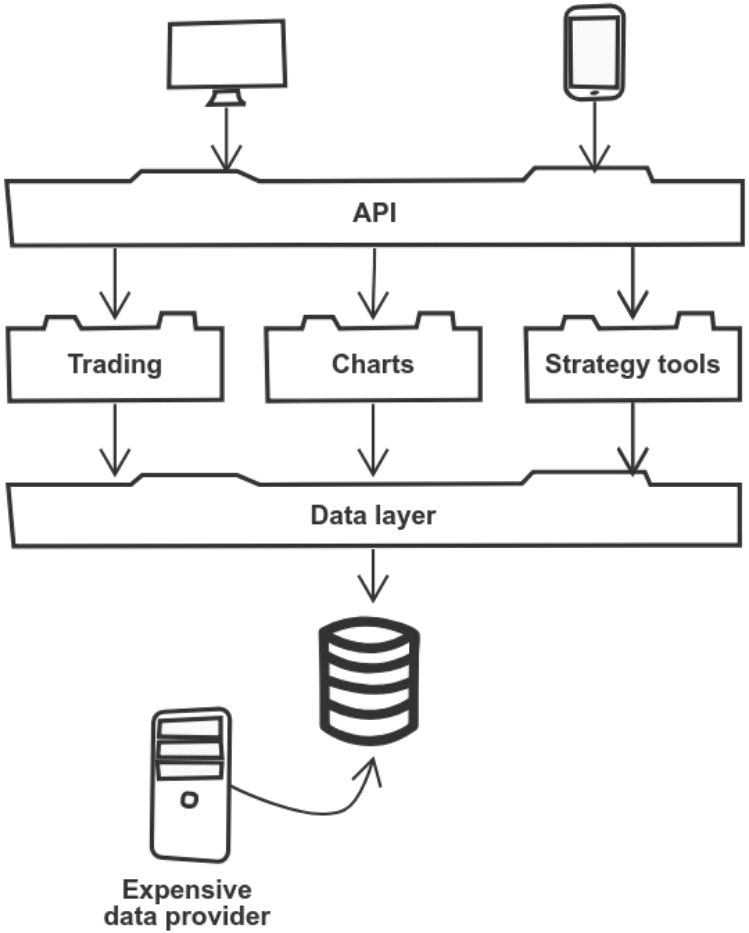
1. Understanding business goals
2. Working with stakeholders
3. Identifying requirements and constraints
4. Making design decisions
5. Documenting and communicating

