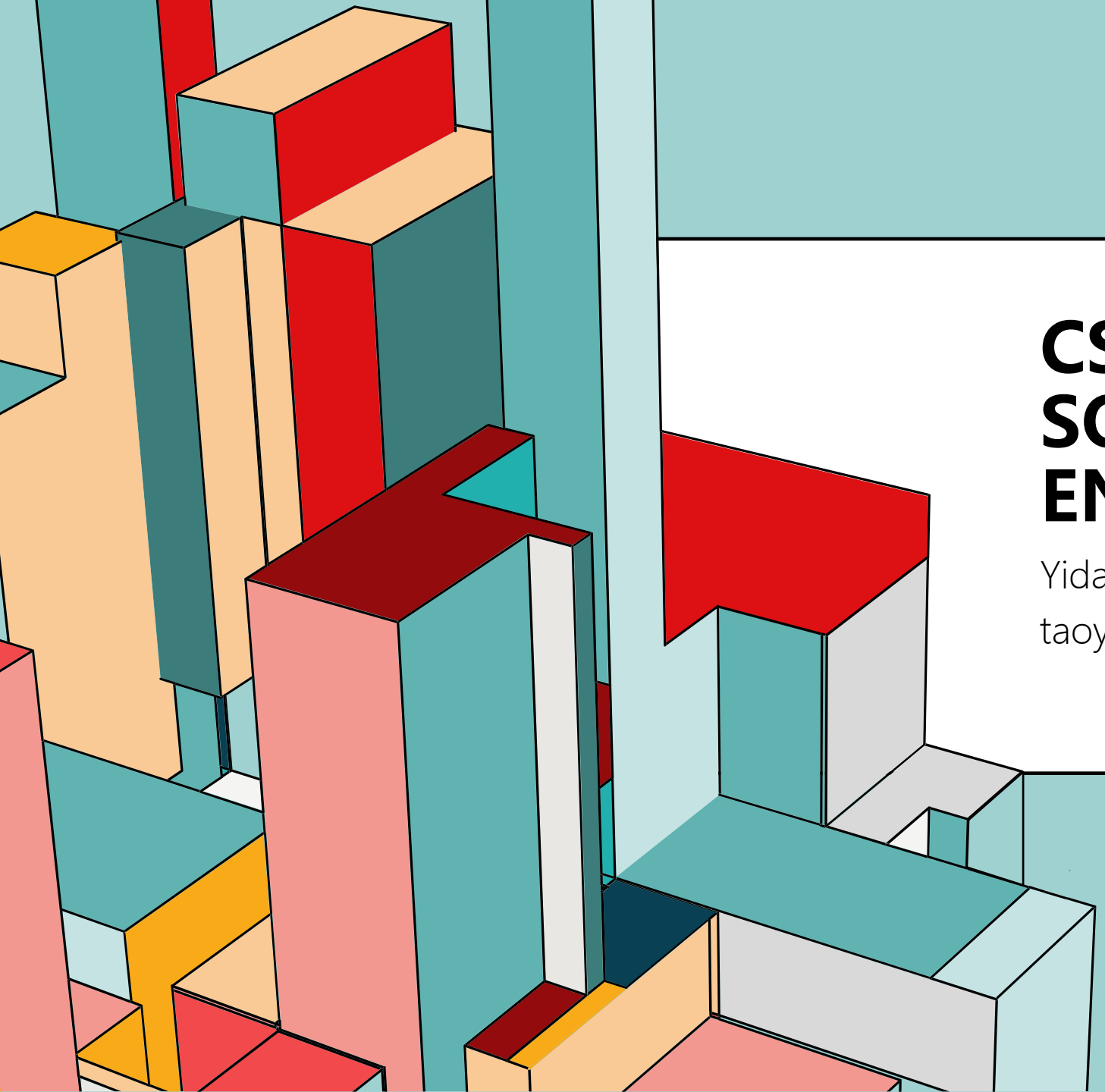


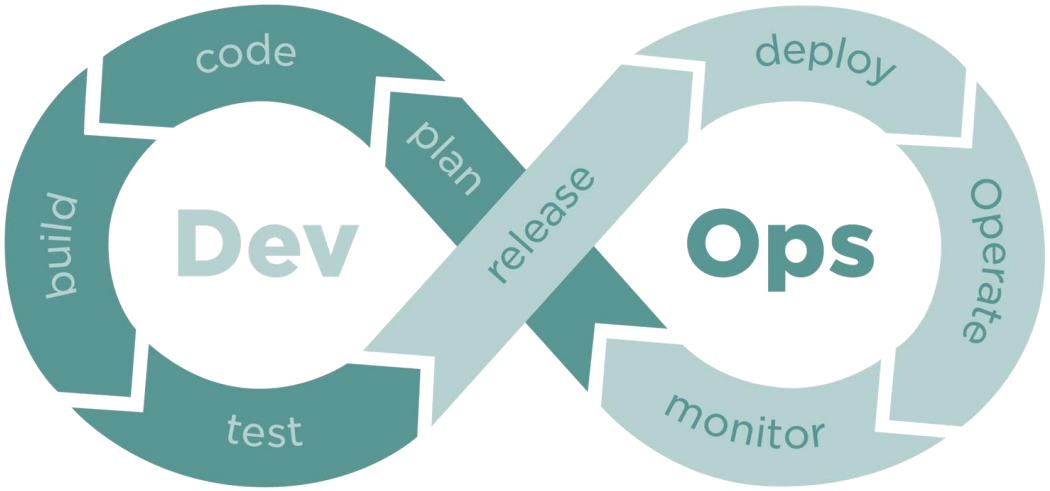
CS304 SOFTWARE ENGINEERING

Yida Tao
taoyd@sustech.edu.cn

WHERE ARE WE NOW?

Plan

- Requirement analysis
- **Design**



WHAT IS DESIGN?

It’s where you stand with a foot in two worlds—the world of technology and the world of people and human purposes—and you try to bring the two together.

- Mitch Kapor, “software design manifesto”

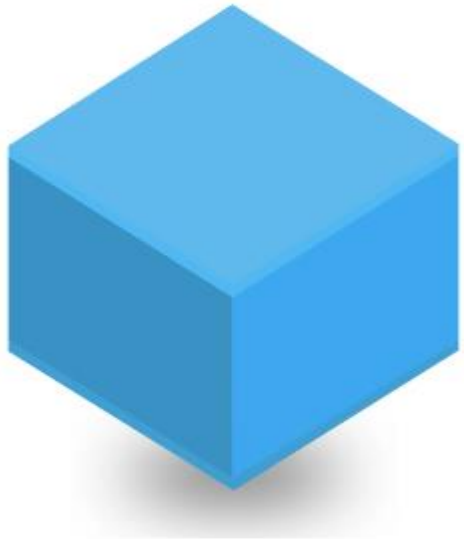
SOFTWARE DESIGN

- **Architectural design (our focus in this lecture)**
- **User interface design (briefly)**
- Data design
- General design concepts
- ……

ARCHITECTURAL STYLE

SOFTWARE ARCHITECTURAL STYLE

Classic, Monolithic

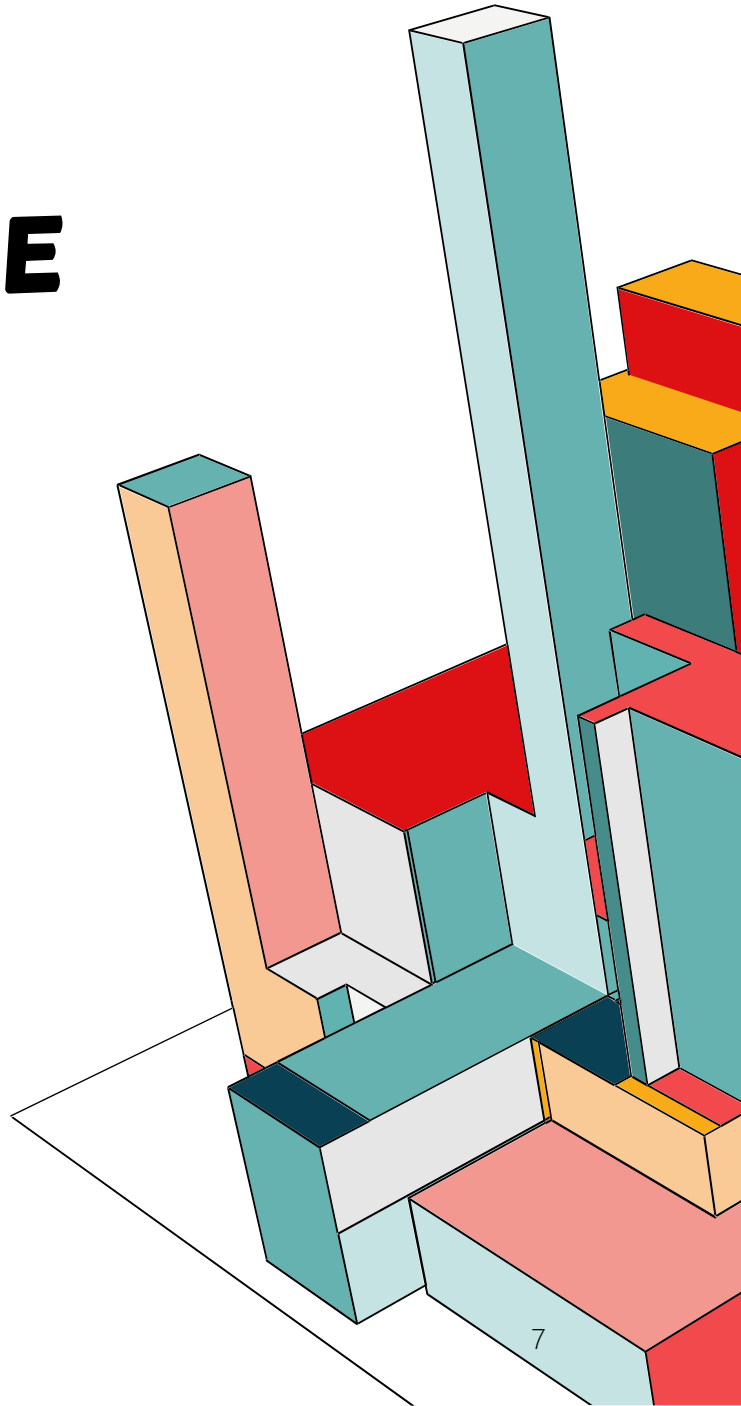


Service-based, Distributed, DevOps

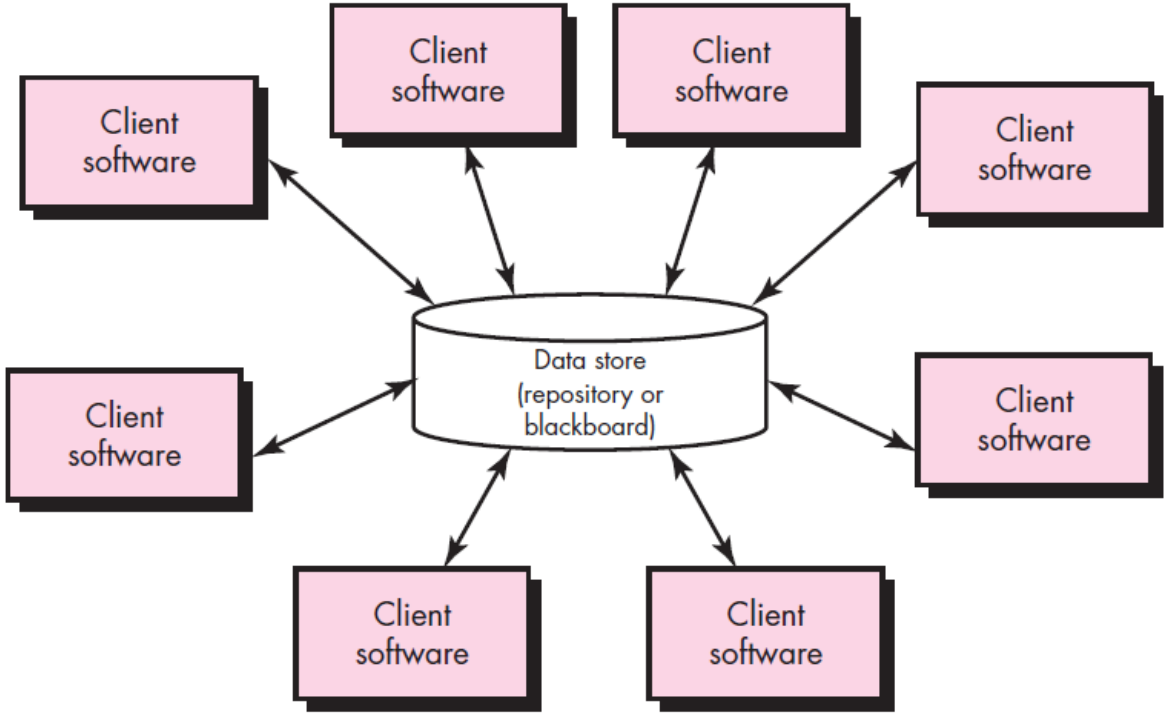
SOFTWARE ARCHITECTURAL STYLE

- Classic, Monolithic
  - Data-centered architecture
  - Data-flow architecture
  - Call-and-return architecture
  - Object-oriented architecture
  - Layered architecture

- Service-based, Distributed, DevOps
  - Microkernel architecture
  - Event-driven architecture
  - Microservice architecture



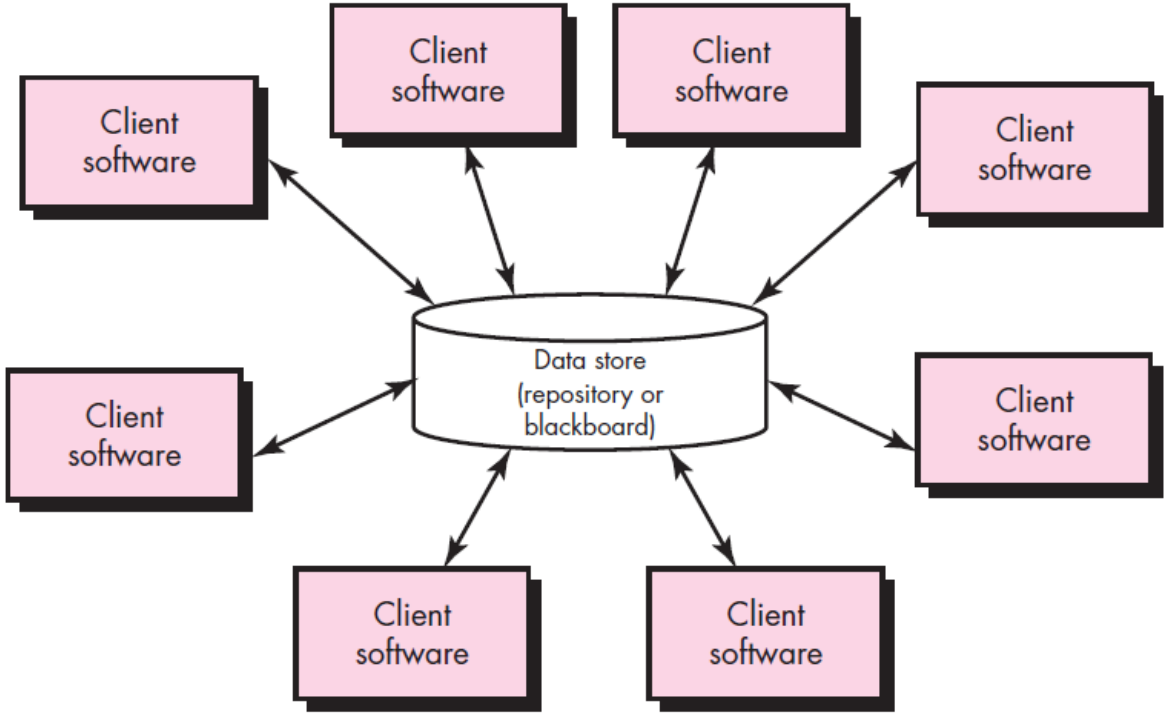
DATA-CENTERED ARCHITECTURES



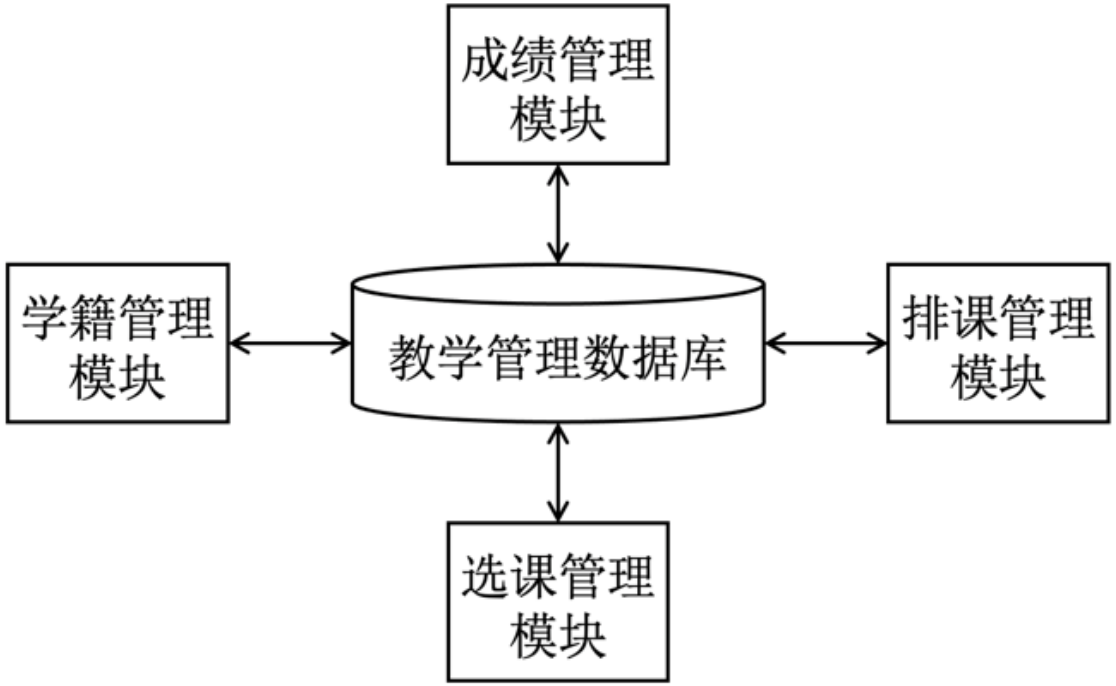
以数据为中心的体系结构

- A data store (e.g., a file or database) resides at the center of this architecture
- The data store is accessed frequently by other components that update, add, delete, or modify data within the store

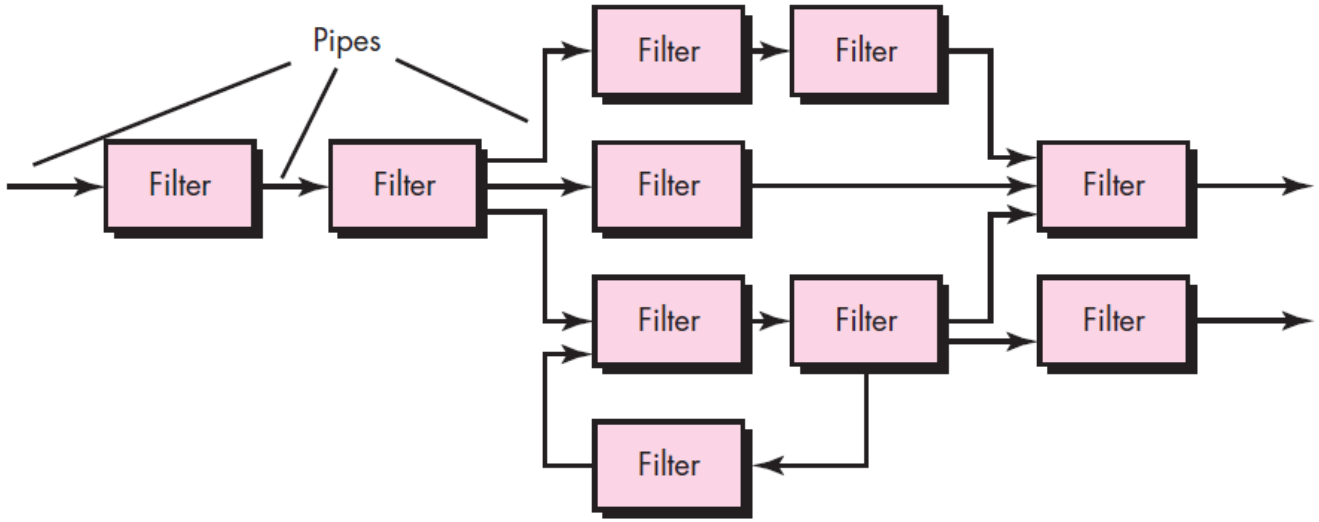
DATA-CENTERED ARCHITECTURES



以数据为中心的体系结构



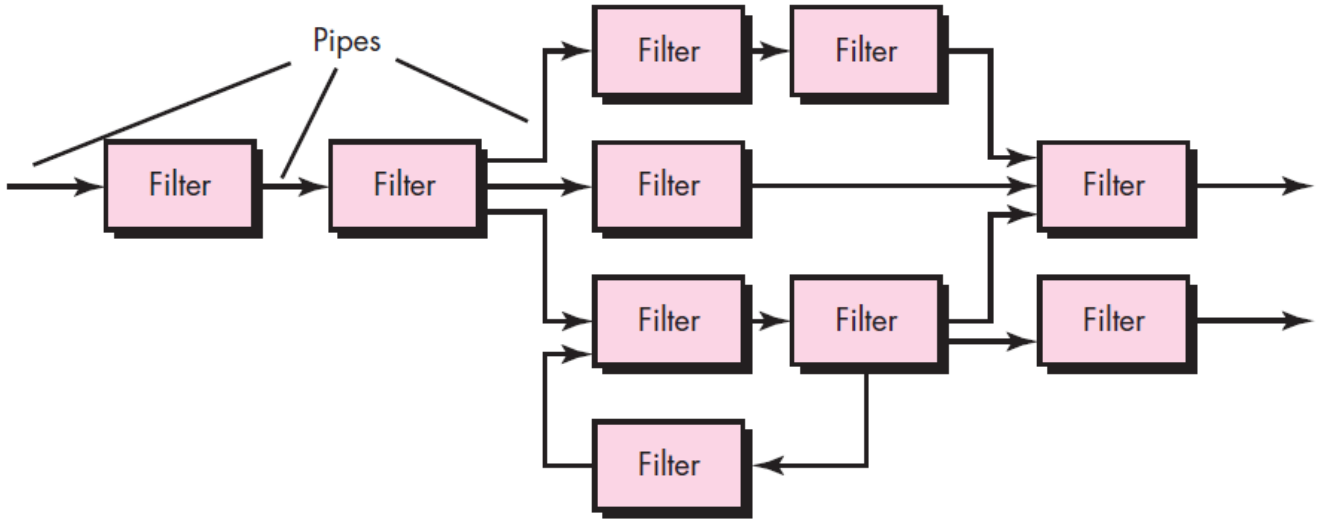
DATA-FLOW ARCHITECTURES



数据流体系结构

This architecture is applied when input data are to be transformed through a series of computational or manipulative components into output data.

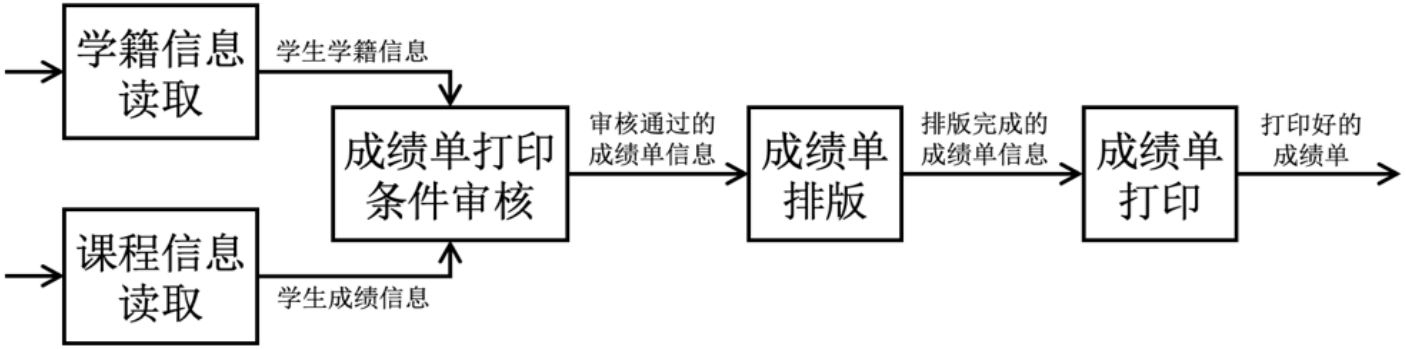
DATA-FLOW ARCHITECTURES



数据流体系结构（管道和过滤器）

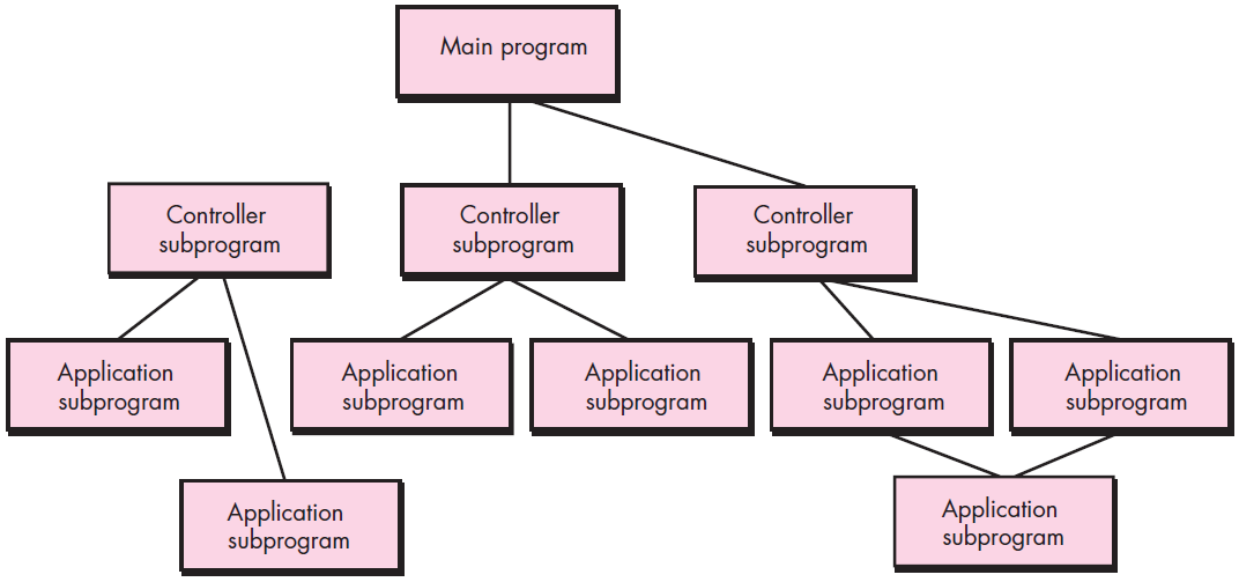
- A pipe-and-filter pattern has a set of components, called filters, connected by pipes that transmit data from one component to the next.
- Each filter works independently of those components upstream and downstream, is designed to expect data input of a certain form, and produces data output (to the next filter) of a specified form.
- The filter does not require knowledge of the workings of its neighboring filters.

DATA-FLOW ARCHITECTURES



- Data-flow architecture is suitable for automated data analysis and transmission systems
- Such systems contain a series of data analysis components, with almost no user interaction
- Data-flow architecture may not be suitable for GUI intensive systems

CALL AND RETURN ARCHITECTURES



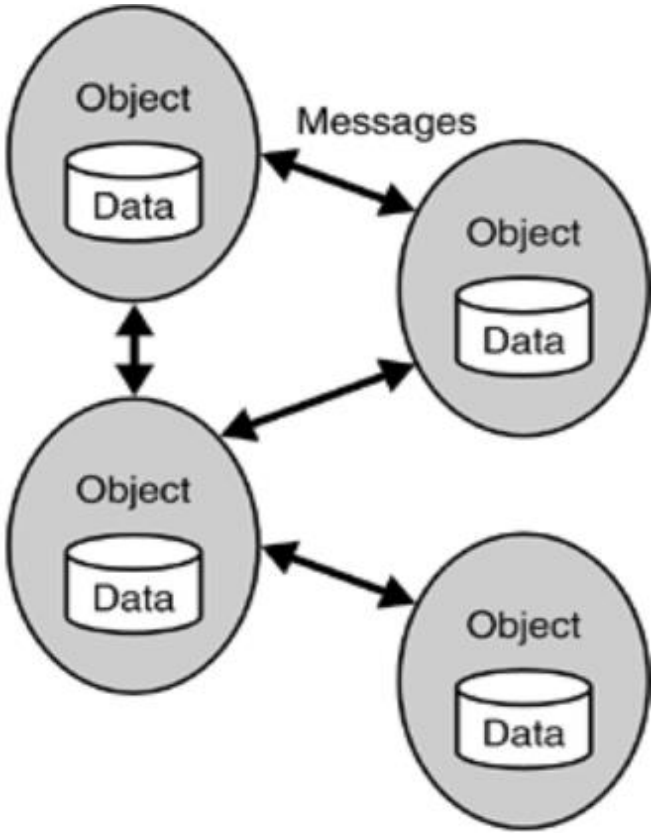
主程序/子程序体系结构

- **Main program/subprogram architecture** decomposes function into a control hierarchy where a “main” program invokes program components that in turn may invoke still other components.

- **Remote procedure call architecture**: The components of a main program and subprogram architecture are distributed across multiple computers on a network.