Traders, Inc

professional trading services

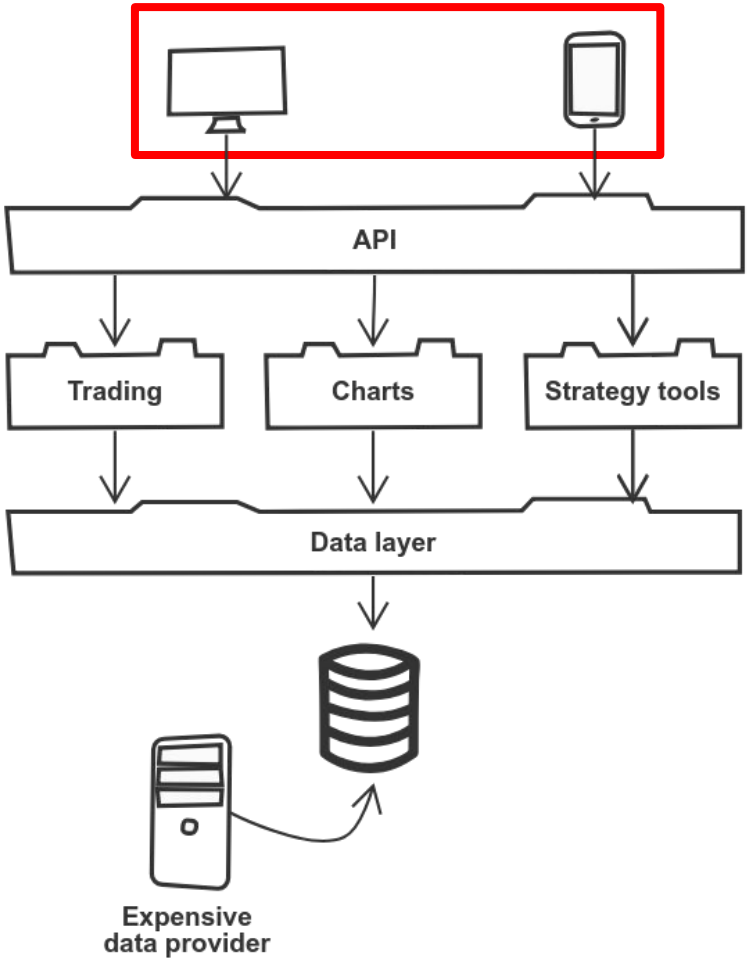
Traders, Inc

Prodessional trading services



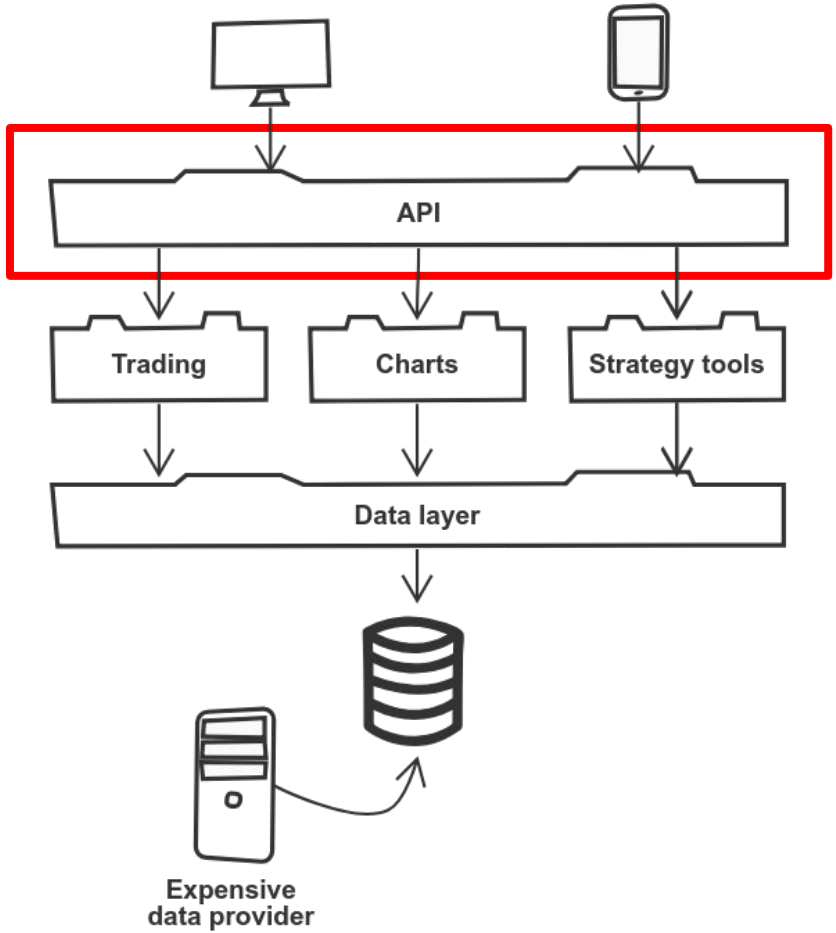
Traders, Inc

Prodessional trading services



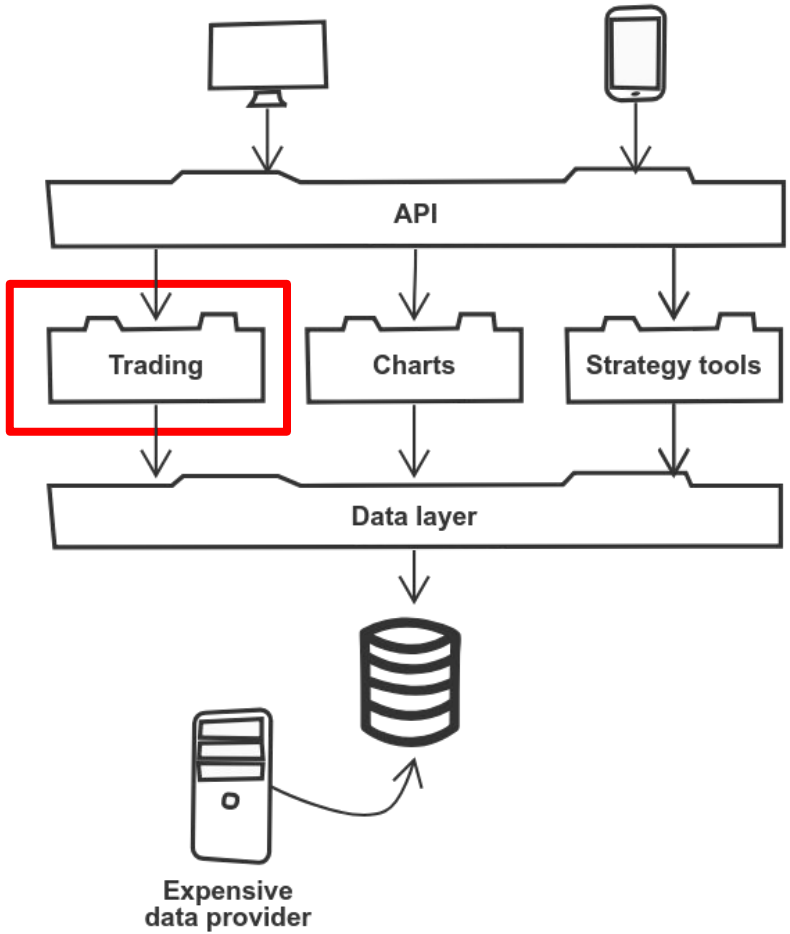
Traders, Inc

Prodessional trading services



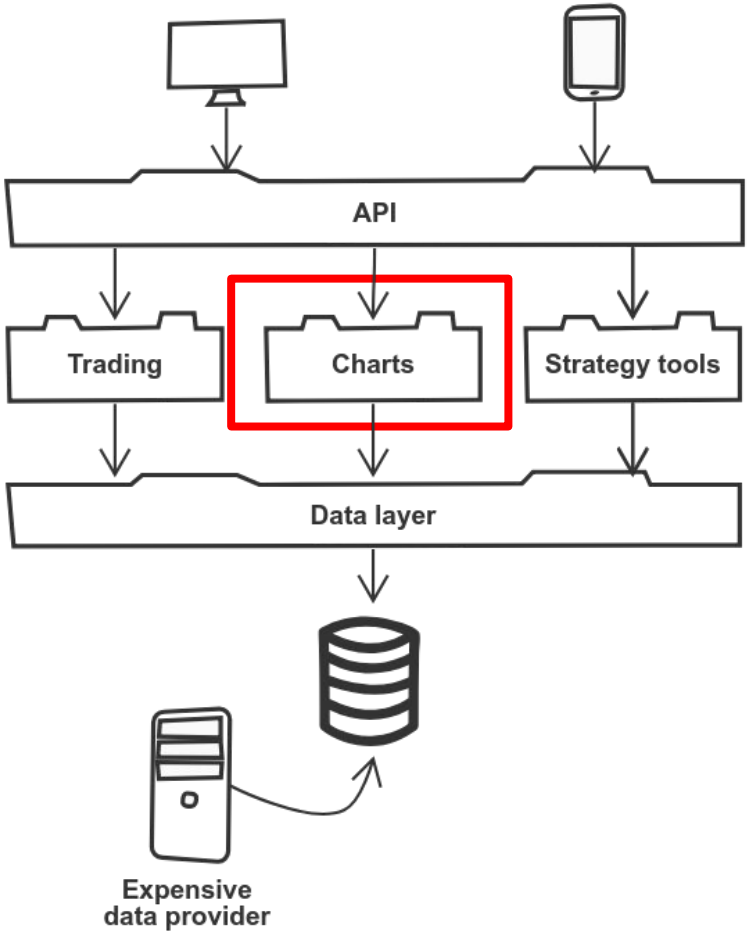
Traders, Inc

Prodessional trading services



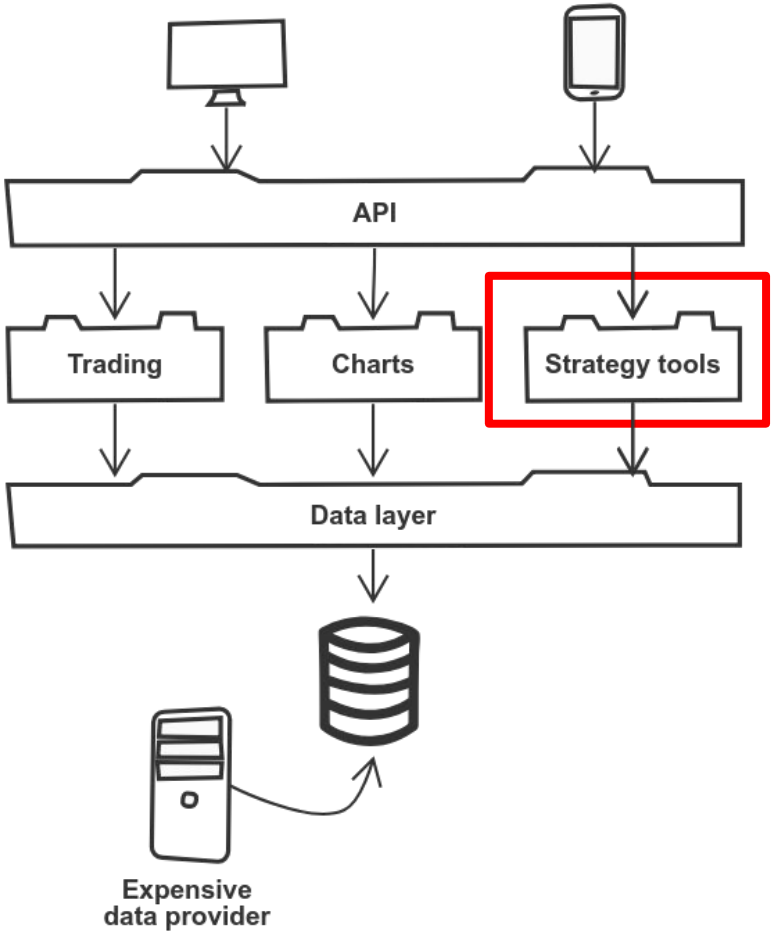
Traders, Inc

Prodessional trading services



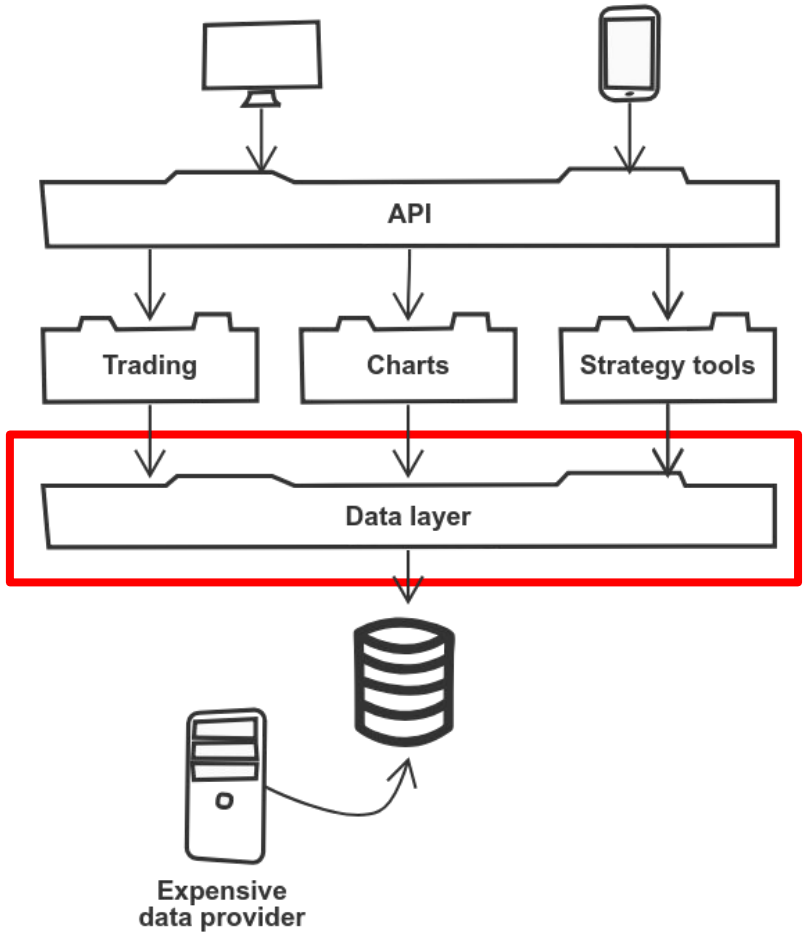
Traders, Inc

Prodessional trading services



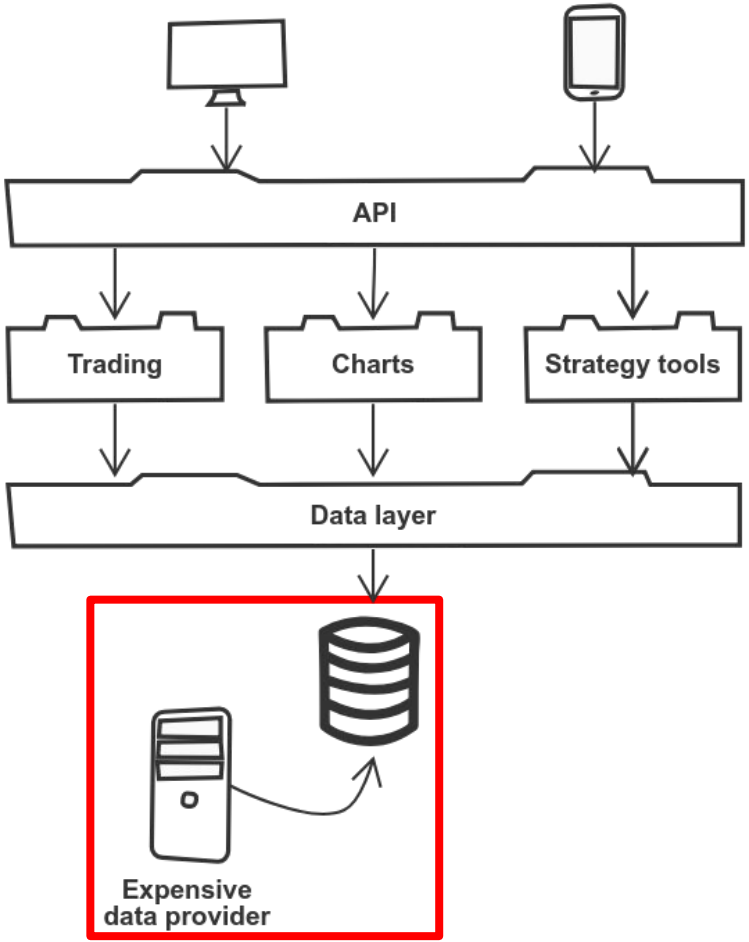
Traders, Inc

Prodessional trading services



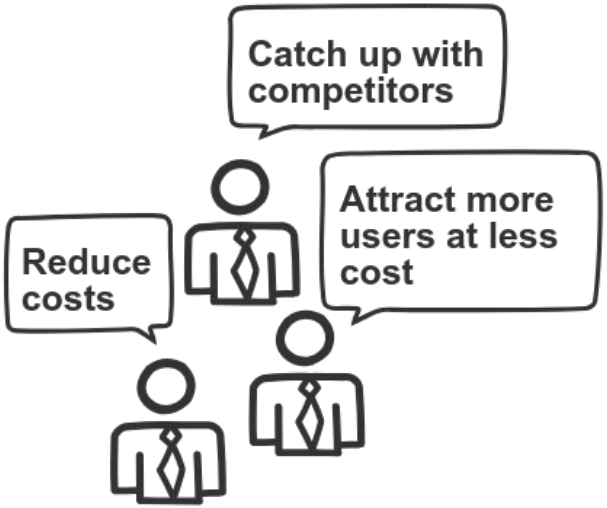
Traders, Inc

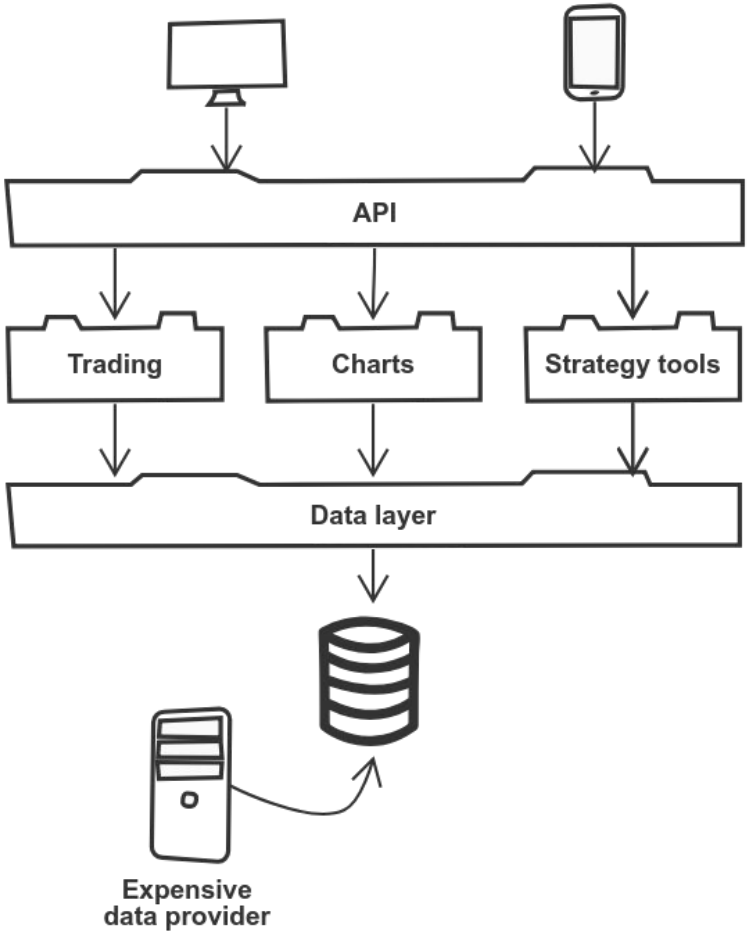
Prodessional trading services



Traders, Inc

Prodessional trading services





Sure!

- I can rewrite it
- Using cool latest technology
- Enhance the API
- Improve performance
- Refactor
- And do much more!

“Poor understanding of business goals is the path to the dark side.

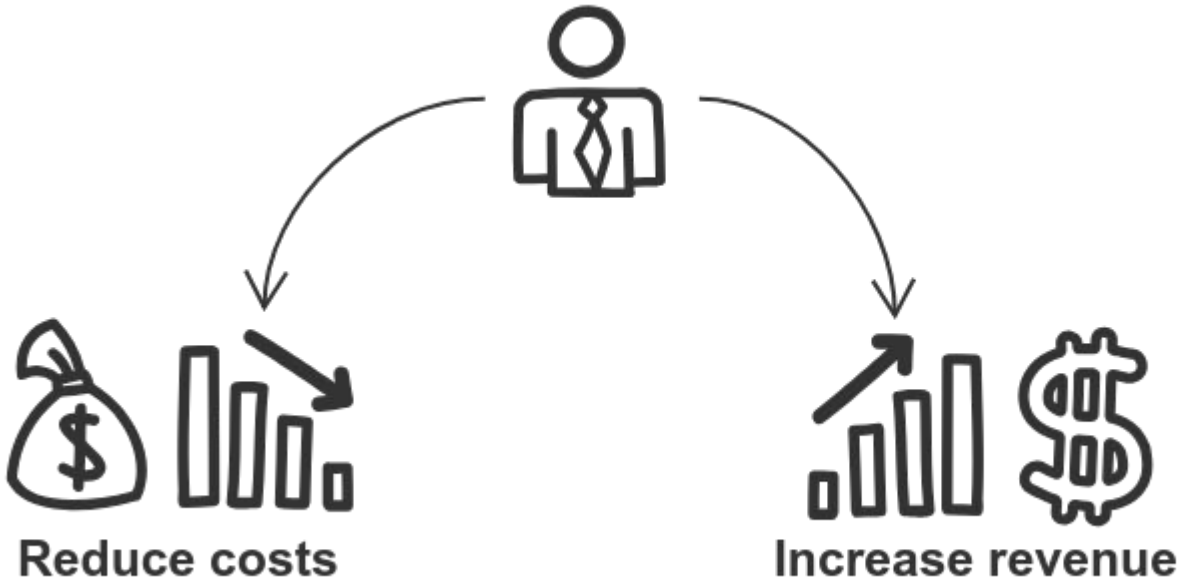
It leads to wrong design decisions.