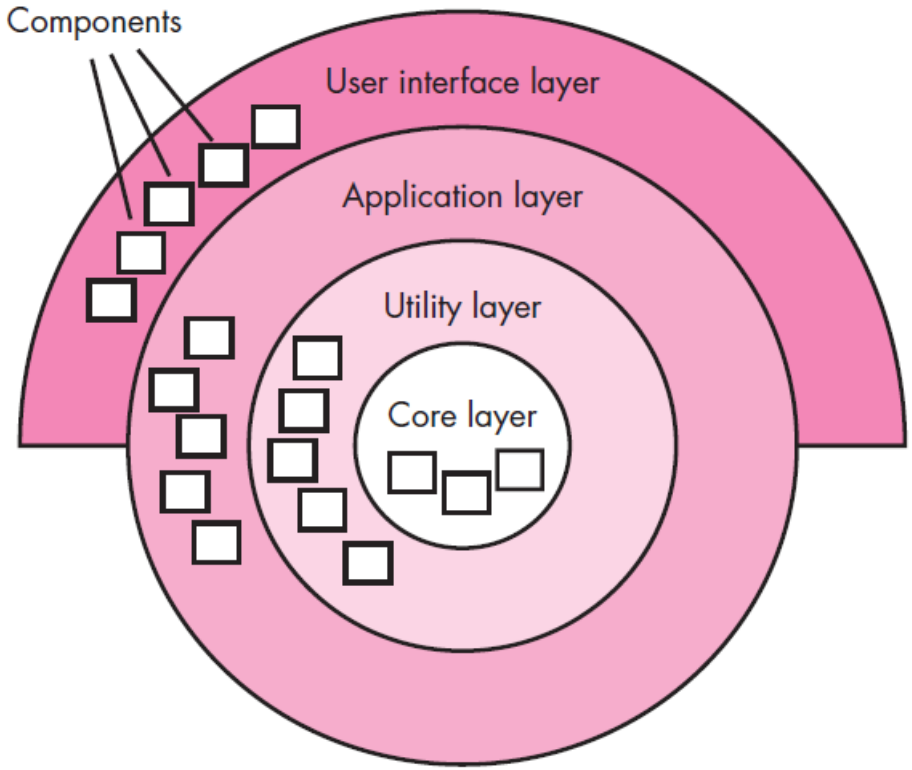
OBJECT-ORIENTED ARCHITECTURES

- The components of a system encapsulate data and the operations that must be applied to manipulate the data.
- Communication and coordination between components are accomplished via message passing

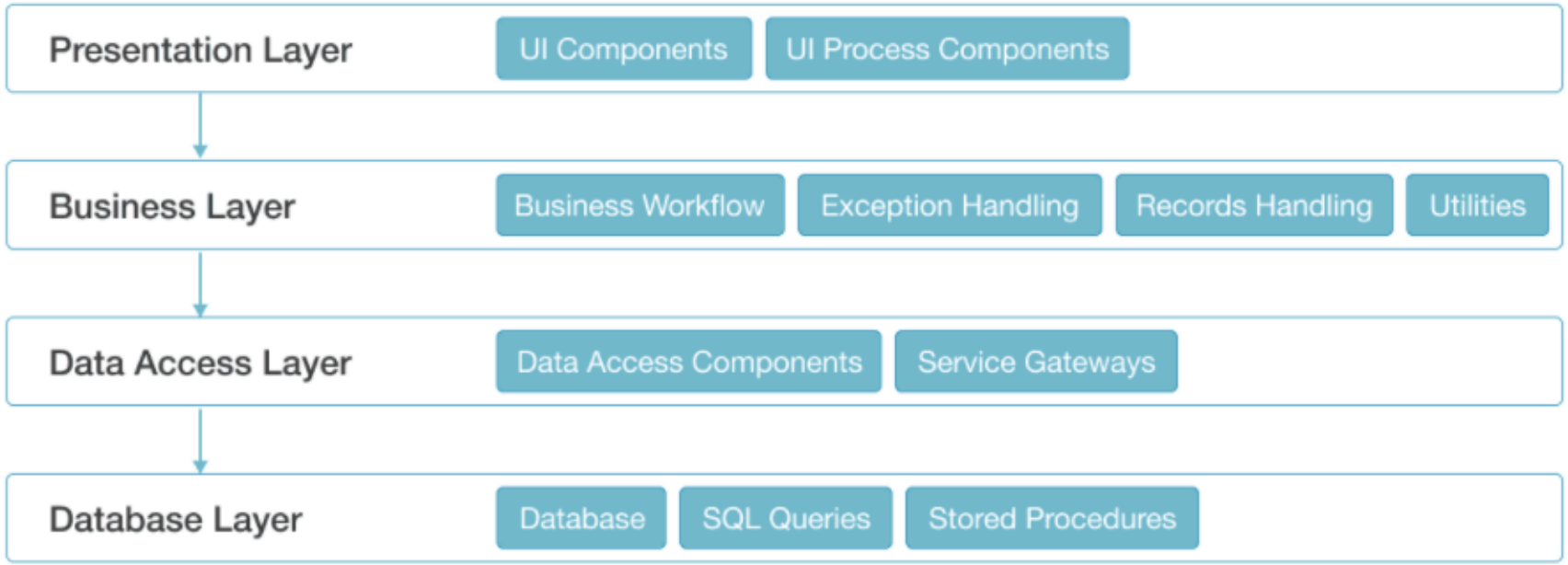


LAYERED ARCHITECTURES

- Different layers are defined, each accomplishing operations that progressively become closer to the machine instruction set.
- At the outer layer, components service user interface operations.
- At the inner layer, components perform operating system interfacing.
- Intermediate layers provide utility services and application software functions.



LAYERED ARCHITECTURES



用户界面层

服务接口层

业务逻辑层

数据访问层

数据库层

Layered architecture for web applications

https://www.simform.com/blog/web-application-architecture/

PROBLEMS WITH MONOLITHIC?

Netflix users suffering service's longest outage ever

An undisclosed error has taken all 55 Netflix distribution centers offline. Customer-facing site remains up.

Rafe Needleman

Aug. 15, 2008 11:04 a.m. PT

2 min read

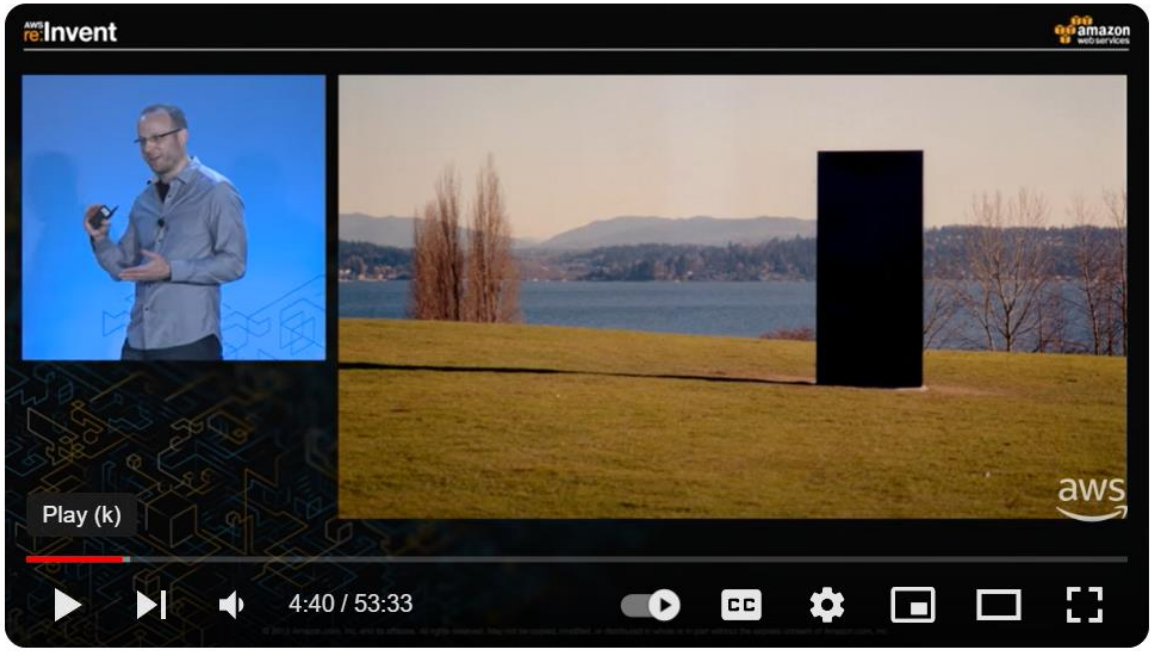
Netflix, and its customers, are currently experiencing the worst system failure in the DVD shipper's history. The company shipped no discs on Tuesday, only "a few" yesterday, and so far none today. Affected customers are receiving e-mails telling them their discs are delayed.

In 2009, Netflix faced growing pains as its **monolithic architecture** struggled to cope with the demand for video streaming services.

Determined to emerge victorious, Netflix migrated to a **cloud-based microservices architecture**, paving the way for its rapid rise.

PROBLEMS WITH MONOLITHIC?

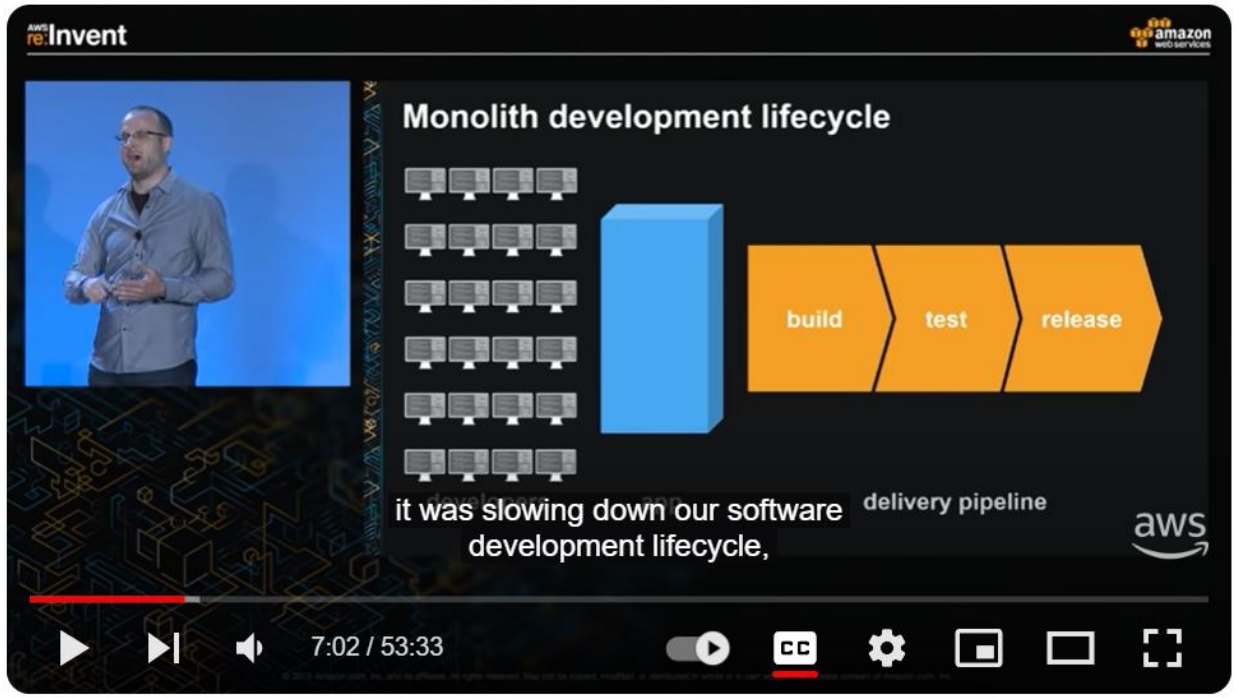
"If you go back to 2001," stated Amazon AWS senior manager for product management Rob Brigham, "the Amazon.com retail website was a large **architectural monolith**."



AWS re:Invent 2015: DevOps at Amazon: A Look at Our Tools and Processes (DVO202)

PROBLEMS WITH MONOLITHIC?

"Monolithic architecture adds large overhead to the process, frustrates developers, and slows down the entire software development lifecycle."



AWS re:Invent 2015: DevOps at Amazon: A Look at Our Tools and Processes (DVO202)

PROBLEMS WITH MONOLITHIC?

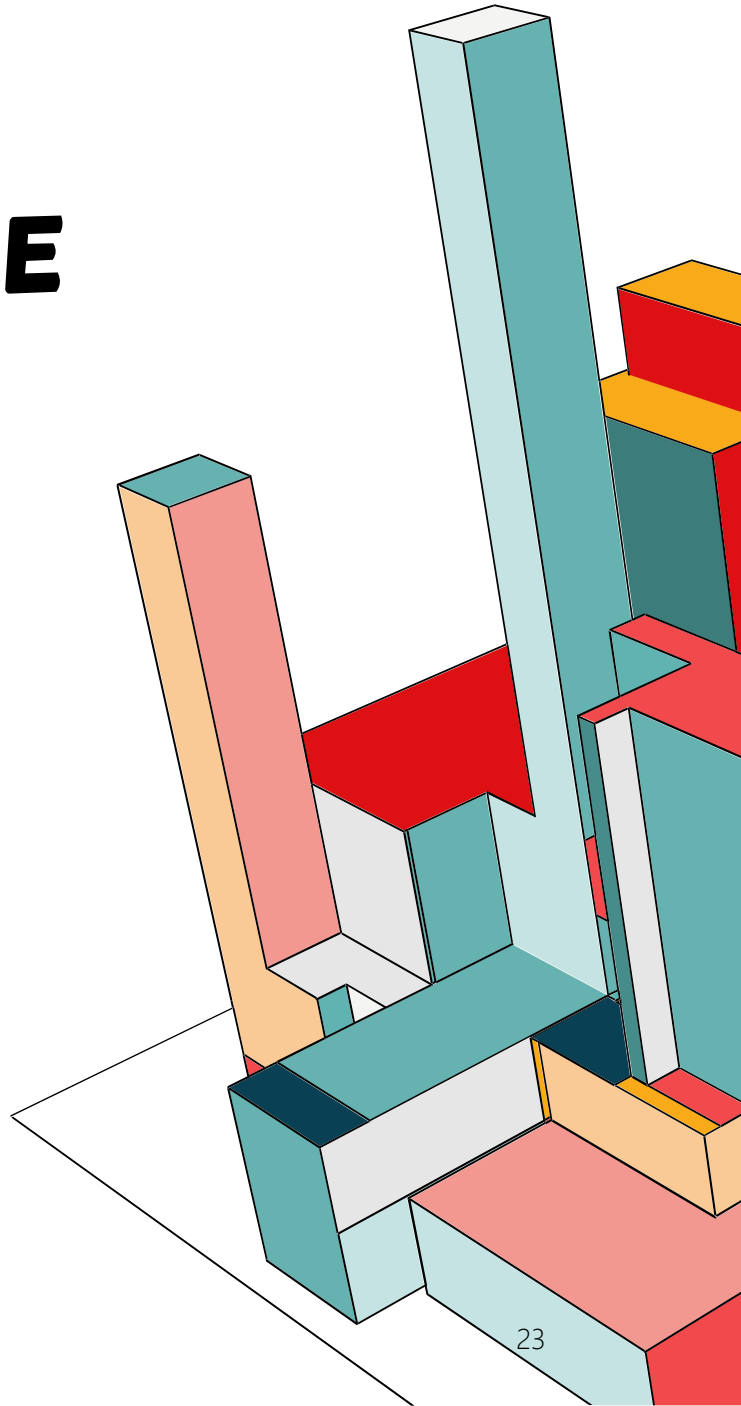
"We teased it apart into service-oriented architecture."

AWS re:Invent 2015: DevOps at Amazon: A Look at Our Tools and Processes (DVO202)

SOFTWARE ARCHITECTURAL STYLE

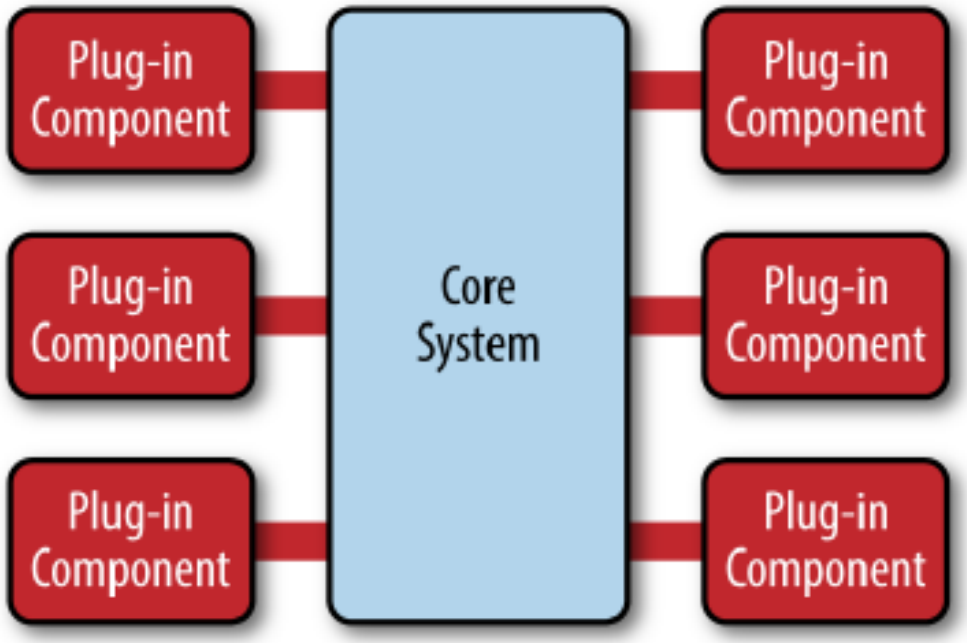
- Classic, Monolithic
  - Data-centered architecture
  - Data-flow architecture
  - Call-and-return architecture
  - Object-oriented architecture
  - Layered architecture

- Service-based, Distributed, DevOps
  - Microkernel architecture
  - Event-driven architecture
  - Microservice architecture



MICROKERNEL ARCHITECTURES

- **Core system**: only the minimal functionality required to make the system operational
- **Plug-in component**: stand-alone, independent components that contain specialized processing, additional features, and custom code that is meant to enhance or extend the core system to produce additional business capabilities.
- Also referred to as the **plug-in architecture pattern**

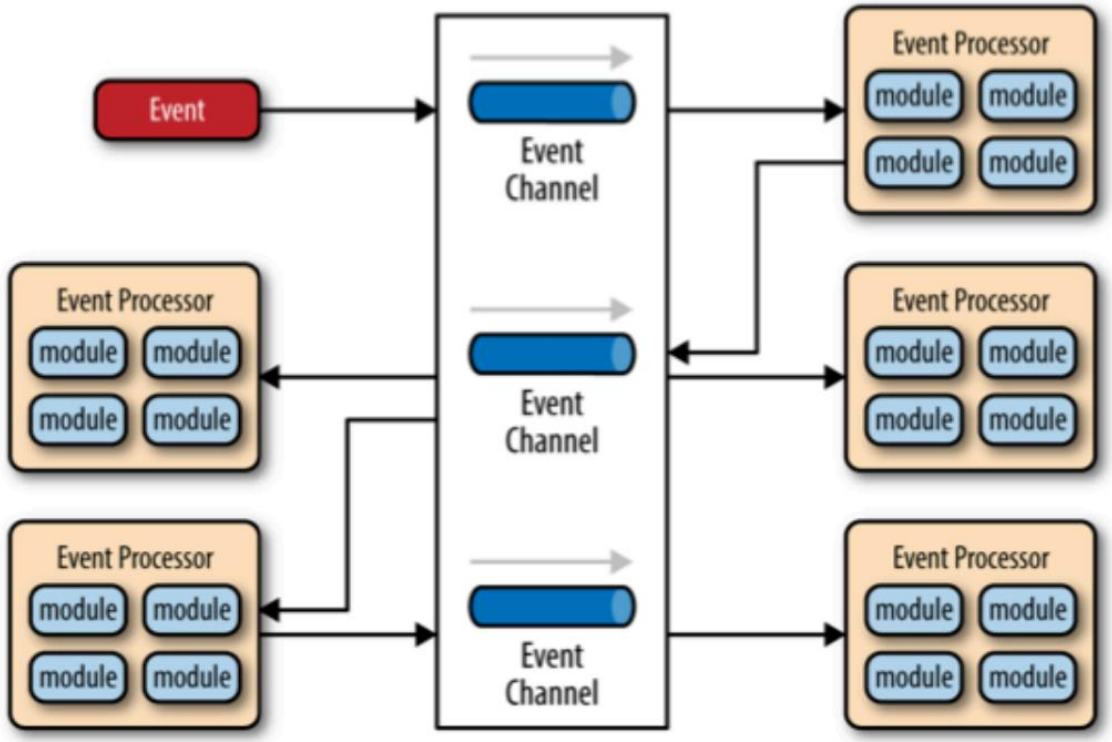


Mark Richards. 2015. Software Architecture Patterns.

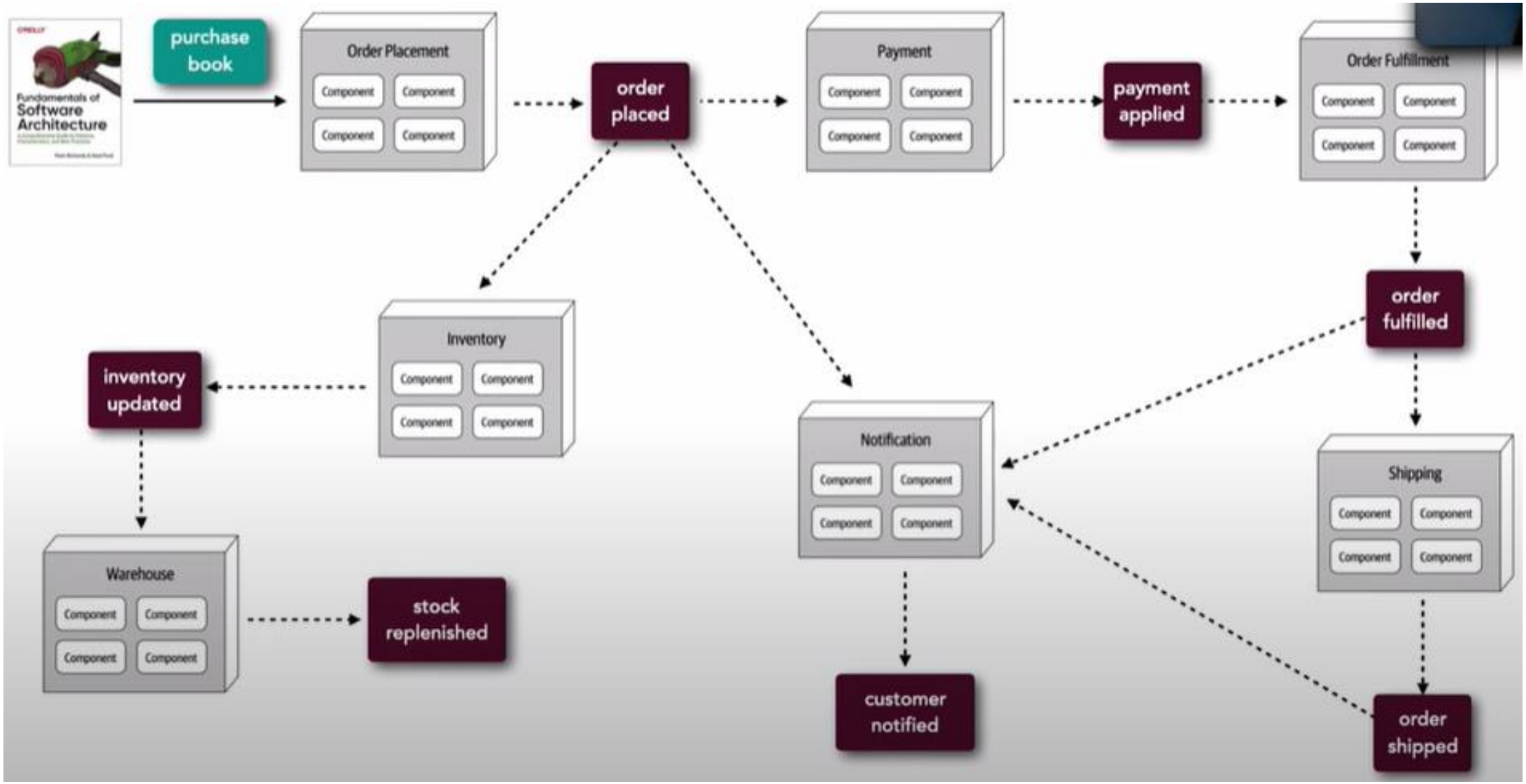
MICROKERNEL ARCHITECTURES

EVENT-DRIVEN ARCHITECTURE

- Core Components:
  - **Event Producer**: Initiates and generates events.
  - **Event Consumer**: Reacts to and processes events.
  - **Event channel/mediator/broker**: orchestration
- Asynchronous and distributed
- Promote the production, detection, consumption, and reaction to events



Mark Richards. 2015. Software Architecture Patterns.



https://www.youtube.com/watch?v=P0aUV4ixvBQ

MICROSERVICE ARCHITECTURE

The microservice architectural style is an approach to developing a single application as a suite of small services, each running in its own process and communicating with lightweight mechanisms, often an HTTP resource API.

These services are built around business capabilities and independently deployable by fully automated deployment machinery.

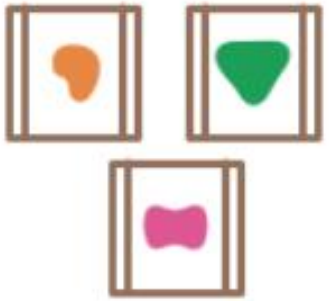
https://martinfowler.com/articles/microservices.html

MICROSERVICE ARCHITECTURE

Services

A monolithic application puts all its functionality into a single process...

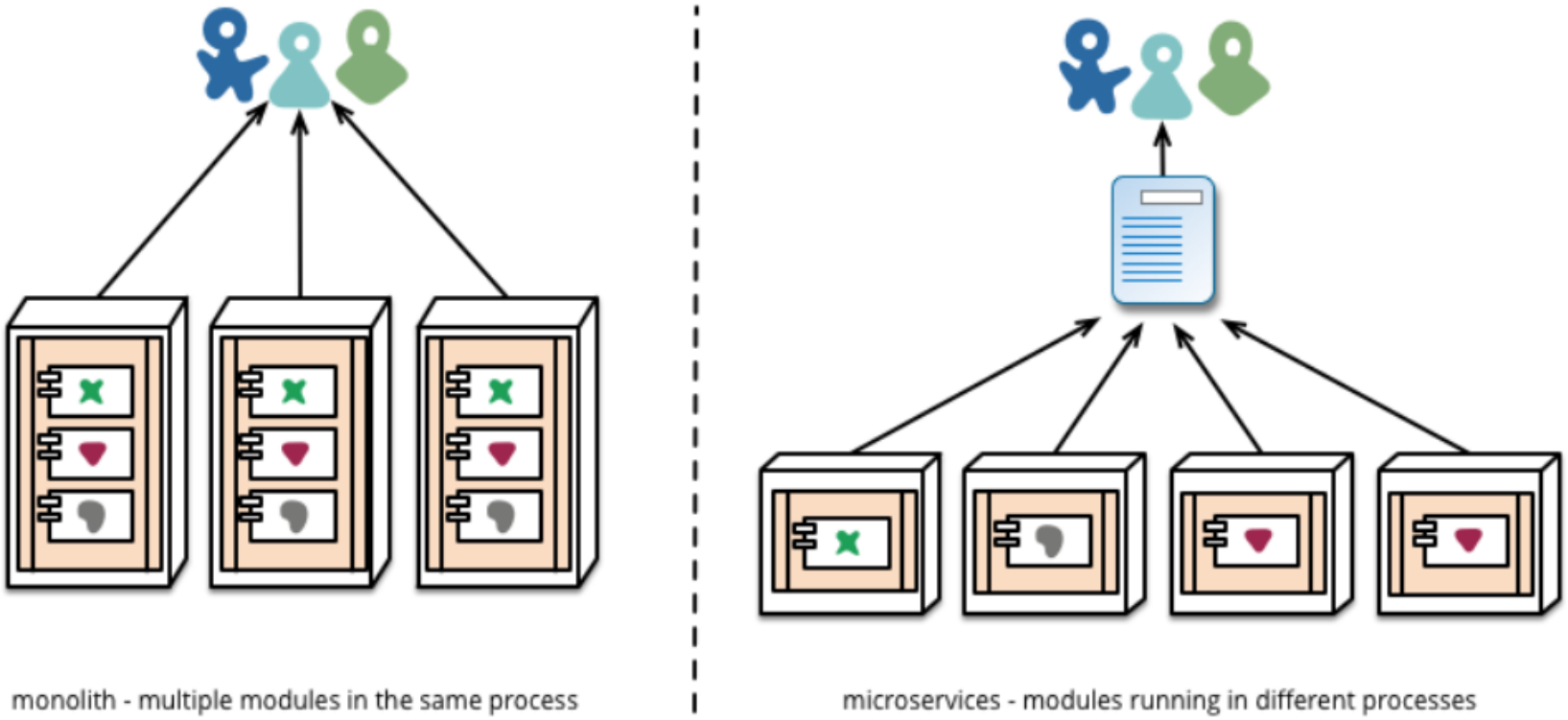
A microservices architecture puts each element of functionality into a separate service...