Wrong design decisions lead to wrong software.

Wrong software leads to business failure.”

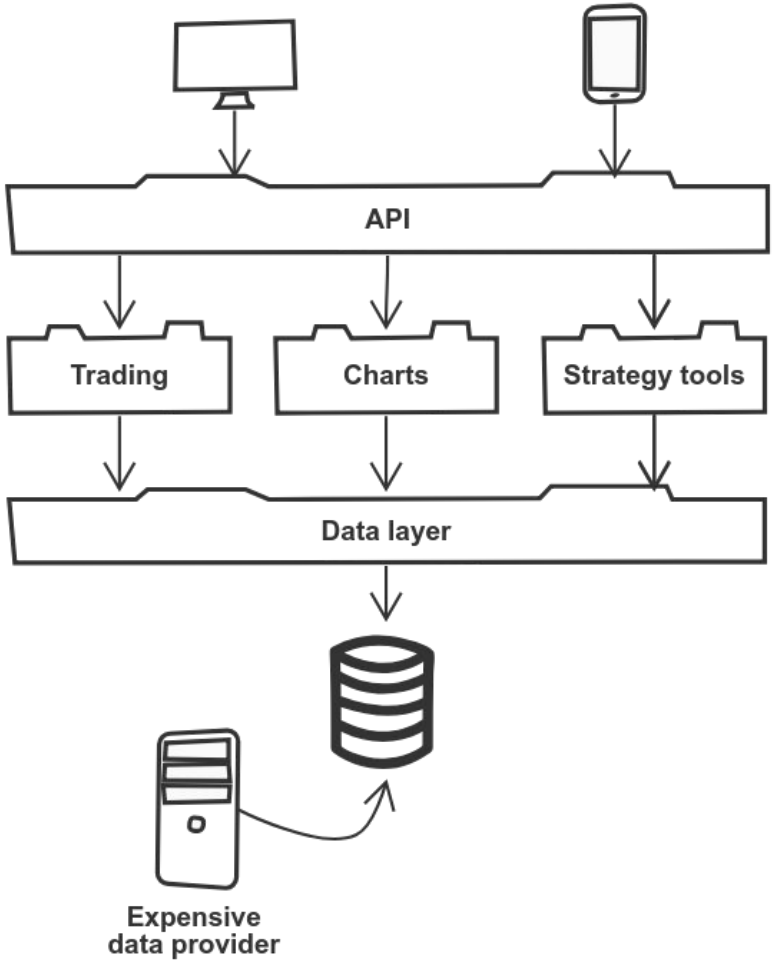
Business goals

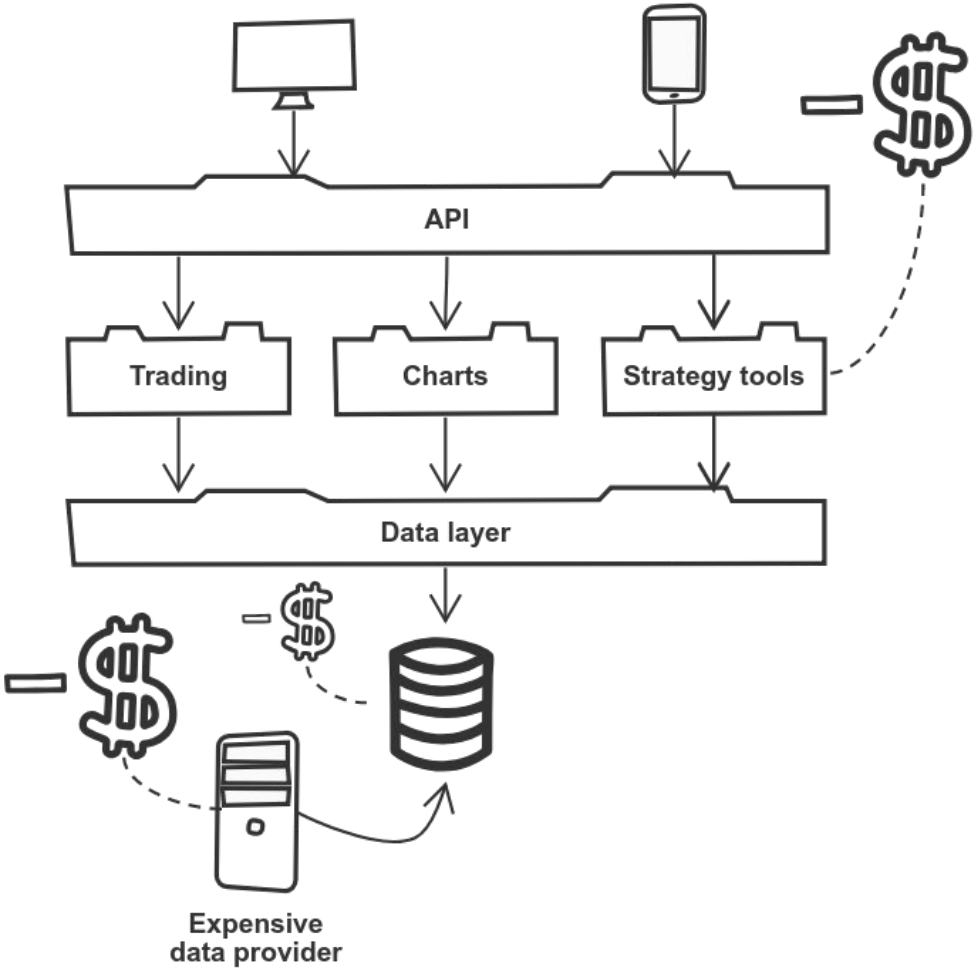
Possible business goals of any company

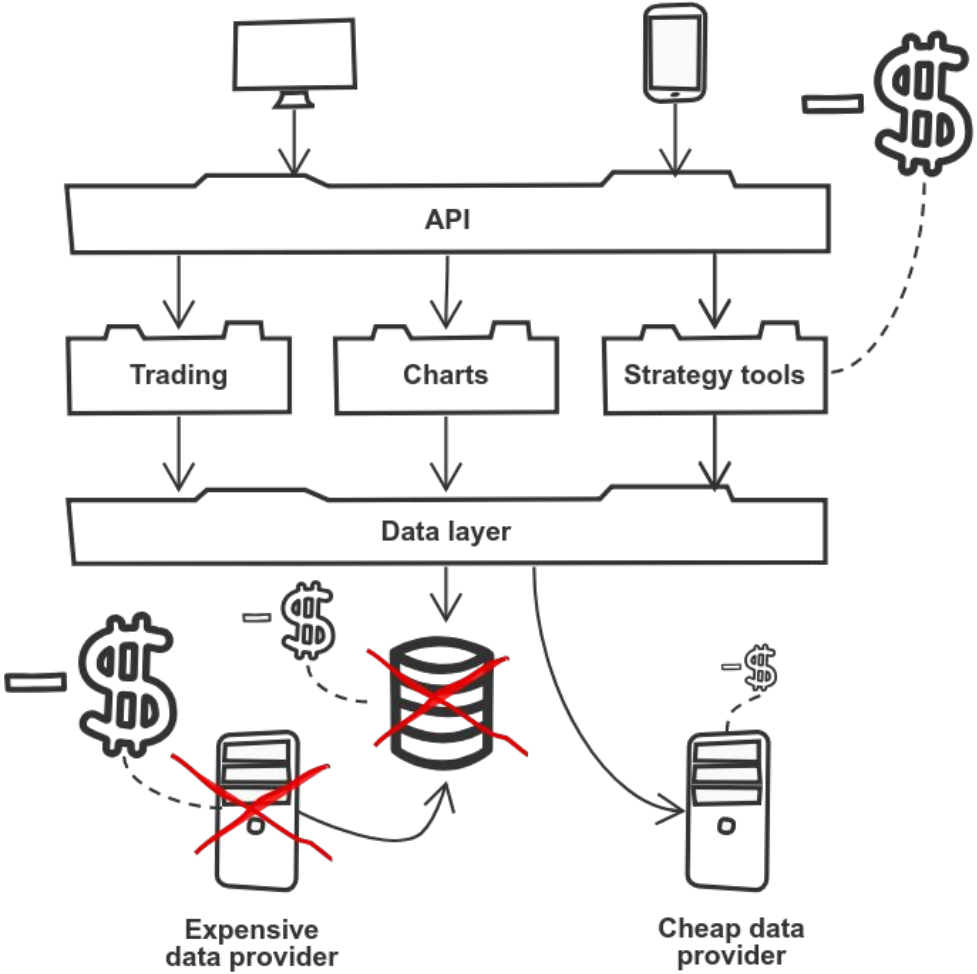


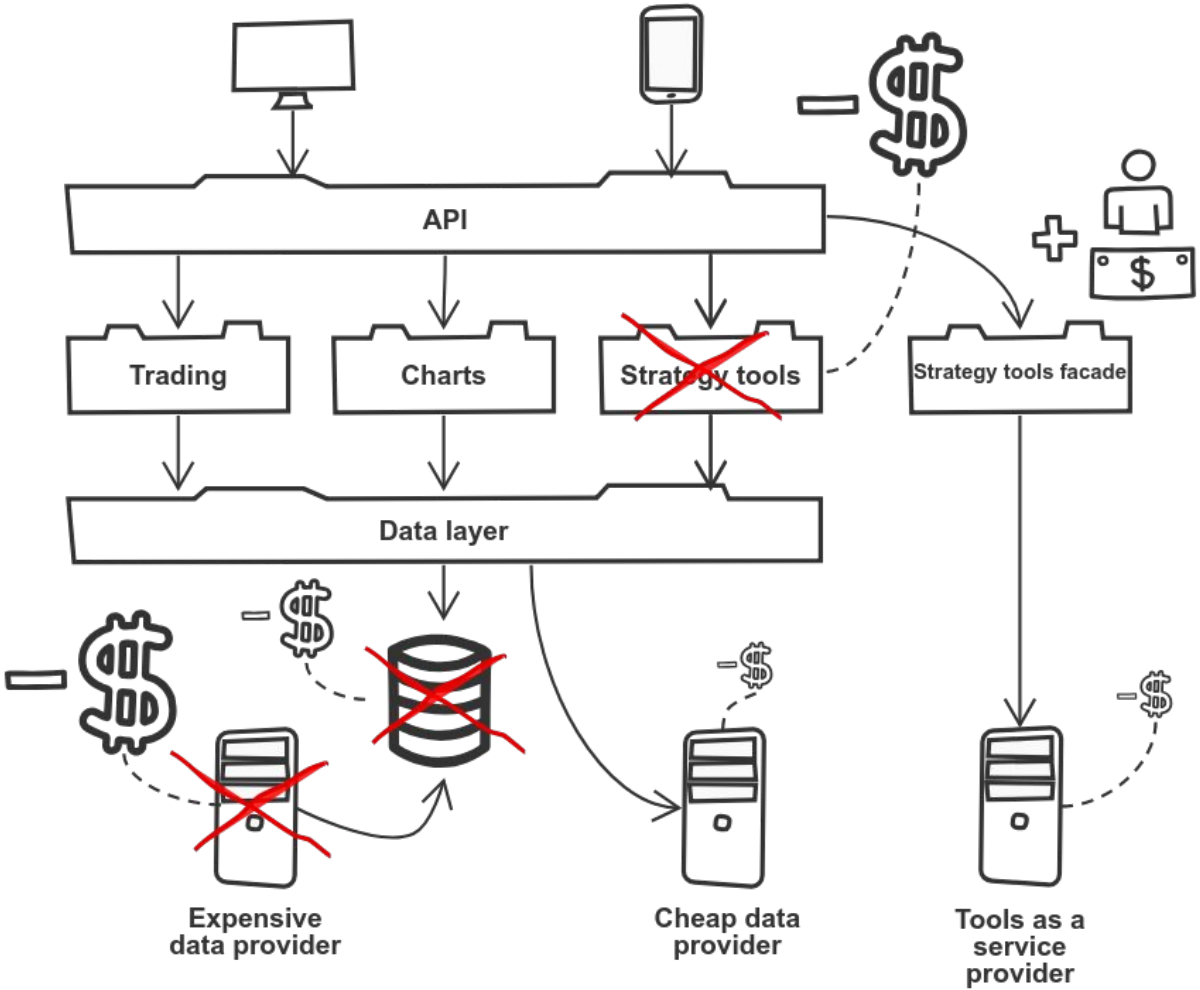
Business goals

1. **Reduce costs** by moving to a cheap data provider
2. **Avoid expenses** for database maintenance
3. **Attract more users** at **less cost** (outsource strategy tools)