https://martinfowler.com/articles/microservices.html

MICROSERVICE ARCHITECTURE

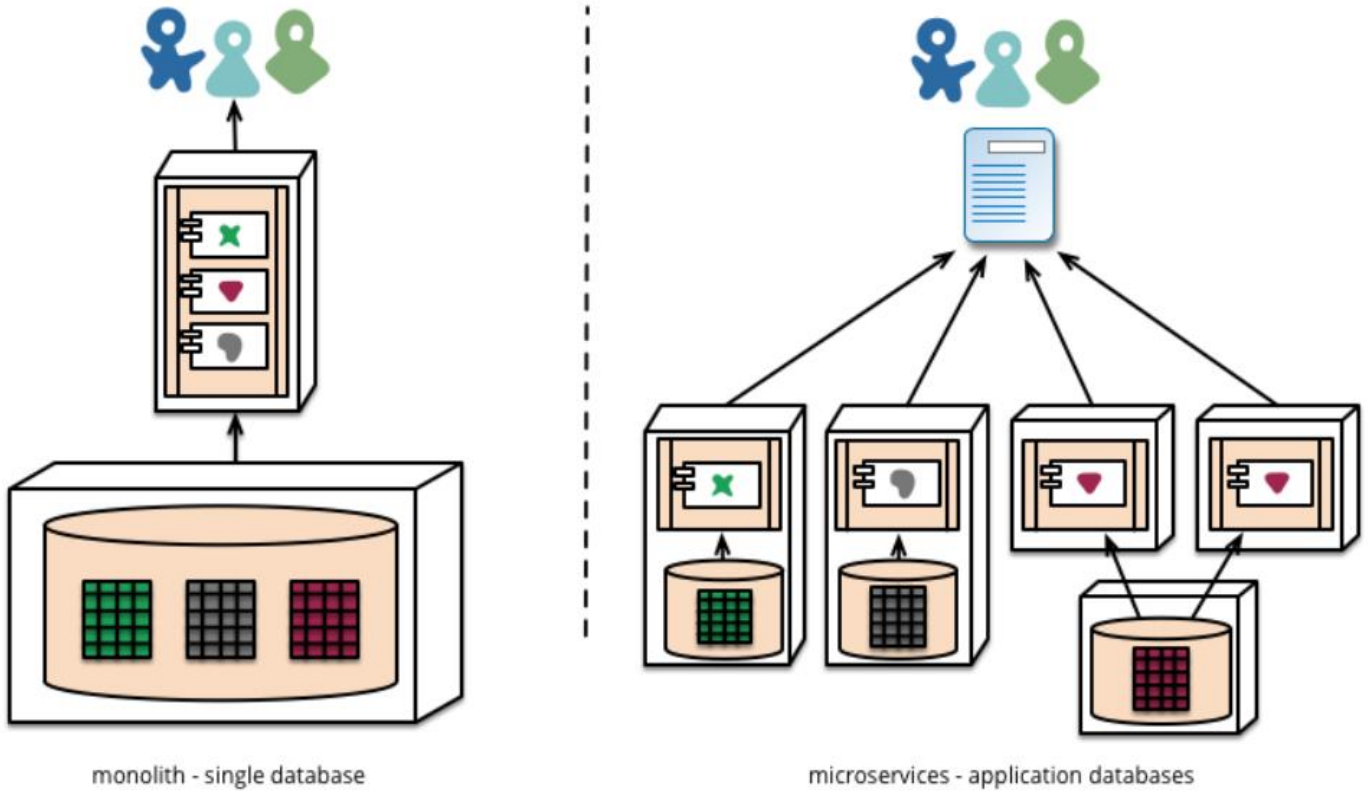
Deployment



https://martinfowler.com/articles/microservices.html

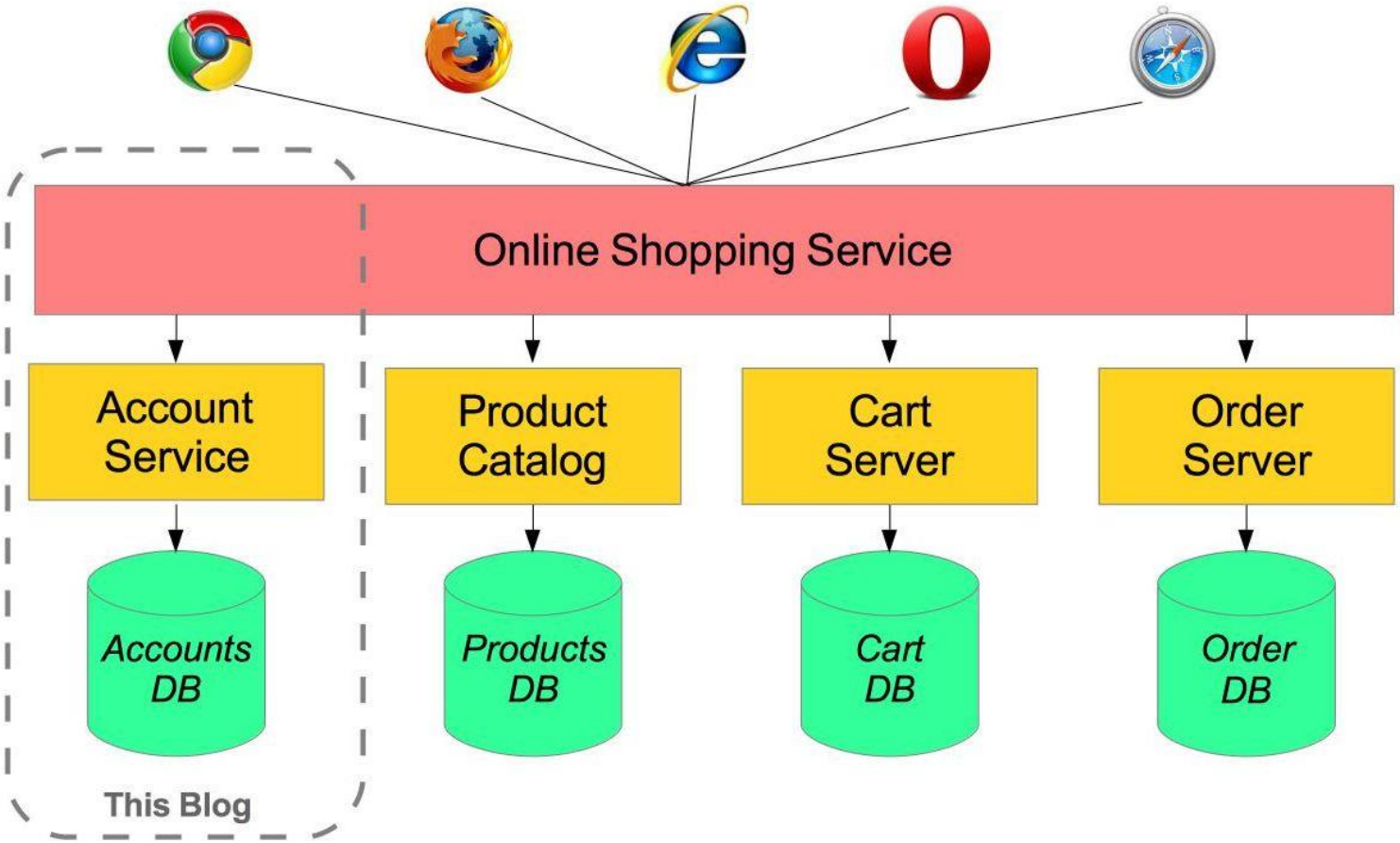
MICROSERVICE ARCHITECTURE

Decentralized Data Management



https://martinfowler.com/articles/microservices.html

MICROSERVICE ARCHITECTURE



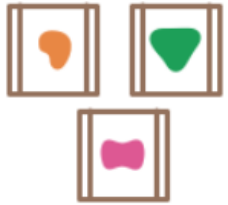
https://spring.io/blog/2015/07/14/microservices-with-spring

MICROSERVICE ARCHITECTURE

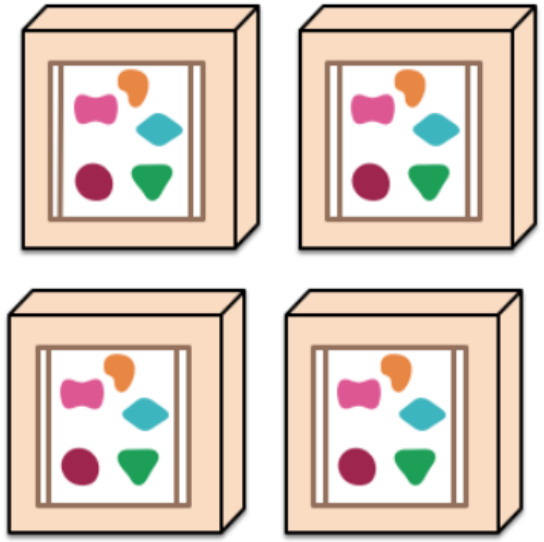
Scale

A monolithic application puts all its functionality into a single process...

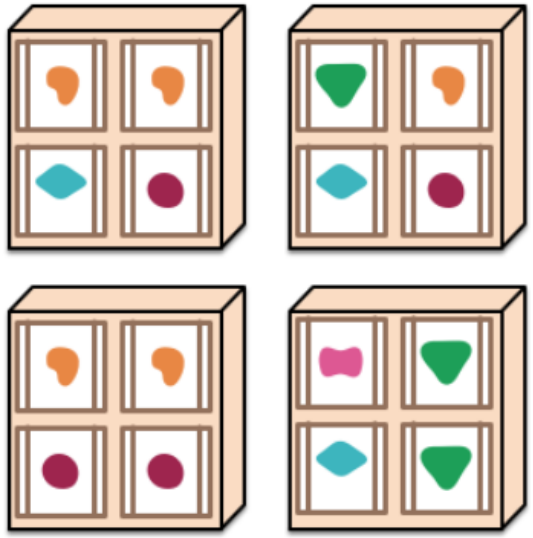
A microservices architecture puts each element of functionality into a separate service...



... and scales by replicating the monolith on multiple servers



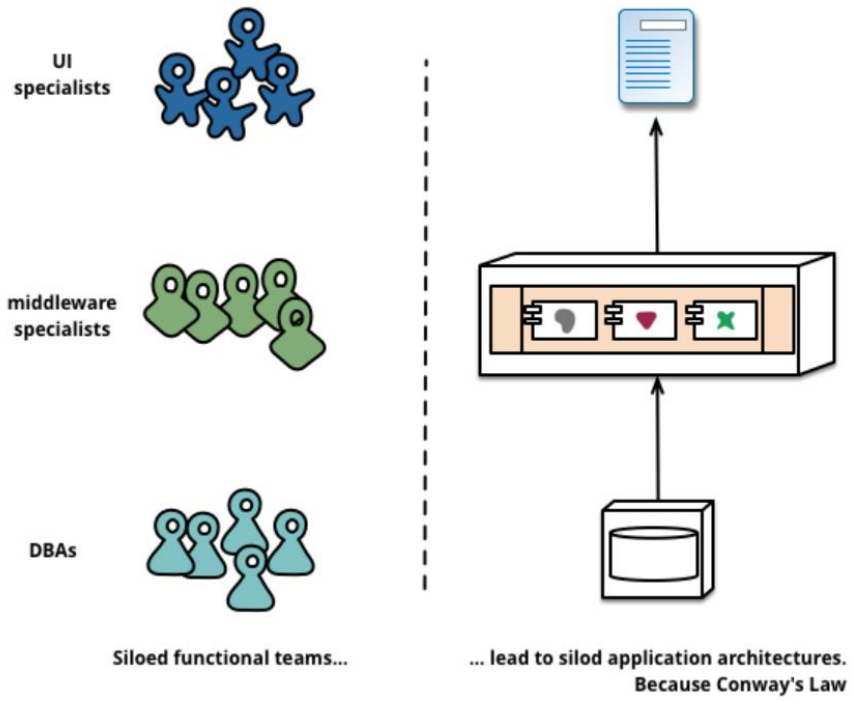
... and scales by distributing these services across servers, replicating as needed.



https://martinfowler.com/articles/microservices.html

MICROSERVICE ARCHITECTURE

Team organization



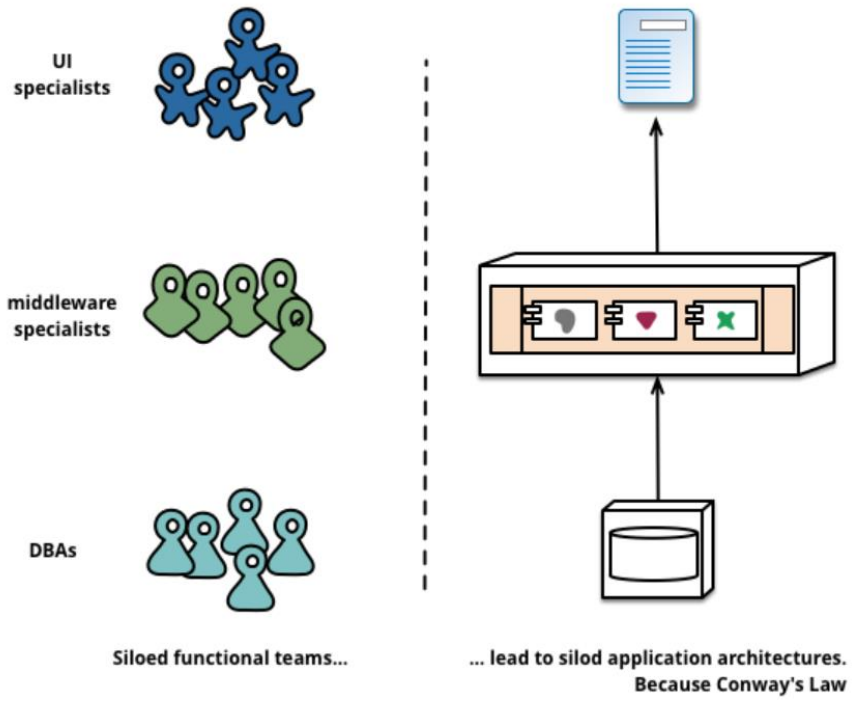
Monolithic (layered)

Conway's Law
"Any organization that designs a system will produce a design whose structure is a copy of the organization's communication structure."

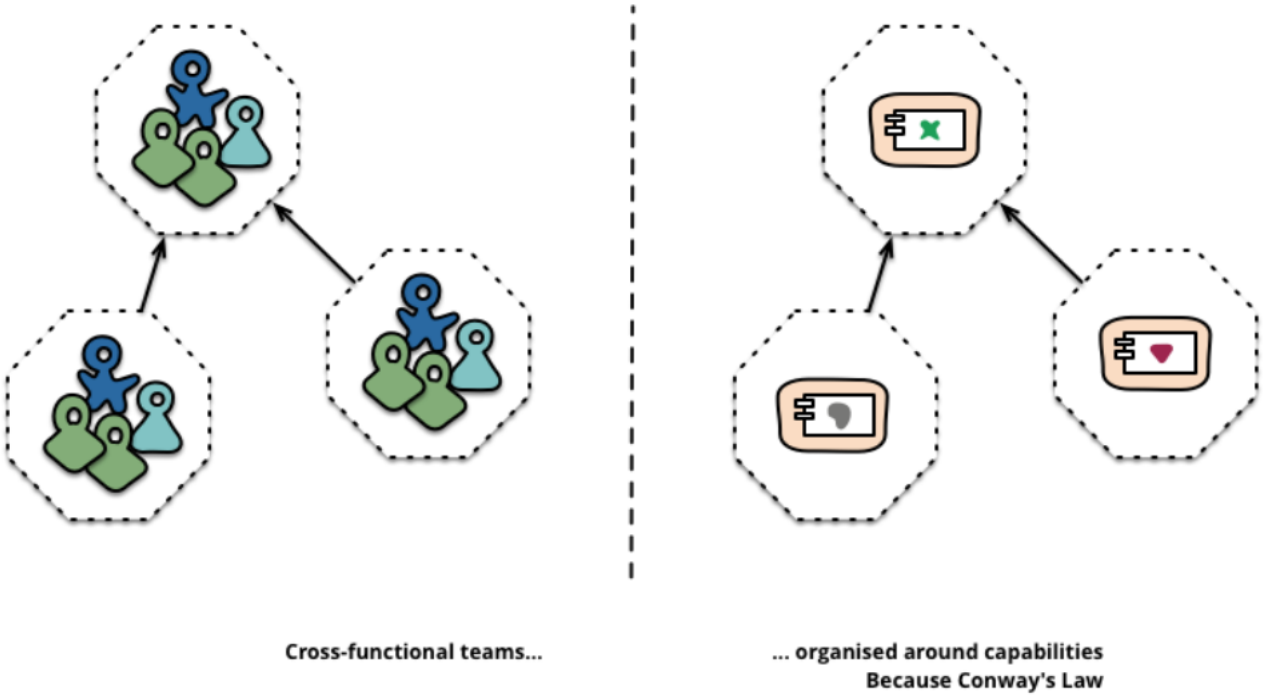
https://martinfowler.com/articles/microservices.html