API
Trading
Charts
Strategy tools
Data layer
Expensive
data provider


API
Trading
Charts
Strategy tools
Data layer
Expensive
data provider


API
Trading
Charts
Strategy tools
Data layer
Expensive data provider
Cheap data provider

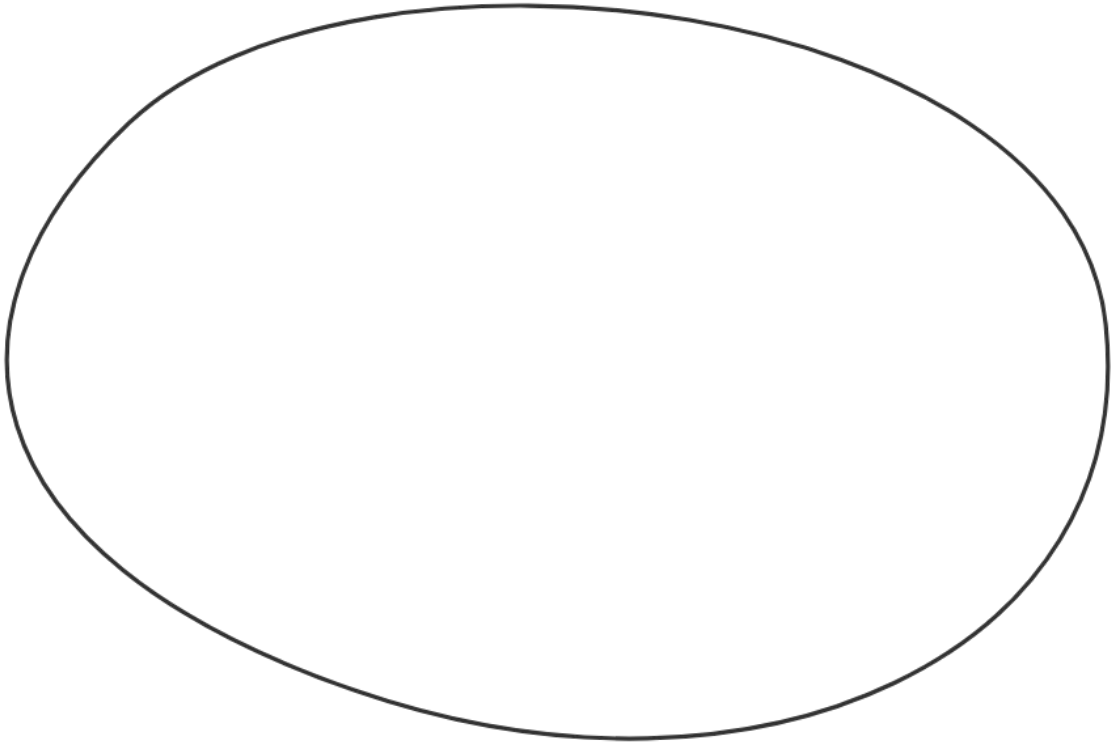

API
Trading
Charts
Strategy tools
Strategy tools facade
Data layer
Expensive data provider
Cheap data provider
Tools as a service provider

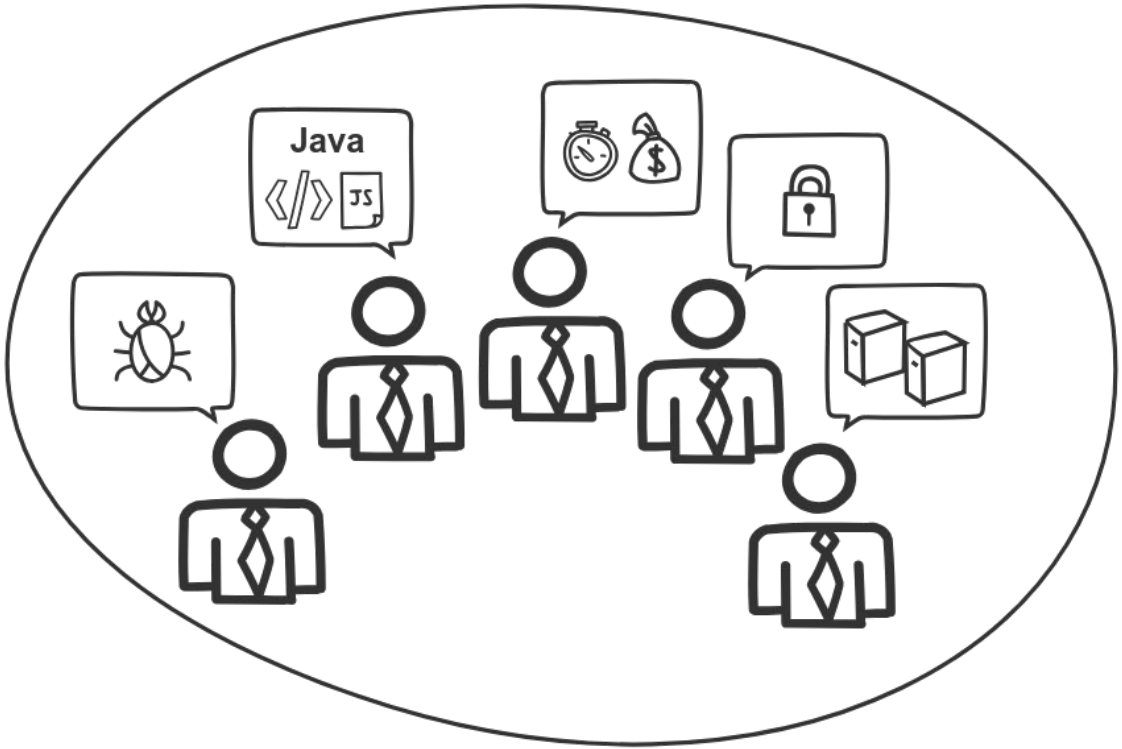
What else do we need before we start coding?

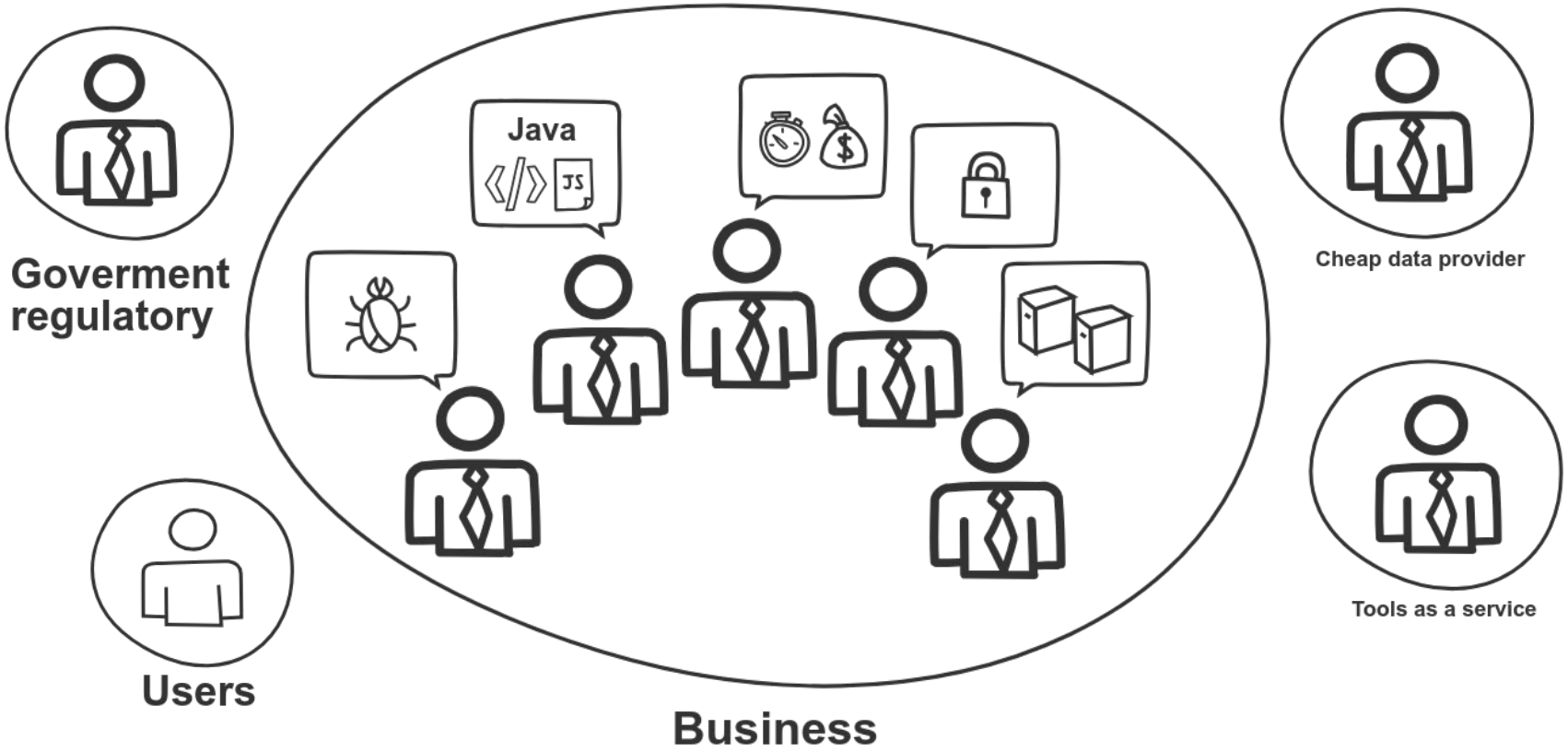
“Patience you must have my young padawan.

You haven’t met the stakeholders.”