MICROSERVICE ARCHITECTURE

Team organization



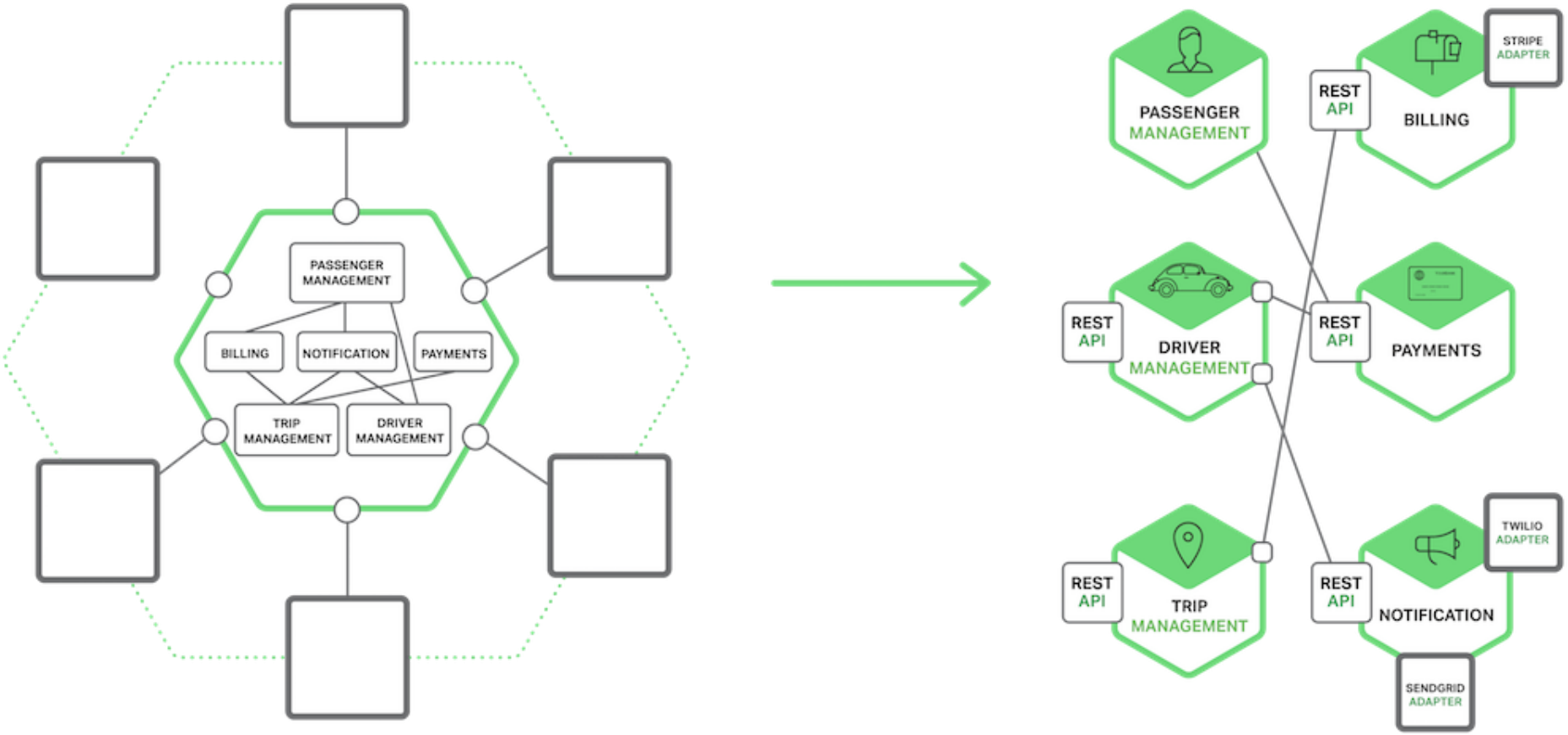
Monolithic (layered)



Microservice

https://martinfowler.com/articles/microservices.html

HOW DO MICROSERVICES COMMUNICATE?



https://www.nginx.com/blog/building-microservices-inter-process-communication/

COMMUNICATION STYLES

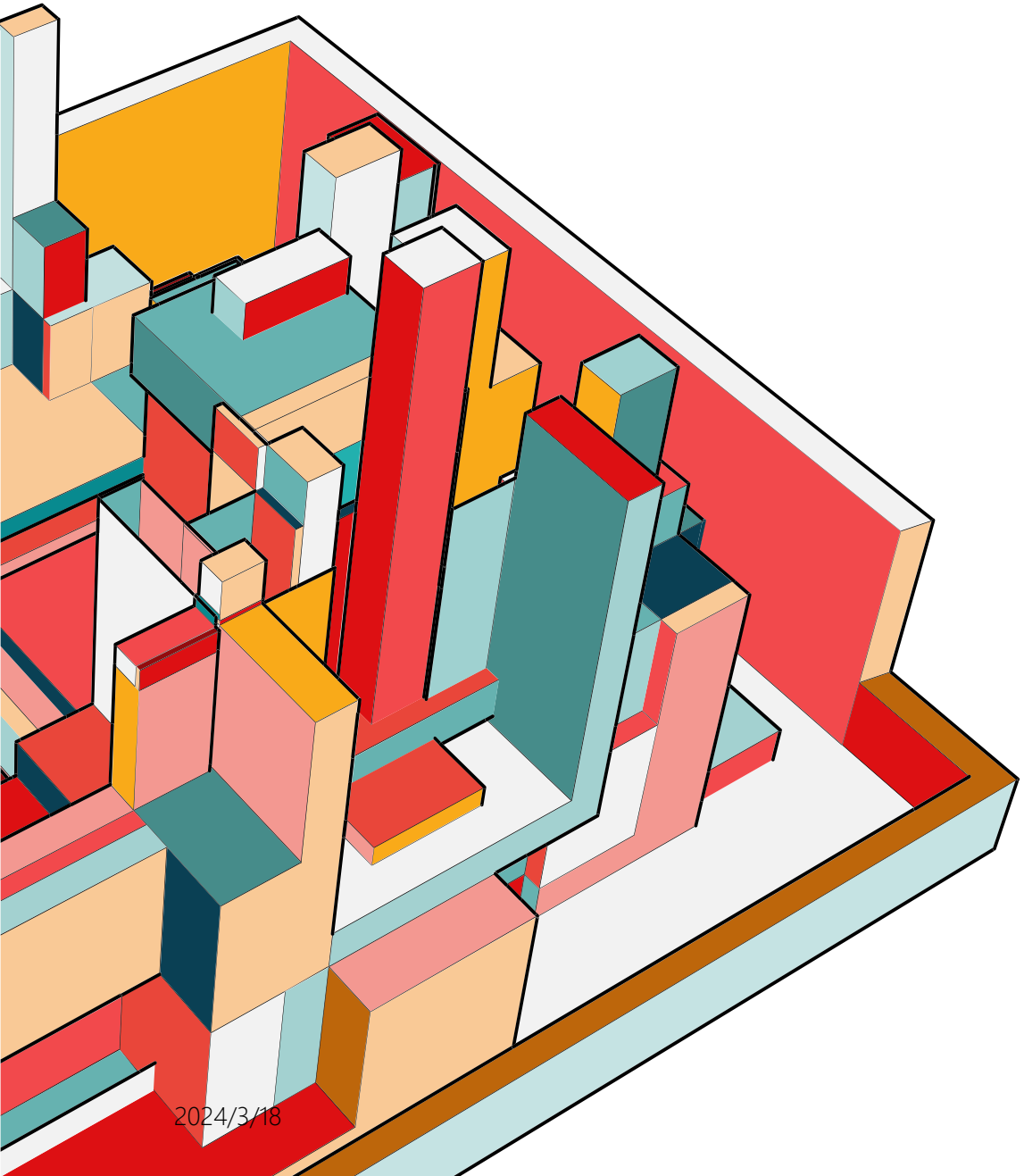
One-to-one or one-to-many

- One-to-one: Each client request is processed by exactly one service.
- One-to-many: Each request is processed by multiple services.

Synchronous or asynchronous

- Synchronous: The client expects a timely response from the service and might even block while it waits.
- Asynchronous: The client doesn’t block, and the response, if any, isn’t necessarily sent immediately.

Microservice Patterns: with Examples in Java. Chris Richardson



COMMUNICATION MECHANISM

Synchronous

- RESTful API
- gRPC

Asynchronous

- Messaging

RESTFUL API

Resource-based API for web servers

https://www.youtube.com/@ByteByteGo

- Client-server: A client-server architecture made up of clients, servers, and resources (info like text, image, video)
- Resources could be accessed using URL
- Stateless: Resource requests should be made independently of one another
- Requests are made using HTTP protocol: GET, POST, PUT, DELETE
- Used by X (Twitter), Youtube, etc.

RESTFUL API

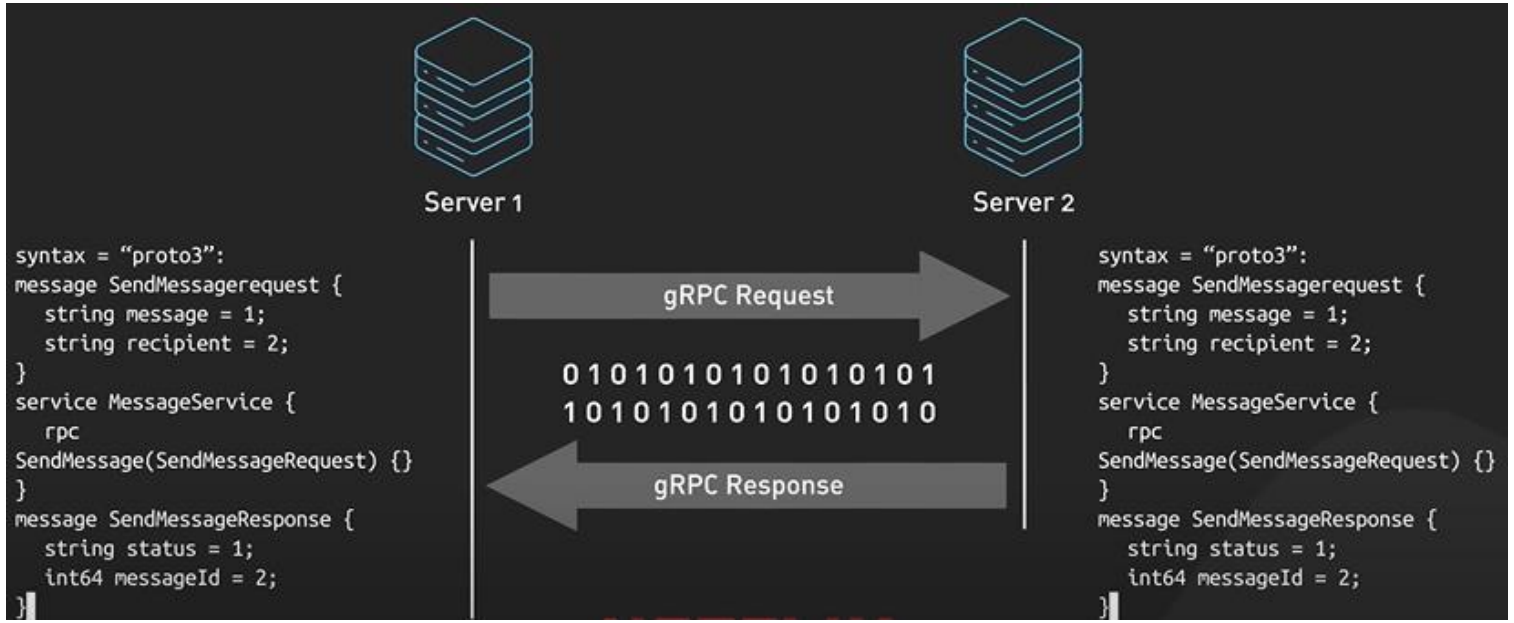
Resource-based API for web servers

https://www.youtube.com/@ByteByteGo

- REST API is best suited for applications with simple data sources where resources are well-defined.
- REST API is not suitable for fetching multiple resources in a single request
- Alternatives
  - GraphQL: Meta (Facebook)
  - Netflix Falcor

GRPC

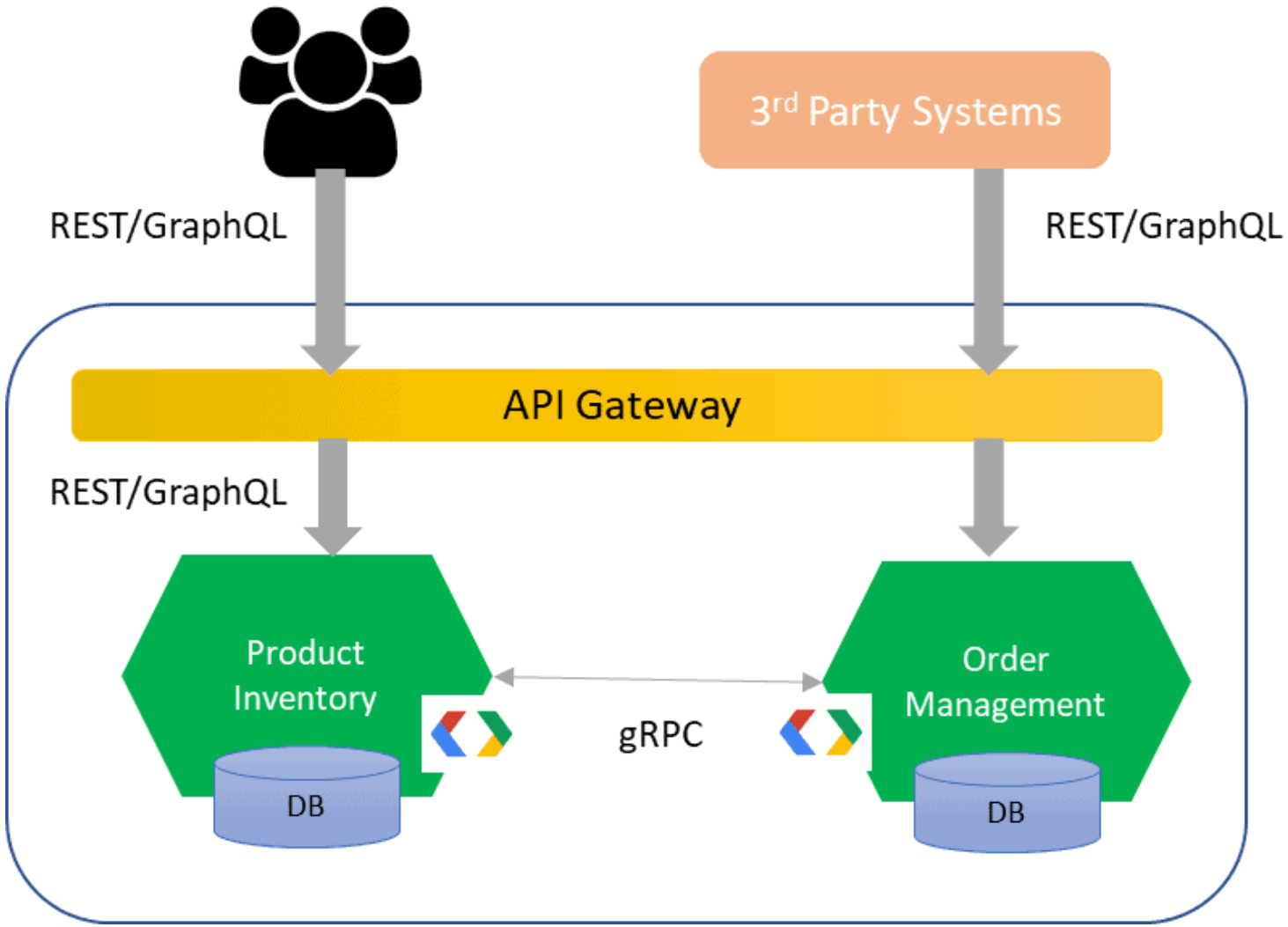
- gRPC is a binary message-based protocol, facilitating efficient communication between distributed systems through Remote Procedure Calls (RPC).
- Support more operations (verbs) than REST
- Best suited for high-performance or data-heavy microservice architectures.
- Used by Netflix, Google, etc.