Stakeholders


Business


Java
JS
Business

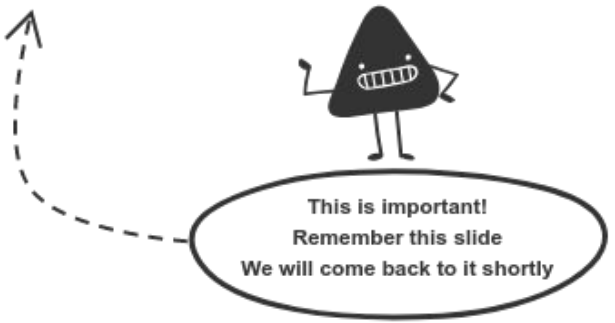

Goverment
regulatory
Java
JS
Cheap data provider
Tools as a service
Users
Business

Stakeholders

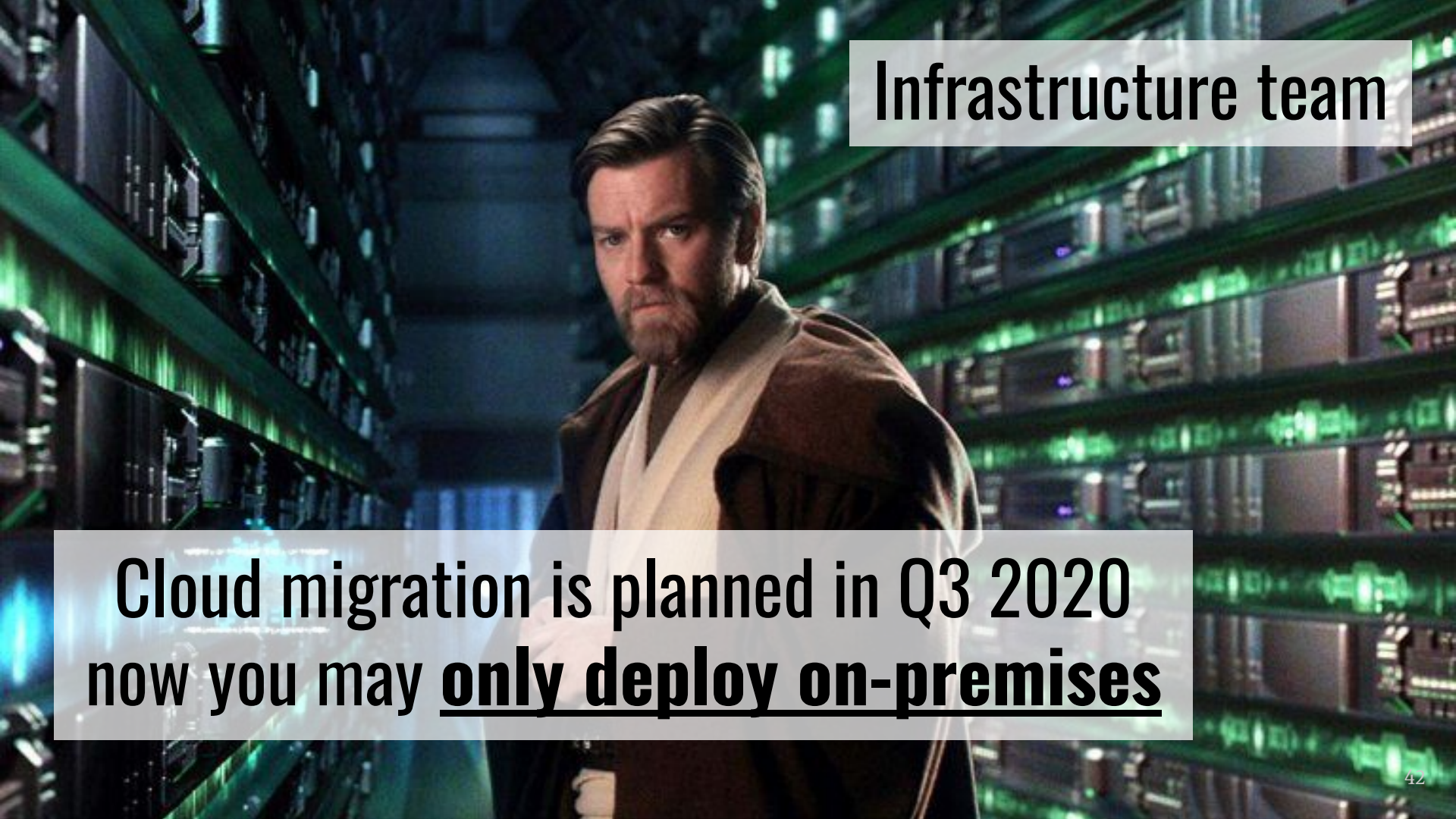
1. Responsible for different requirement areas
2. Sometimes non-technical people
3. Impact on design decisions

Requirements
Constraints

Smart architects communicate with different stakeholders differently!




Product manager
After switching to a new data provider
performance and uptime
should not decrease


Infrastructure team
Cloud migration is planned in Q3 2020
now you may only deploy on-premises


Web and mobile developers
Sorry, we are busy with other projects now

Requirements

1. Response time < 3 seconds
   (95% calls for 1000 concurrent users)
2. No single point of failure

Requirements

1. Response time < 3 seconds
   (95% calls for 1000 concurrent users)
2. No single point of failure

Constraints

3. API must stay backward-compatible
4. Deployment only on-premises

Constraints

1. Time
2. Budget
3. Team skills
4. Used technology stack
5. Approved technology stack
6. ...

Requirements

1. Response time < 3 seconds
   (95% calls for 1000 concurrent users)*
2. No single point of failure**

we assume that:

Assumptions

**SLA of data provider will match performance and uptime requirements*

***Tools as a service provides failover connection*

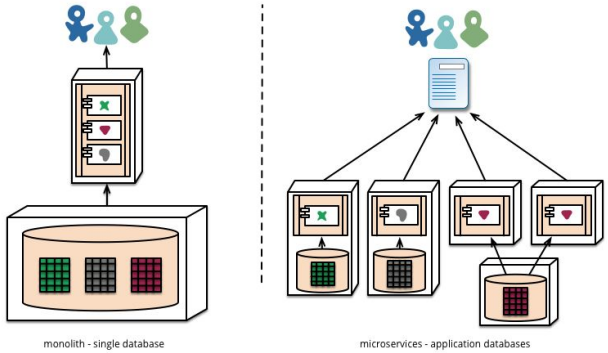
Grouping requirements

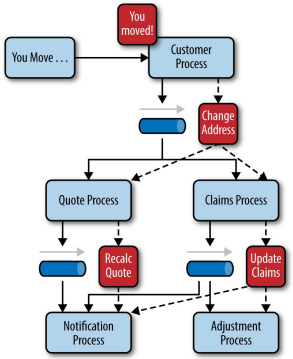
Performance	Response time, throughput
Availability	Uptime, downtime, fault-tolerance
Scalability	What can grow, how we handle
Security	Authentication, authorization, encryption, ...
Manageability	Metrics, health checks, ...
Interoperability	Message formats and protocols
...	

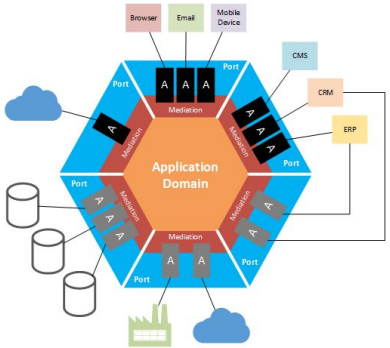
Architecture design

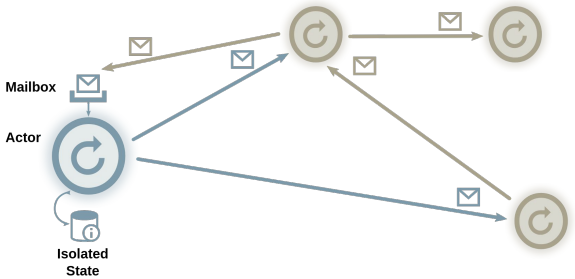
Architecture styles

- Monolith
- Microservices
- Event-driven
- Actor based
- SOA
- Hexagonical
- Serverless
- …





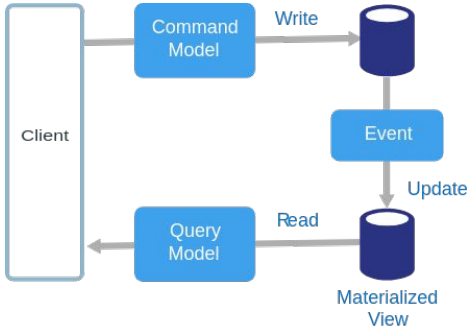


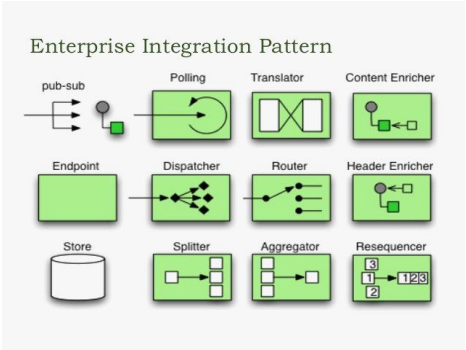


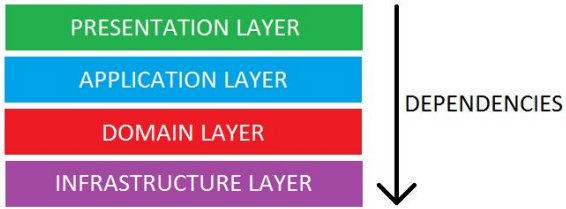
https://martinfowler.com/articles/microservices.html
https://pradeeploganathan.com/architecture/hexagonal-architecture/
https://www.oreilly.com/library/view/software-architecture-patterns/9781491971437/
https://berb.github.io/diploma-thesis/original/054_actors.html

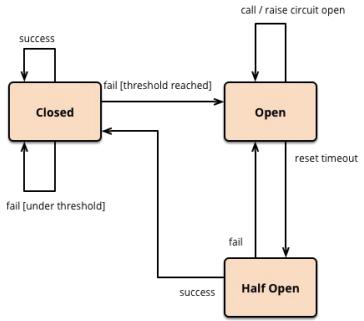
Architecture patterns

- Layered
- Module-based
- CQRS
- Integration patterns
- Fault-tolerance patterns
- …









https://docs.microsoft.com/bs-latn-ba/azure/architecture/guide/architecture-styles/cqrs?view=azuremgmtcompute-14.1.0-prerelease
https://dzone.com/articles/layered-architecture-is-good
https://oldworldbaking.co/amp/enterprise-integration-patterns/enterprise-integration-patterns-awesome-integration-patterns-and-practices-for-cloud-and-mobile/
https://martinfowler.com/bliki/CircuitBreaker.html