https://www.youtube.com/@ByteByteGo

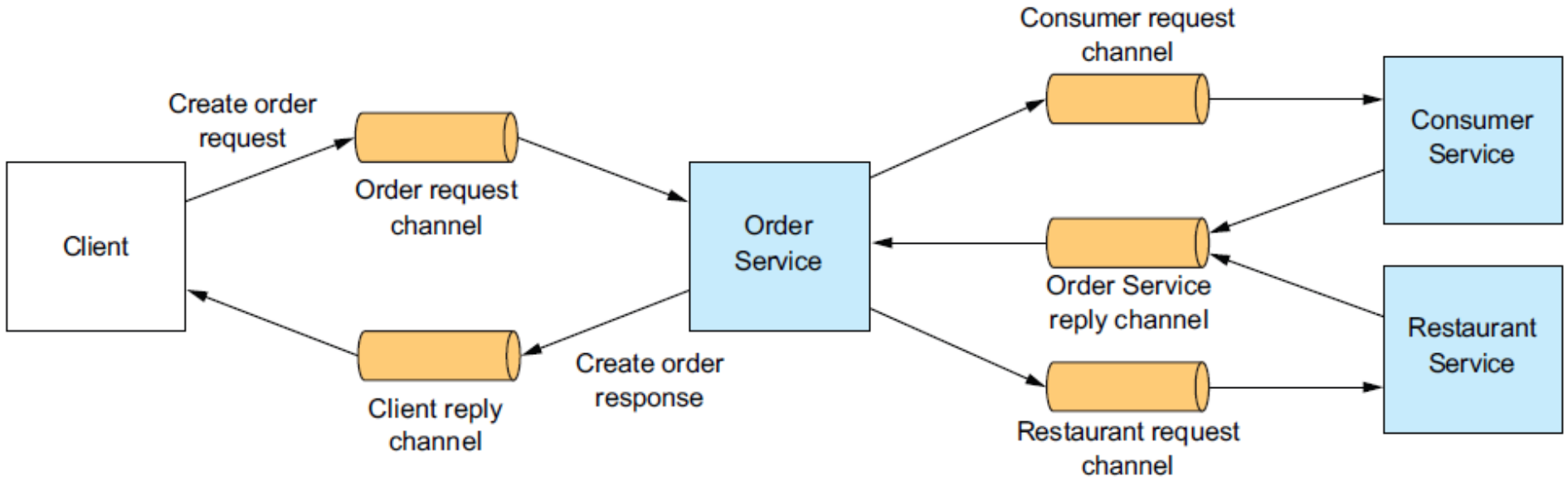
EXAMPLE

- Client-facing service: REST
- Inter-service communication: gRPC



MESSAGING

- In the messaging model, messages are exchanged over message channels.
  - A sender (e.g., a service) writes a message to a channel
  - a receiver reads messages from a channel.
- Open source: RabbitMQ, Apache Kafka



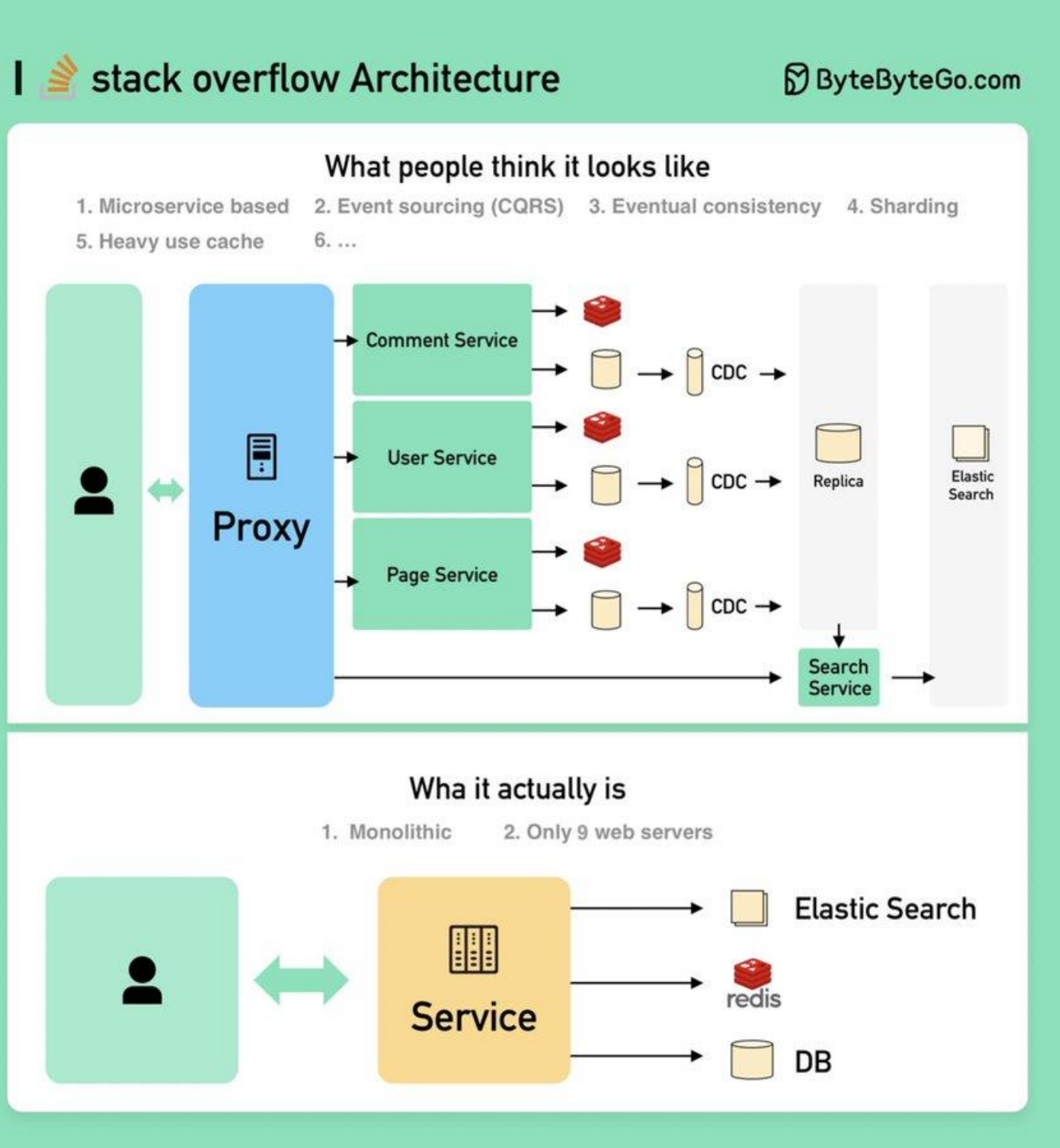
Microservice Patterns: with Examples in Java. Chris Richardson

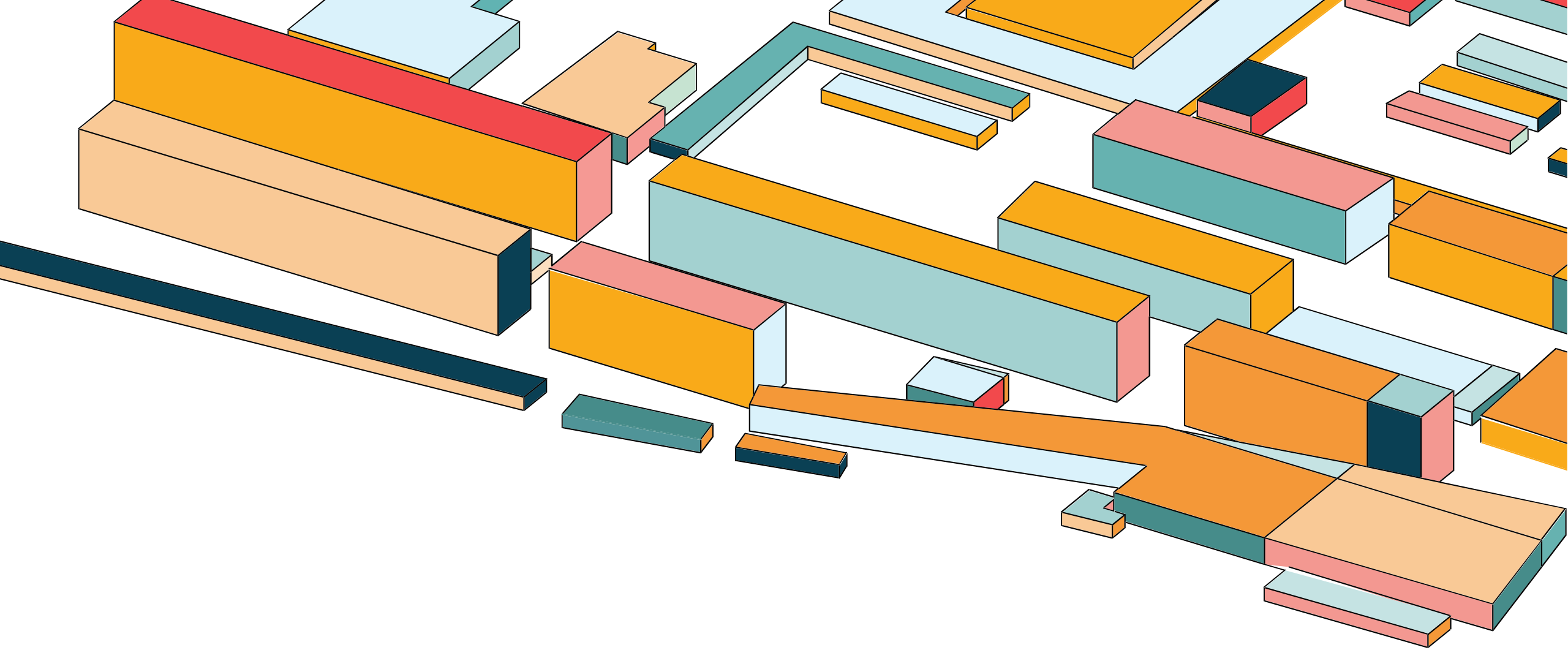
CHOOSING ARCHITECTURAL STYLES

- We’ve introduced only a subset of available architectural styles
- Once requirements engineering uncovers the characteristics and constraints of the system to be built, the architectural style that best fits those characteristics and constraints can be chosen.
- Different architectural styles are NOT mutually exclusive; instead, they are often applied in combination (e.g., a layered style can be combined with a data-centered architecture in many database applications.)

CHOOSING ARCHITECTURES

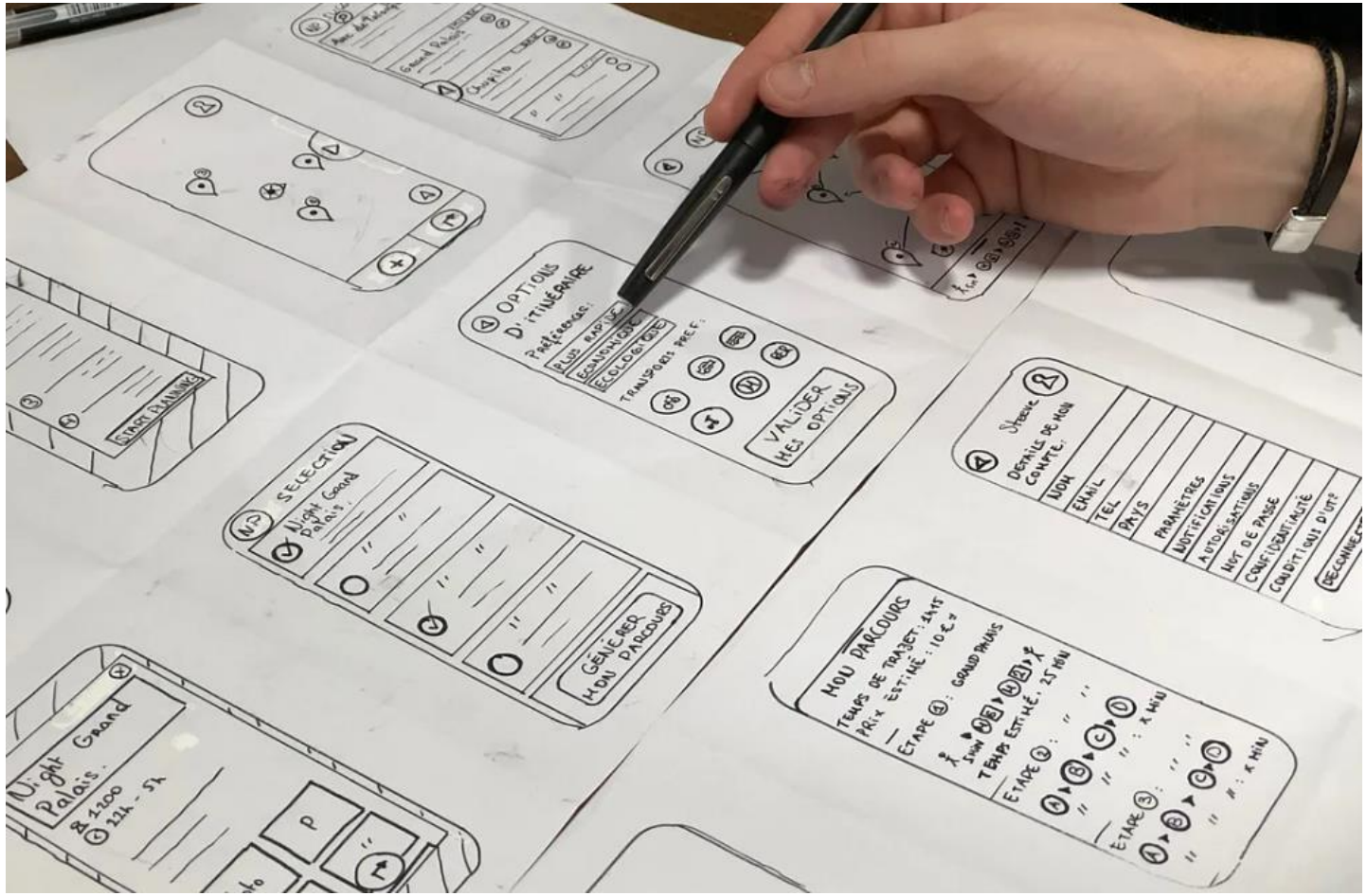
- No good or bad architecture (We are not saying microservice is better than monolithic)
- Choose what's suitable based on your application, resources, user base, business style, team structure, etc.