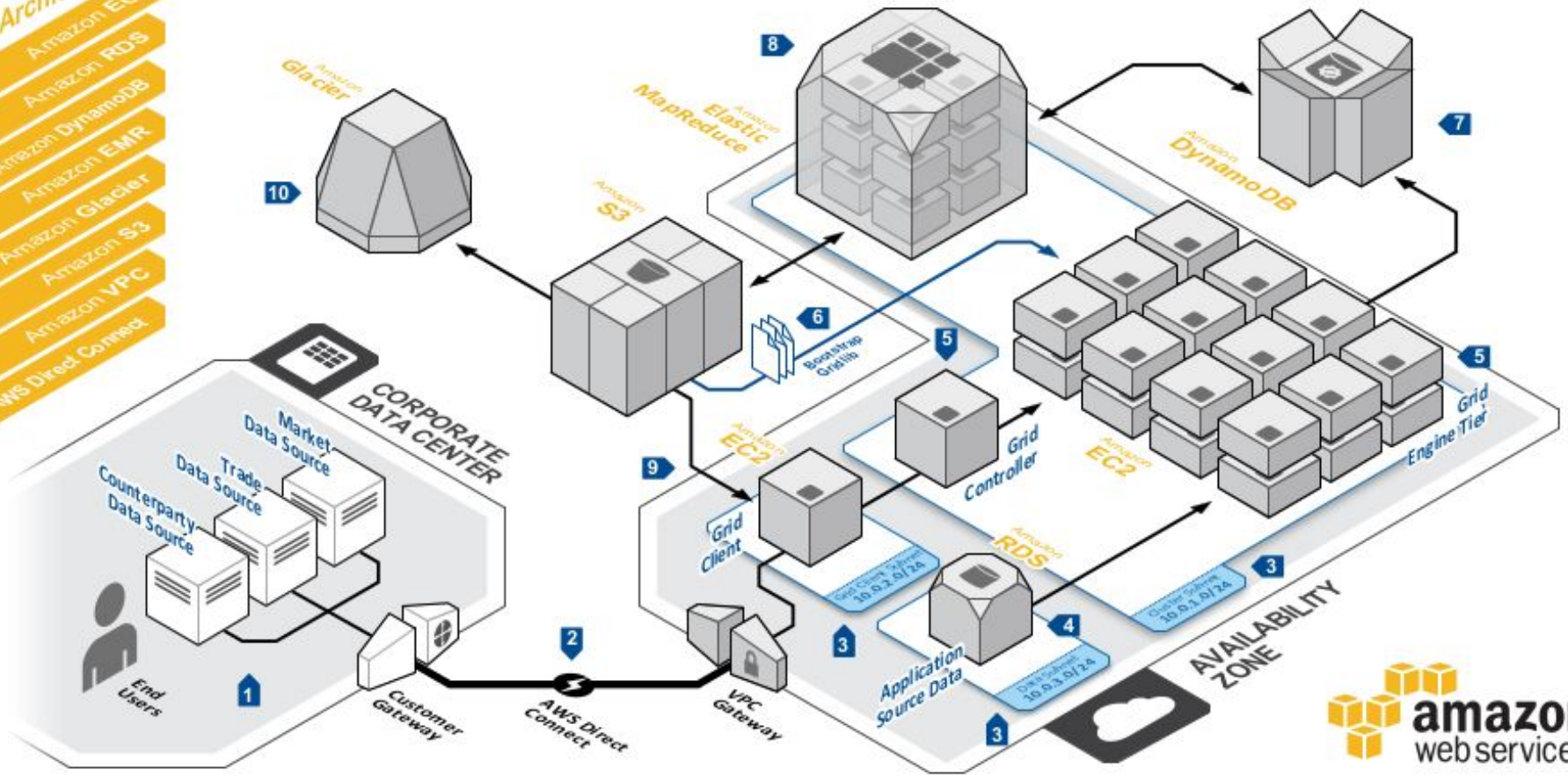
AWS Reference Architectures

- Amazon EC2
- Amazon RDS
- Amazon DynamoDB
- Amazon EMR
- Amazon Glacier
- Amazon S3
- Amazon VPC
- AWS Direct Connect

FINANCIAL SERVICES GRID COMPUTING

Financial services grid computing on the cloud provides dynamic scalability and elasticity for operation when compute jobs are required, and utilizing services for aggregation that simplify the development of grid software.

On demand provisioning of hardware, and template driven deployment, combined with low latency access to existing on-premise data sources make AWS a powerful platform for high performance grid computing systems.



amazon web services™

“A Jedi makes design decisions for achieving business goals and requirements”


Caution!
Debatable things

Design decision	Requirement/constraint
Deploy on-premises as standalone services	Usage of cloud services not possible till Q3 2020, we can’t wait so long

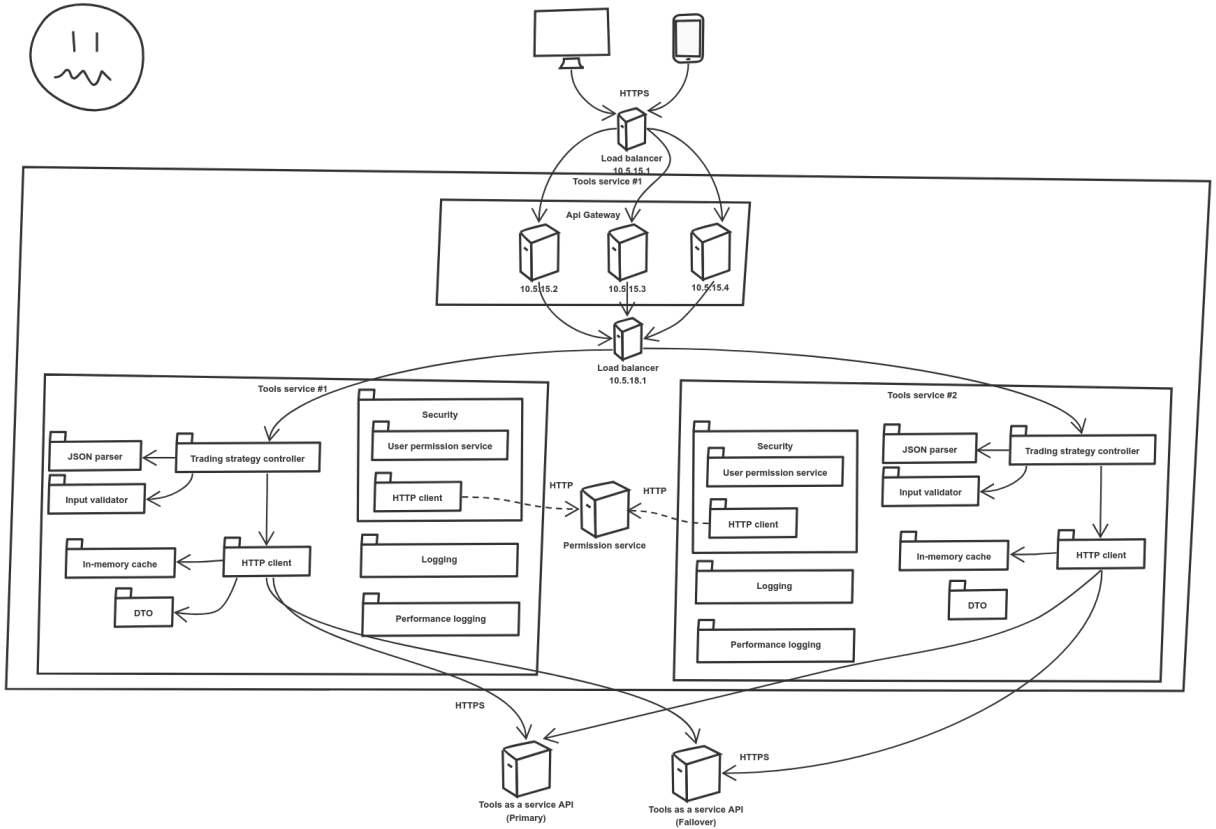
Design decision	Requirement/constraint
Deploy on-premises as standalone services	Usage of cloud services not possible till Q3 2020, we can’t wait so long
Use distributed cache for charts	**Performance requirements**

Design decision	Requirement/constraint
Deploy on-premises as standalone services	Usage of cloud services not possible till Q3 2020, we can't wait so long
Use distributed cache for charts	Performance requirements
Deploy at least 2 instances in 2 data-centers (active) + DR (cold)	**No single point of failure**
Circuit breaker for primary and failover connections to Tools as a service provider*	**No single point of failure** ***Tools as a service provides primary and failover endpoints***

Design decision	Requirement/constraint
Deploy on-premises as standalone services	Usage of cloud services not possible till Q3 2020, we can't wait so long
Use distributed cache for charts	Performance requirements
Deploy at least 2 instances in 2 data-centers (active) + DR (cold)	No single point of failure
Circuit breaker for primary and failover connections to Tools as a service provider*	No single point of failure **Tools as a service provides primary and failover endpoints*
Using JSON, same data format as existing API	**Must stay backward-compatible with existing API as client changes are not possible**

Communicating design decisions

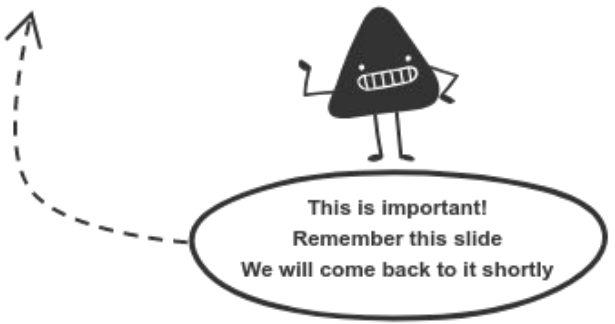
“Always pass on what you have designed”


HTTPS
Load balancer
10.5.15.1
Tools service #1
Api Gateway
10.5.15.2
10.5.15.3
10.5.15.4
Load balancer
10.5.18.1
Tools service #1
JSON parser
Trading strategy controller
Input validator
In-memory cache
HTTP client
DTO
Security
User permission service
HTTP client
Logging
Performance logging
HTTP
HTTP
Permission service
Tools service #2
Security
User permission service
HTTP client
Logging
Performance logging
JSON parser
Trading strategy controller
Input validator
In-memory cache
HTTP client
DTO
HTTPS
HTTPS
Tools as a service API
(Primary)
Tools as a service API
(Failover)

“Understanding your diagram is pain.

Terrible pain”

Smart architects communicate with different stakeholders differently!



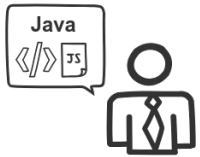
Points of view on architecture

Context view

Points of view on architecture

Context view

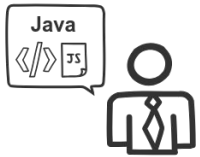
Component view



Points of view on architecture

Context view

Component view

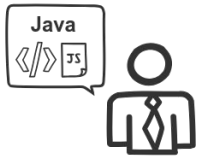


Deployment view

Points of view on architecture

Context view

Component view

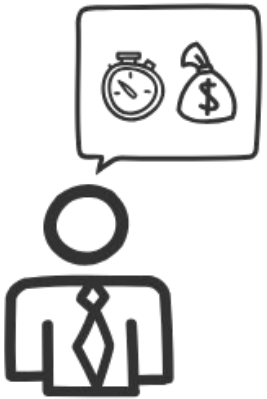


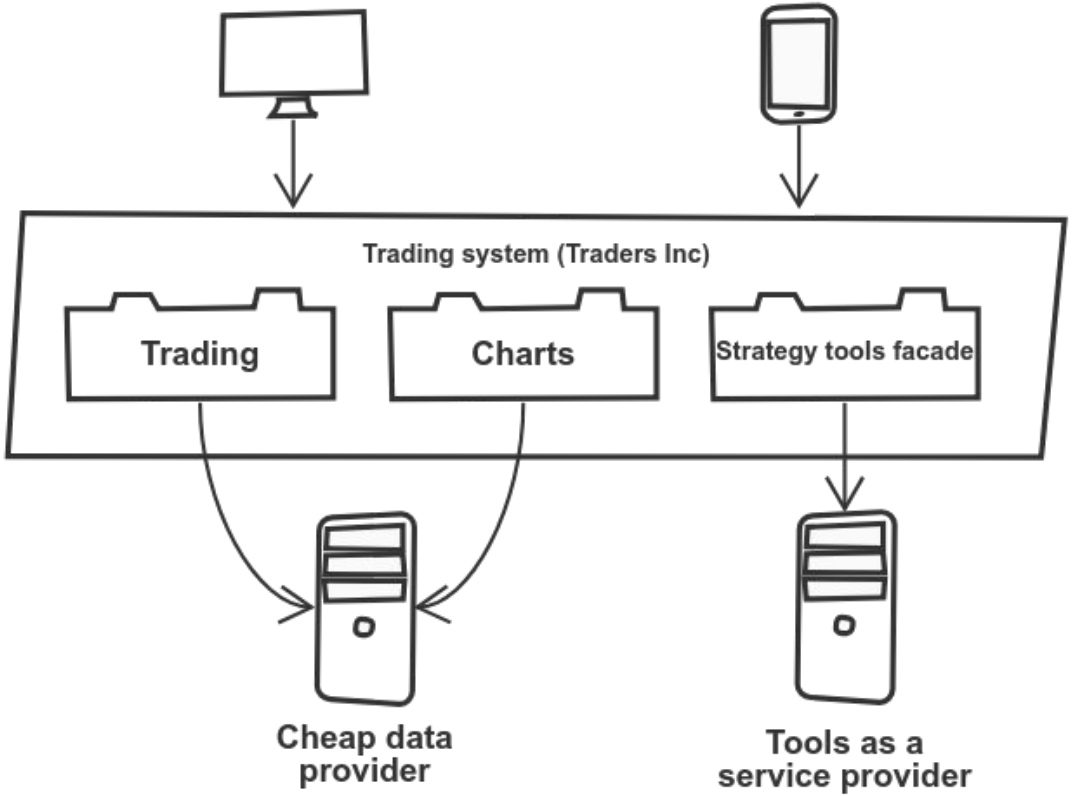
Deployment view

Other views

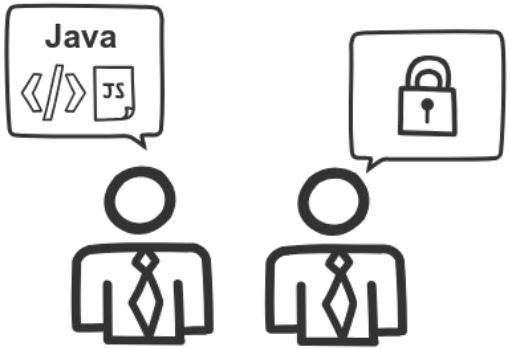
other stakeholders

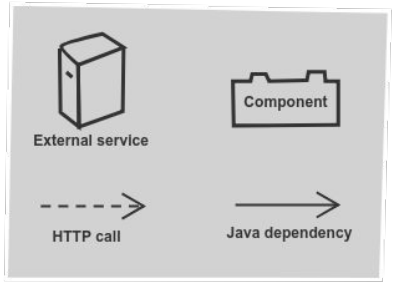
Context diagram

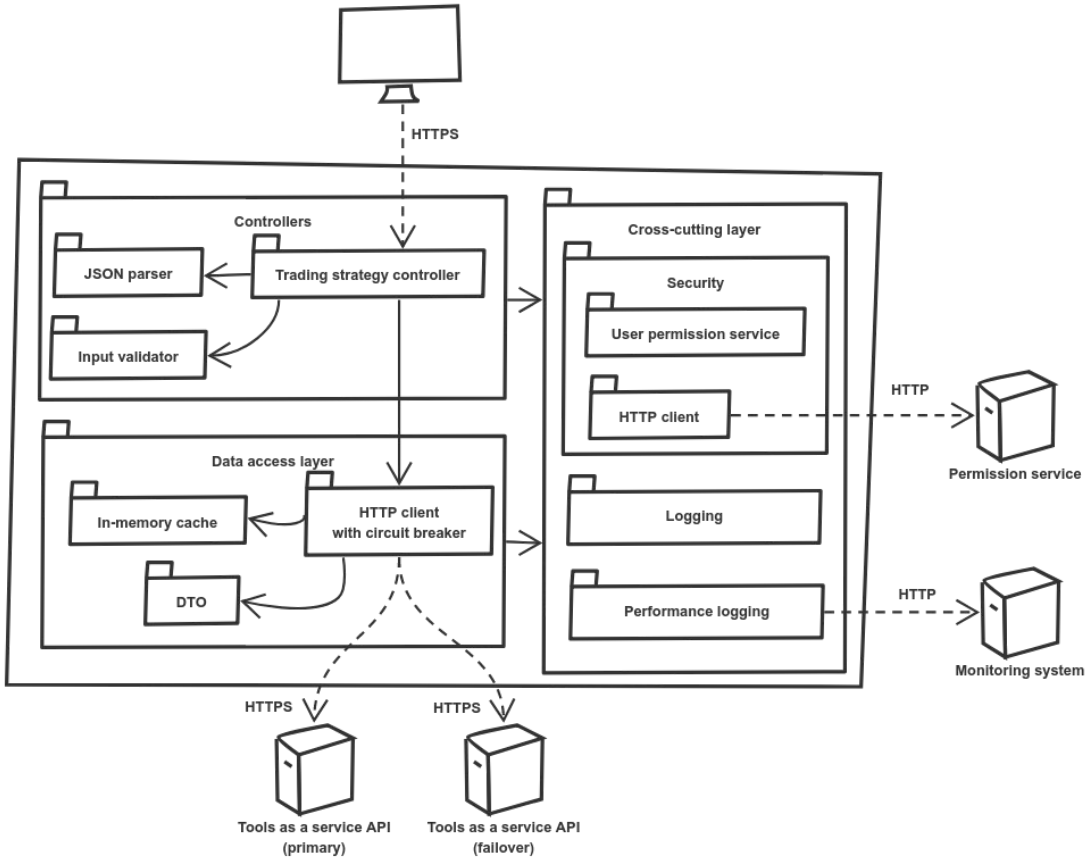




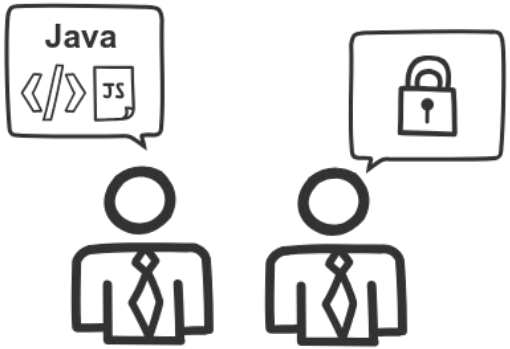
Component diagram

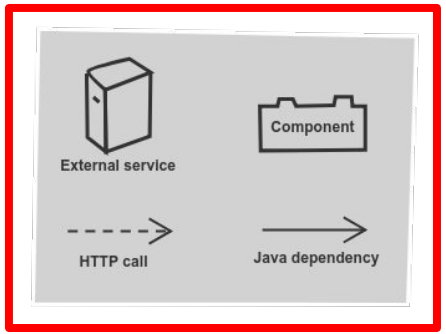


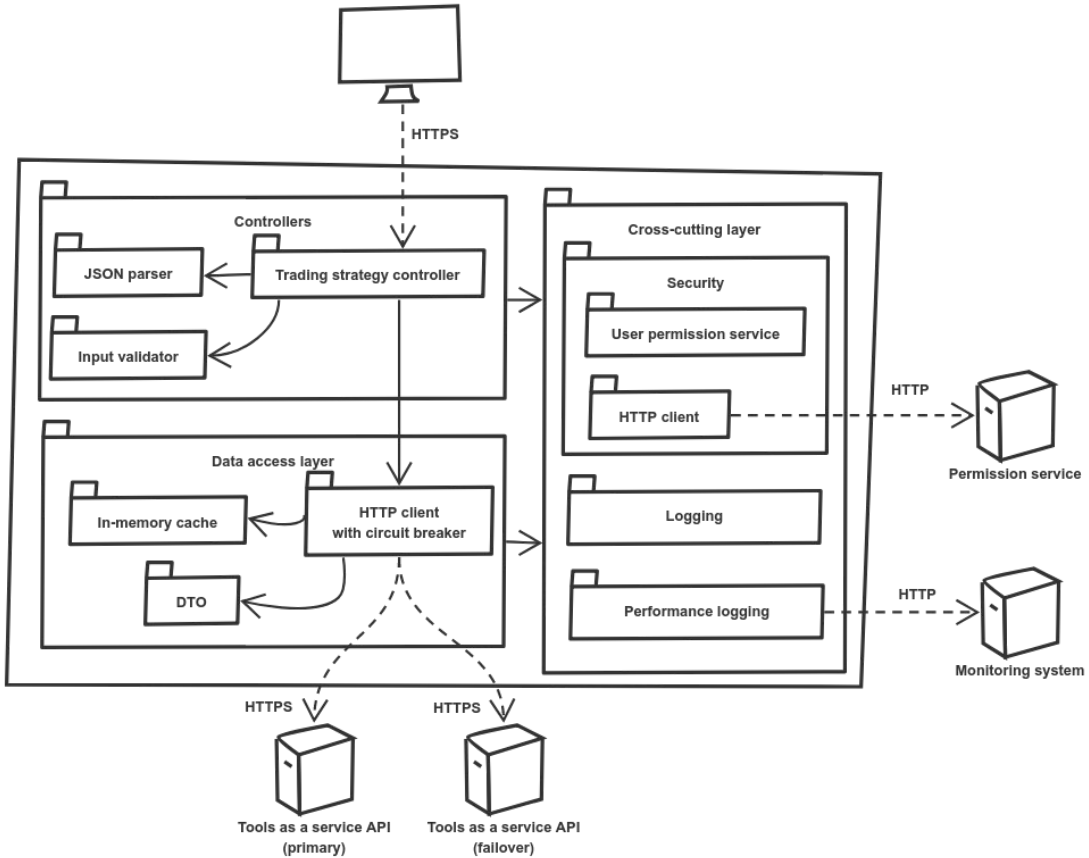




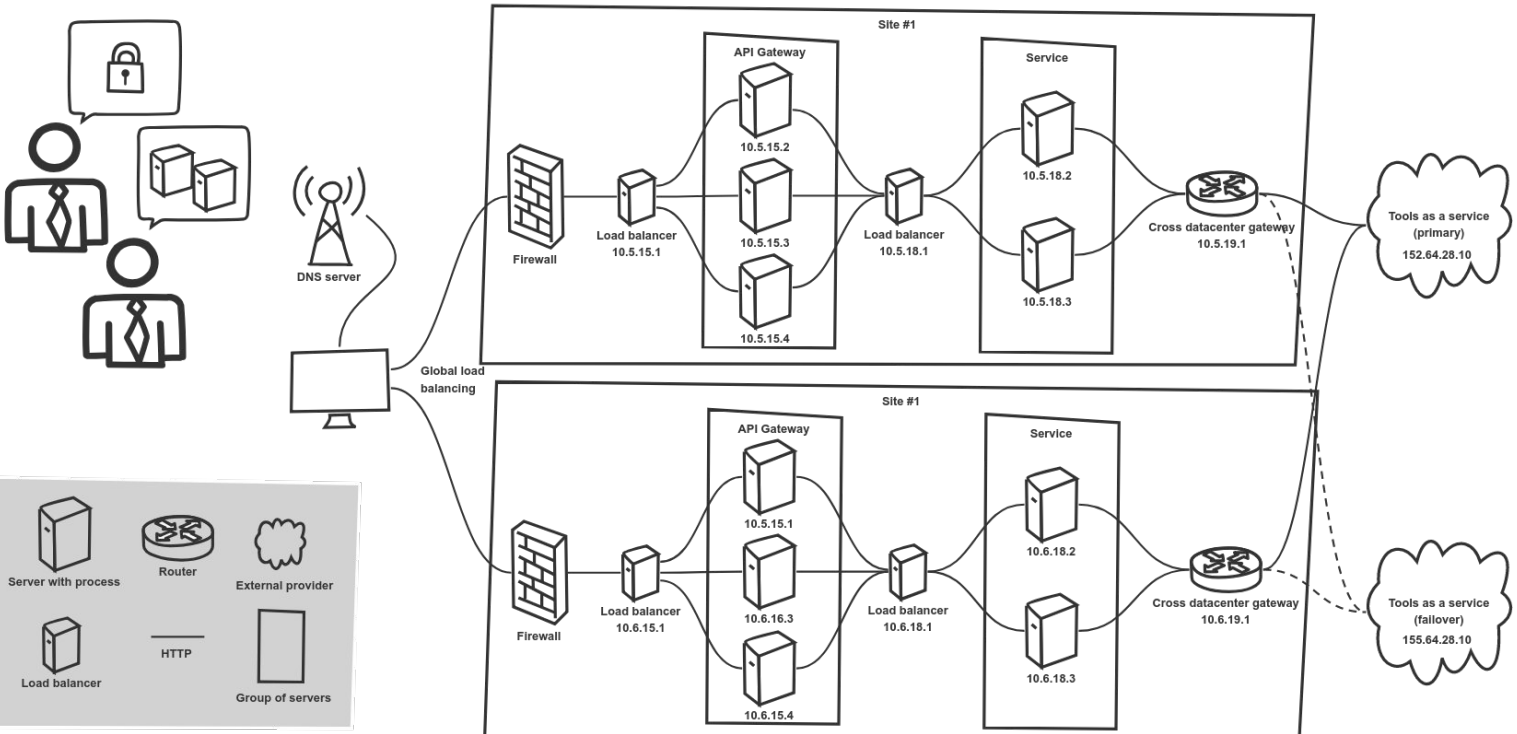
Component diagram







Deployment diagram

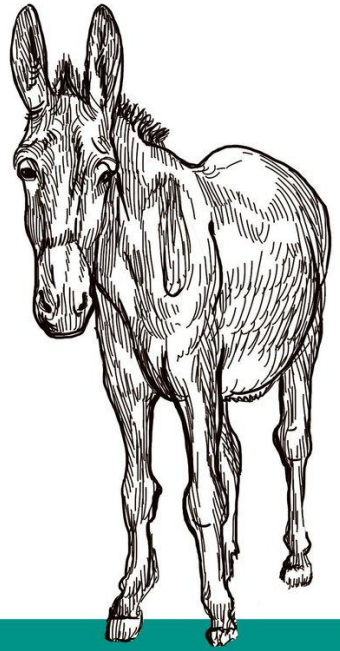


Architecture decision log

Date	Design decision	Requirement/constraint
Sep 15, 2019	Deploy on-premises as standalone services	Usage of cloud services not possible till Q3 2020, we can't wait so long
Sep 15, 2019	Use distributed cache for charts	Performance requirements
Sep 16, 2019	Deploy at least 2 instances in 2 data-centers (active) + DR (cold)	No single point of failure
Sep 16, 2019	Circuit breaker for primary and failover connections to Tools as a service provider*	No single point of failure **Tools as a service provides primary and failover endpoints*
Sep 18, 2019	Using JSON, same data format as existing API	Must stay backward-compatible with existing API as client changes are not possible

Architecture documentation

Where's the fun in just knowing what the code is supposed to do?