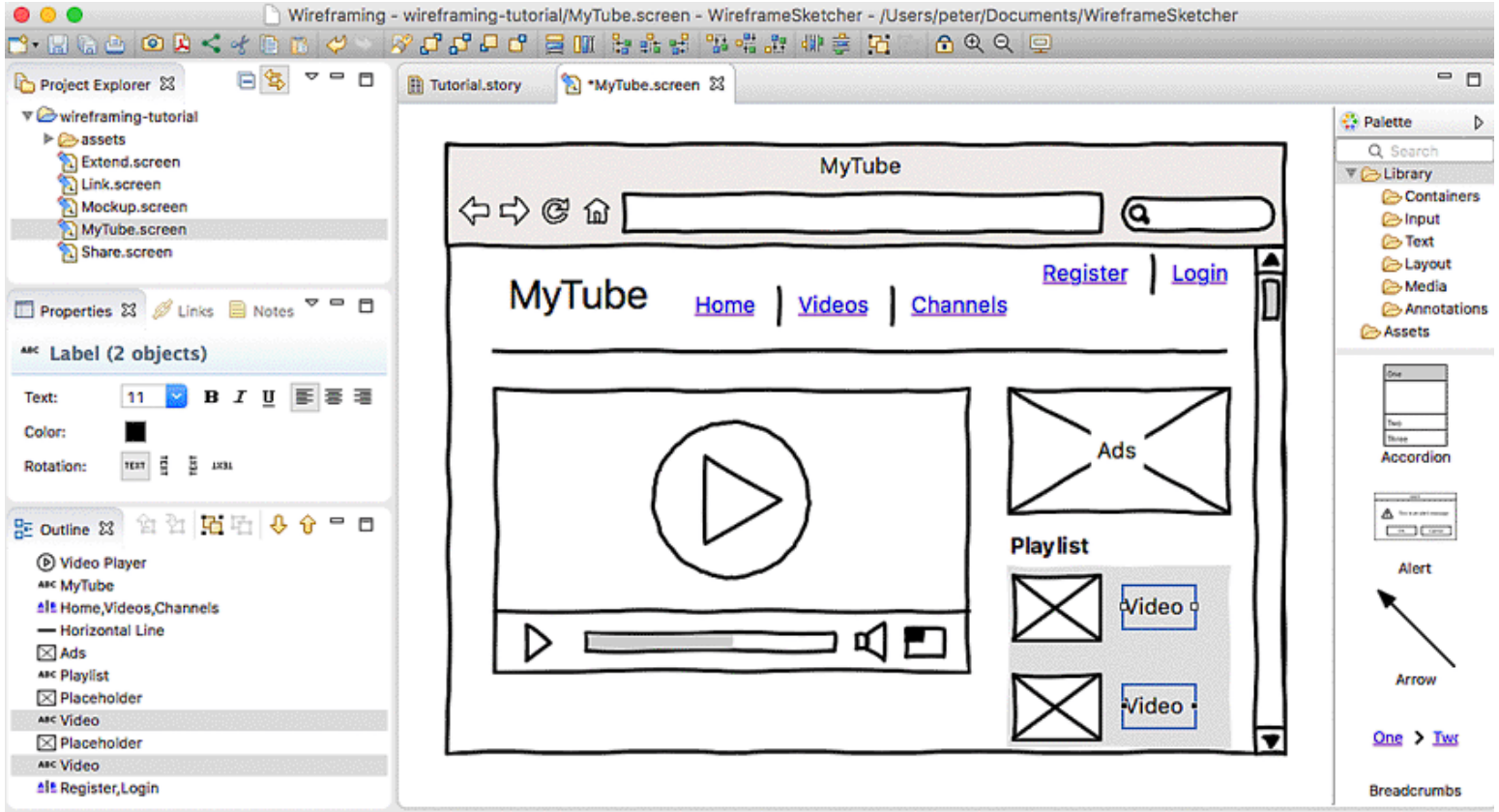


SOFTWARE UI DESIGN

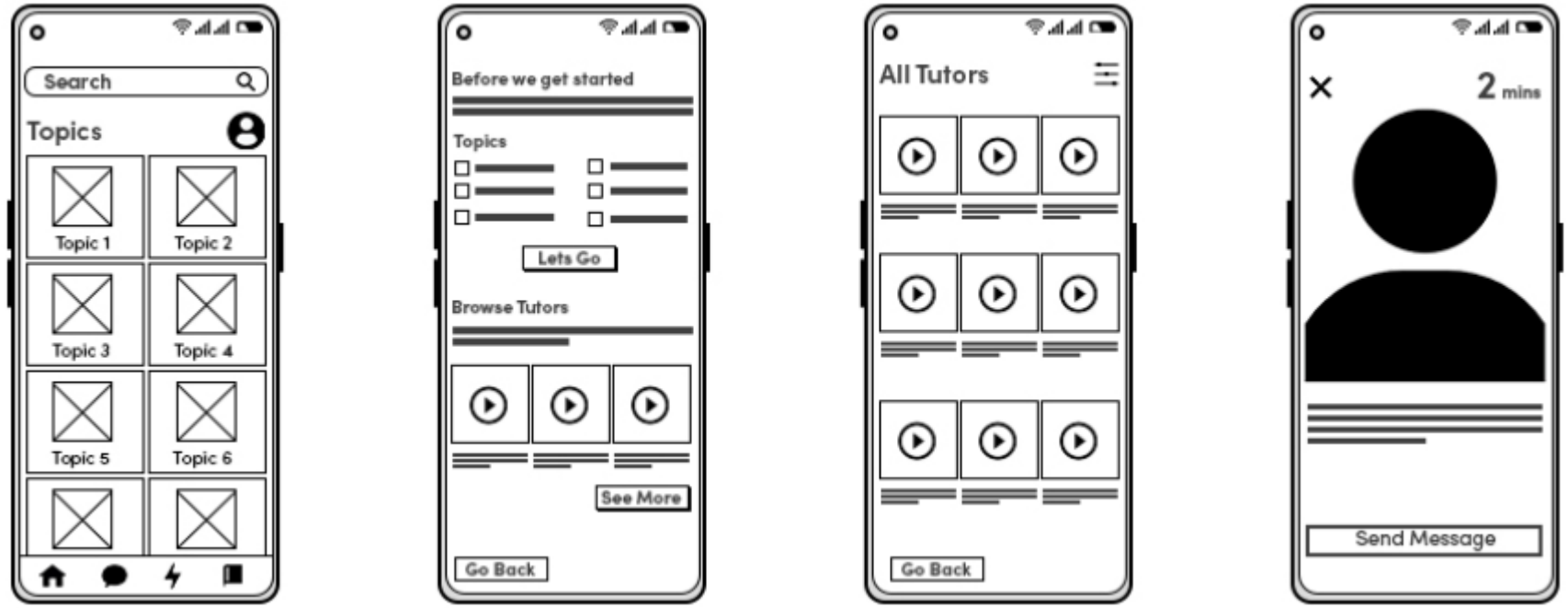
DESIGN BY SKETCH



DESIGN WITH WIREFRAMING TOOLS

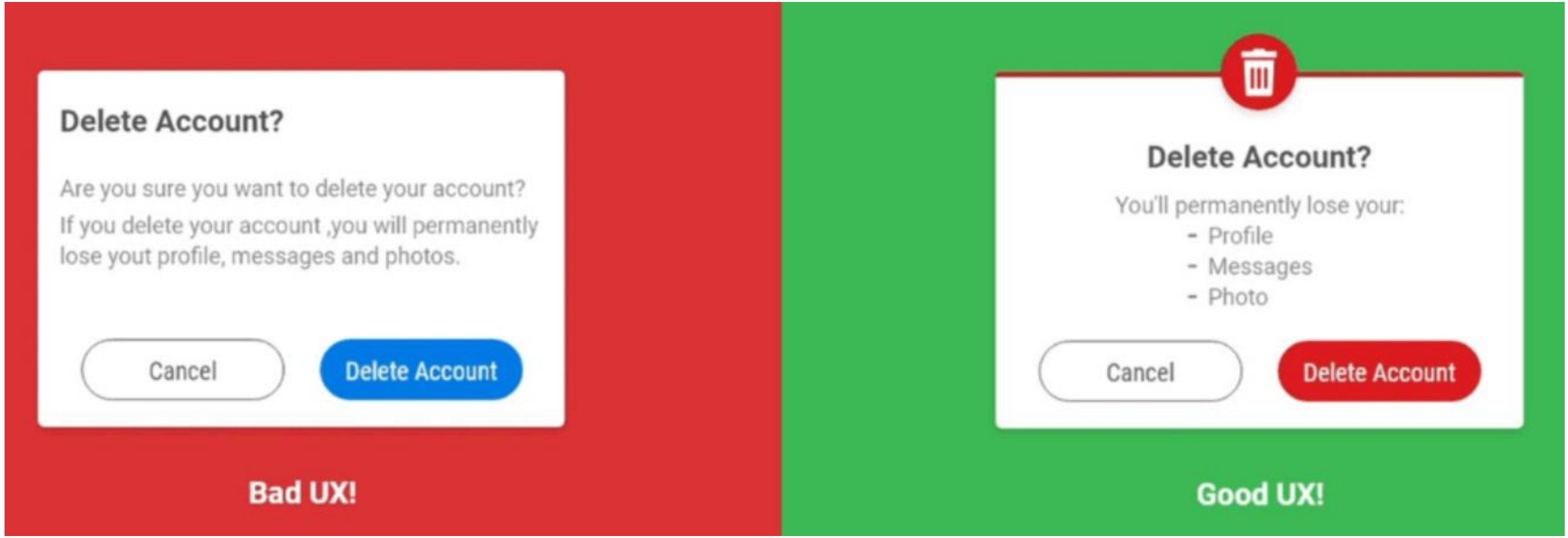


DESIGN WITH WIREFRAMING TOOLS

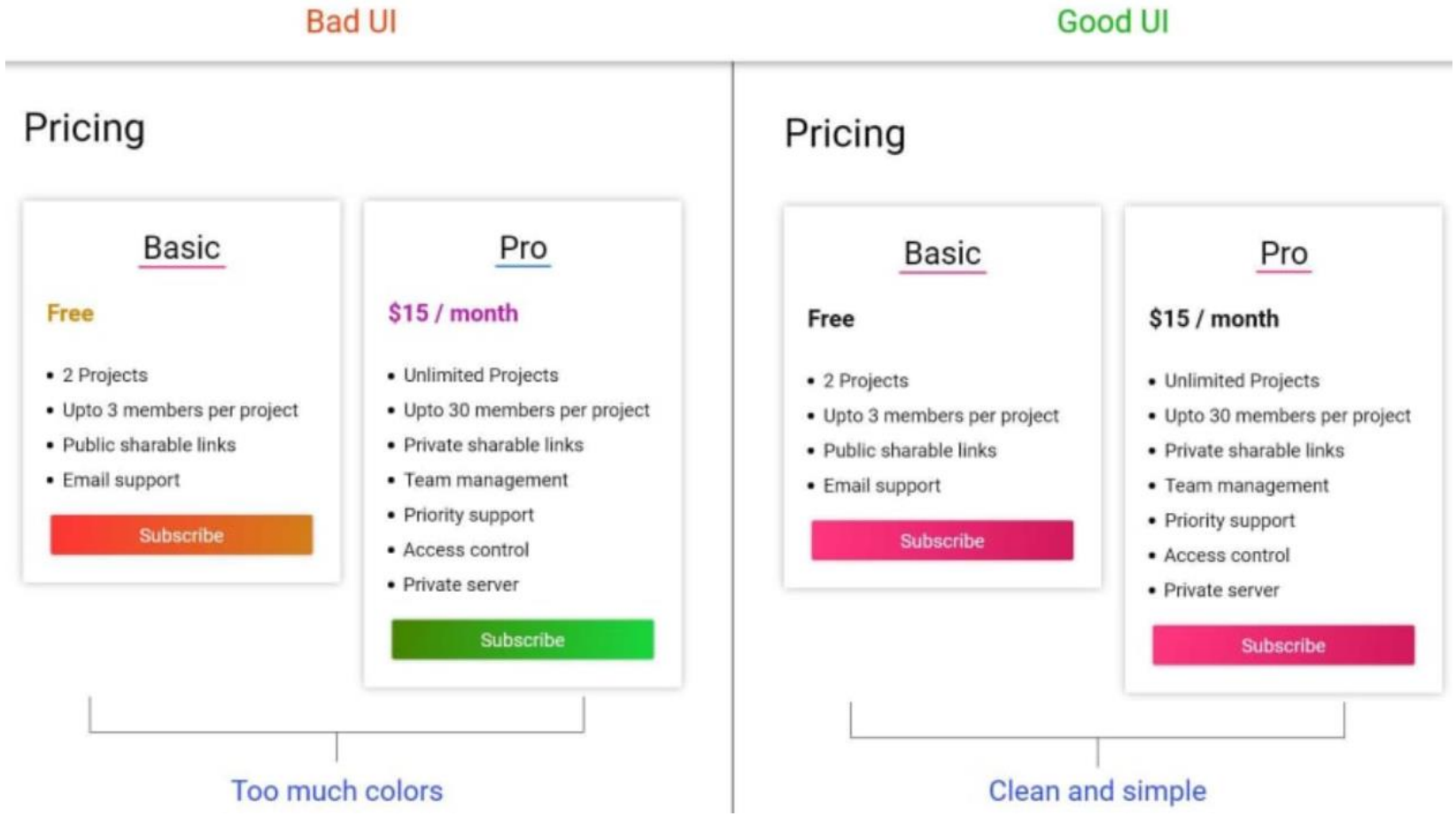


BAD UI DESIGN

BAD UI DESIGN



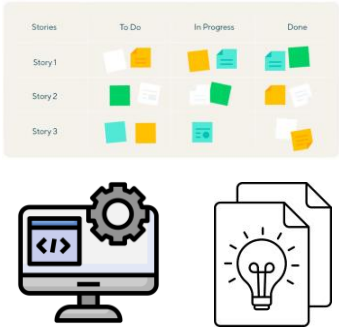
BAD UI DESIGN

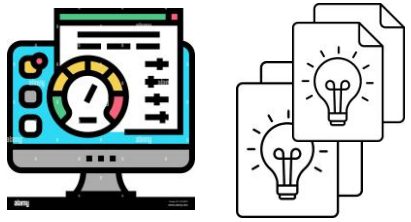


https://dev.to/piyushpawar17/ui-design-for-front-end-developers-2f2o

OUR TEAM PROJECT

1. Architectural design
2. UI design
3. Git collaboration
4. Demo
5. Scrum board for the next sprint





Week 1
Team up

Week 5
Proposal

Week 9
Sprint 1

Week 16
Sprint 2

NEXT

- Build Systems