Essential

Excuses for Not Writing Documentation

O RLY?

@ThePracticalDev

Purpose of architecture documentation

Knowledge sharing

- Whole architecture
- Maintenance effort
- Useful to everyone at the company

Purpose of architecture documentation

Knowledge sharing

- Whole architecture
- Maintenance effort
- Useful to everyone at the company

Design review

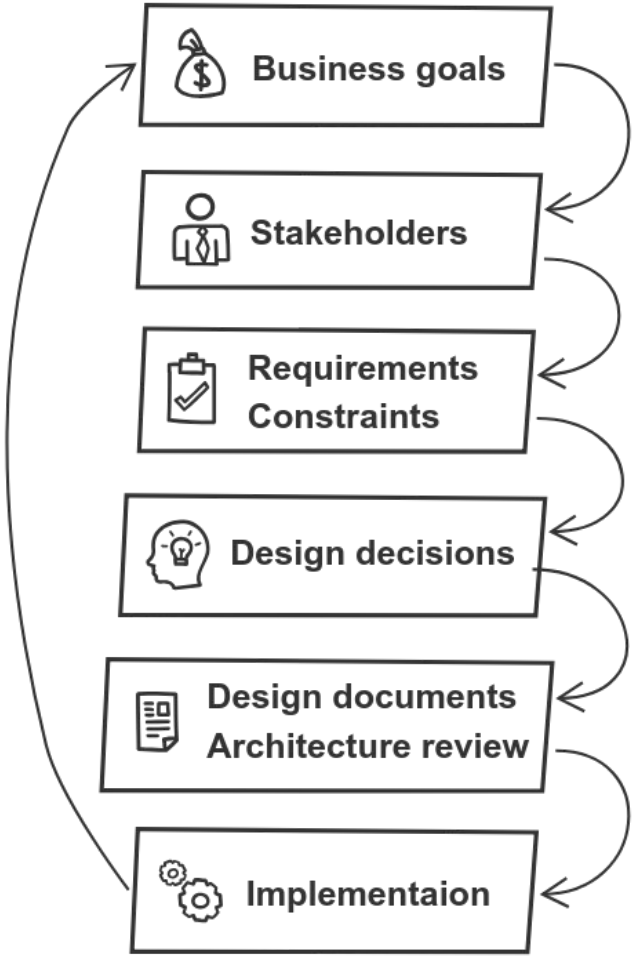
- Show what's changed
- New components
- Switching to a new version
- Rollback plan
- Alternatives

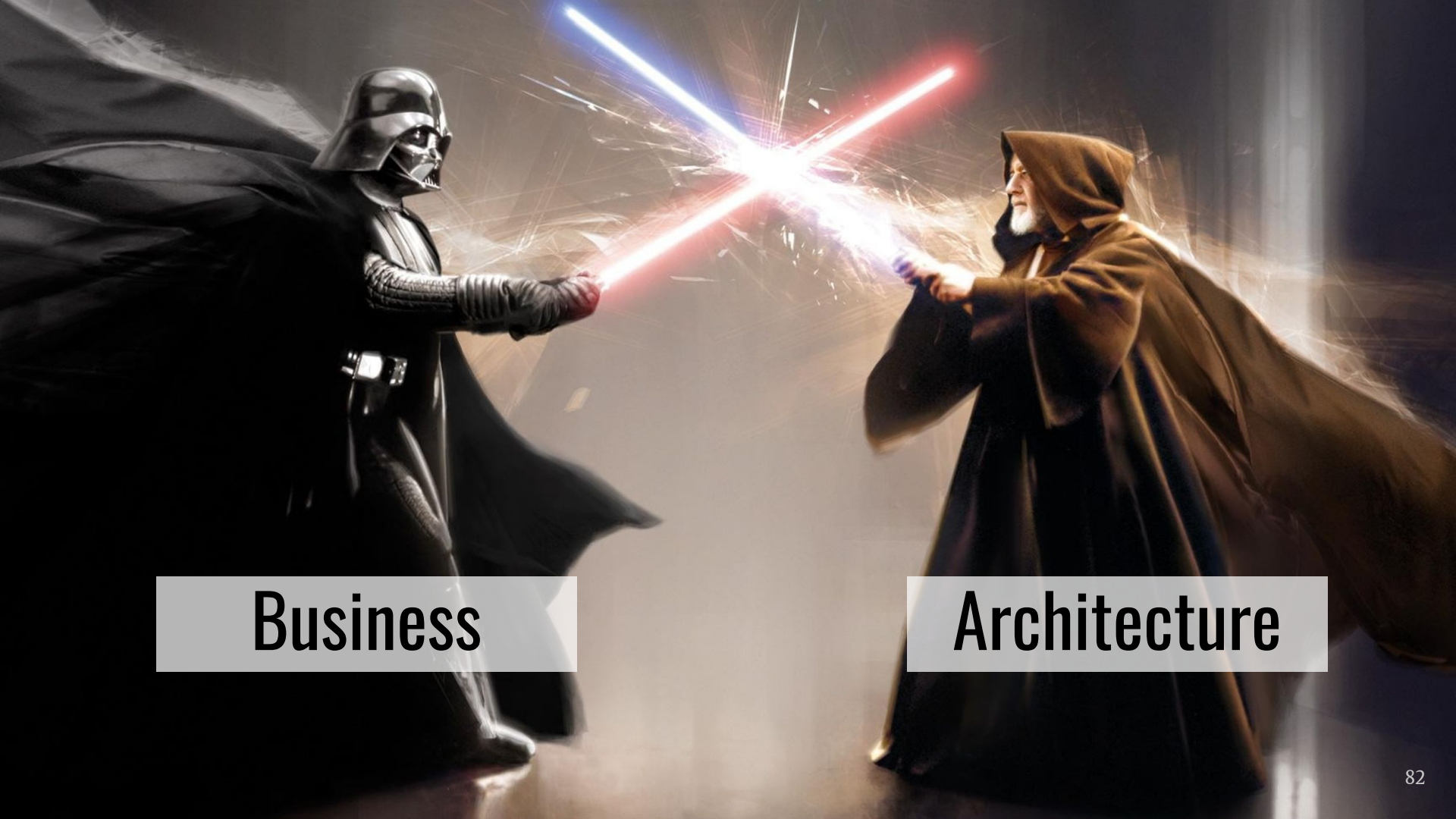
Architecture document may have

1. Business goals
2. List of stakeholders
3. Requirements, constraints
4. Assumptions
5. Views and diagrams
6. Decision log
7. ...

Conclusion

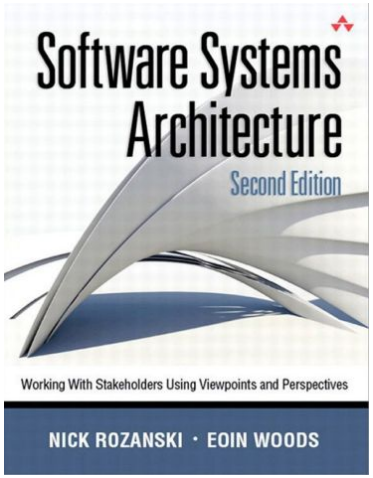
“Always in motion is the future...”

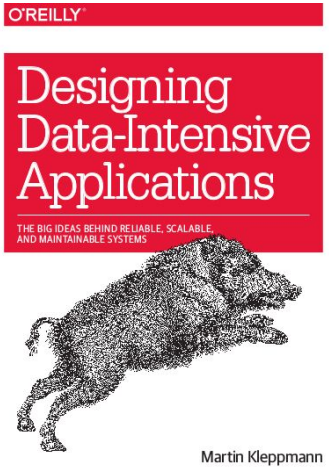

Business goals
Stakeholders
Requirements
Constraints
Design decisions
Design documents
Architecture review
Implementaion

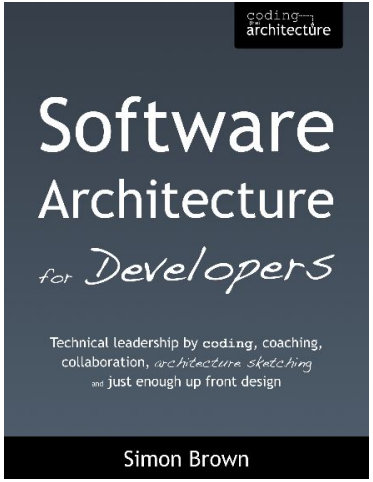

Business
Architecture

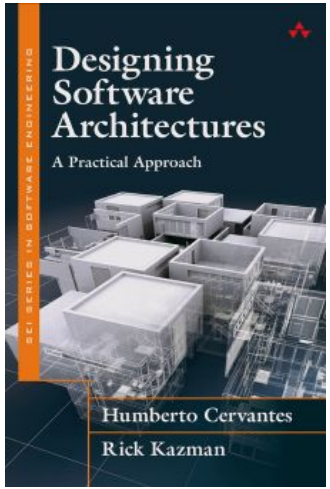
What's next?

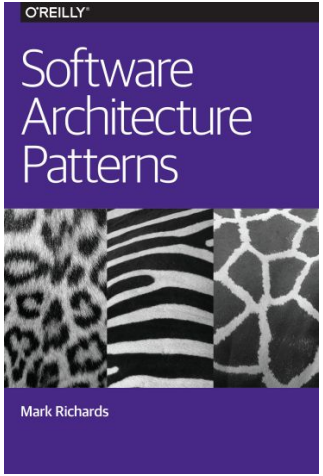

The Pragmatic Programmers
Release It!
Second Edition
Design and Deploy
Production-Ready Software
Michael T. Nygard
Edited by Katharine Dvorak


Software Systems Architecture
Second Edition
Working With Stakeholders Using Viewpoints and Perspectives
NICK ROZANSKI · EOIN WOODS


O'REILLY
Designing Data-Intensive Applications
THE BIG IDEAS BEHIND RELIABLE, SCALABLE, AND MAINTAINABLE SYSTEMS
Martin Kleppmann


coding the architecture
Software Architecture for Developers
Technical leadership by coding, coaching, collaboration, architecture sketching and just enough up front design
Simon Brown


SEI SERIES IN SOFTWARE ENGINEERING
Designing Software Architectures
A Practical Approach
Humberto Cervantes
Rick Kazman


O'REILLY
Software Architecture Patterns
Mark Richards

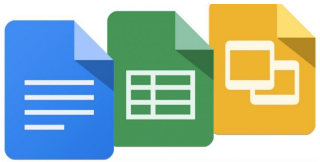
Diagrams

Draw.io

MS Visio

Sketchboard.io

Documents



Google Docs

Confluence

Bitbucket

Questions?

Konstantin Slisenko
Solution Architect at EPAM

twitter: @kslisenko
facebook.com/konstantin.slisenko.1