

Architectural Skills

CSSE6400

Richard Thomas

May 29, 2023

Quote

Architecture is the stuff you can't Google.

– Mark Richards [1]

Quote

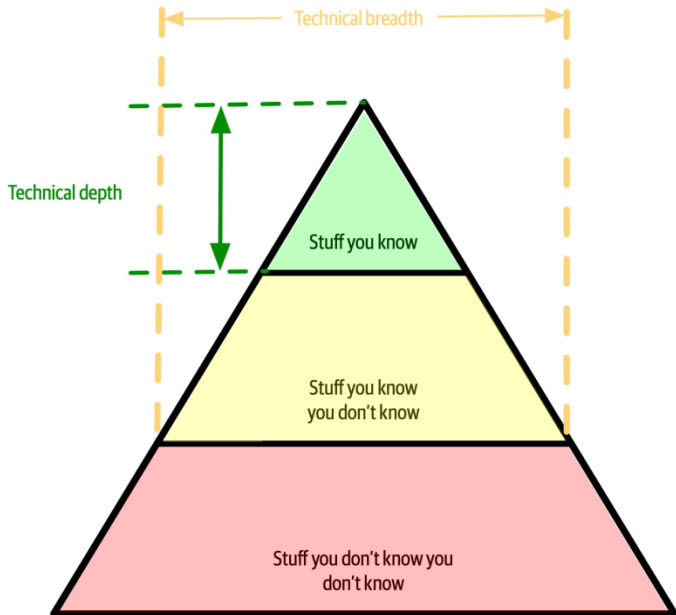
There are no right or wrong answers in architecture—only trade-offs.

– Neal Ford [\[1\]](#)

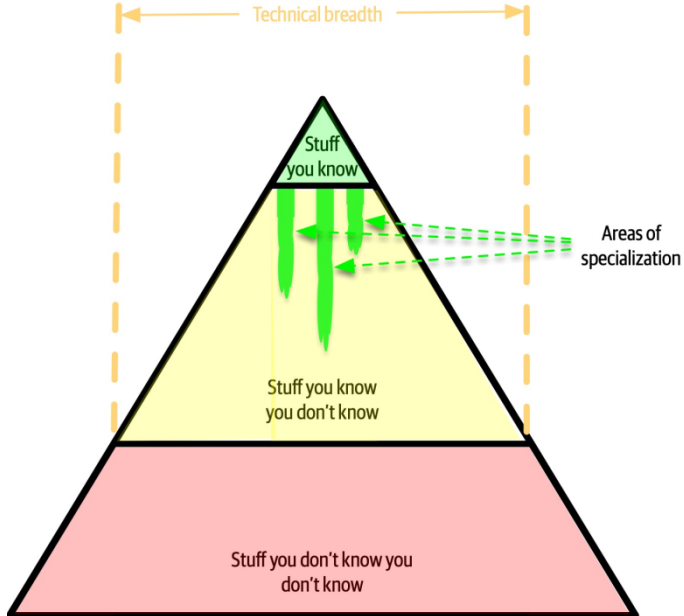
Architectural Design

Architects use knowledge and experience to analyse trade-offs to design architectures appropriate to the system context.

Developers – Technical Depth [1]



Architects – Technical Breadth [1]



Question

What are the benefits of a monolith architecture?

Question

What are the benefits of a monolith architecture?

Answer

- Simple deployment

Question

What are the benefits of a monolith architecture?

Answer

- Simple deployment
- Simple communication between modules

Question

What are the benefits of a monolith architecture?

Answer

- Simple deployment
- Simple communication between modules
- Simple system testing & debugging

Question

Why do monoliths have a bad name?

Question

Why do monoliths have a bad name?

Answer

- Many legacy system nightmares were monoliths

Question

Why do monoliths have a bad name?

Answer

- Many legacy system nightmares were monoliths
- Easy to defeat modularity

Question

Why do monoliths have a bad name?

Answer

- Many legacy system nightmares were monoliths
- Easy to defeat modularity
- Cannot scale components of system

Question

Why do monoliths have a bad name?

Answer

- Many legacy system nightmares were monoliths
- Easy to defeat modularity
- Cannot scale components of system
- Monolith databases scale poorly

Question

What can be done if a monolith architecture is no longer suitable?

Question

What can be done if a monolith architecture is no longer suitable?

Answer

- Greenfields replacement

Question

What can be done if a monolith architecture is no longer suitable?

Answer

- Greenfields replacement
- Migrate to another architecture

Question

How do I migrate a monolith to a new architecture?

Question

How do I migrate a monolith to a new architecture?

Answer

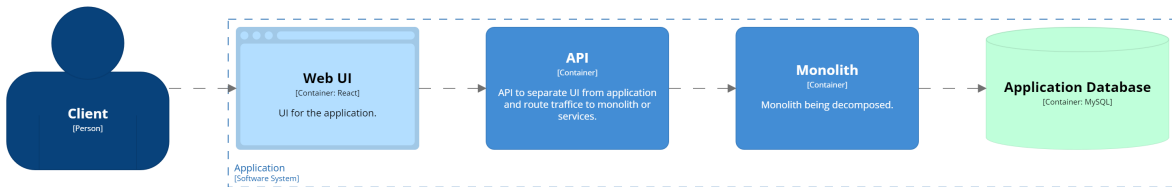
Decompose the monolith into services.

Strangler Fig Pattern

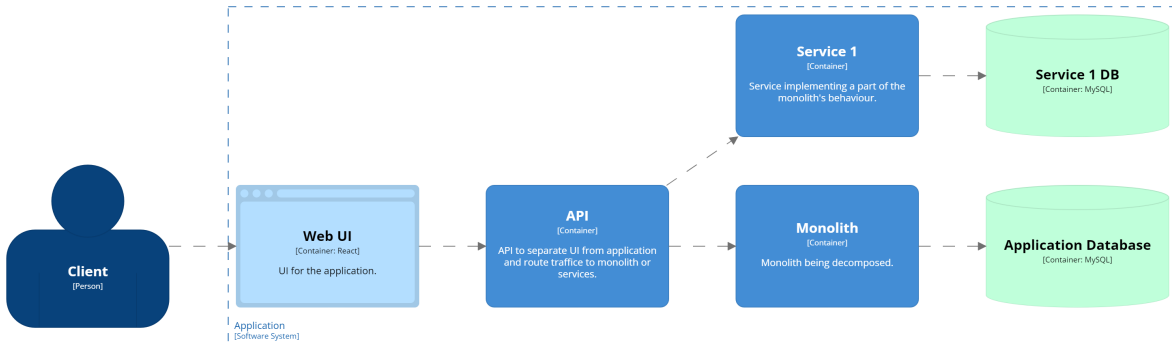
- Develop API for application's UI
- Proxy intercepts API calls
 - Proxy directs calls to application or new services
- Implement a service
 - Redirect calls to service
- Progressively replace monolith
- Shadow & Blue-Green Deployment



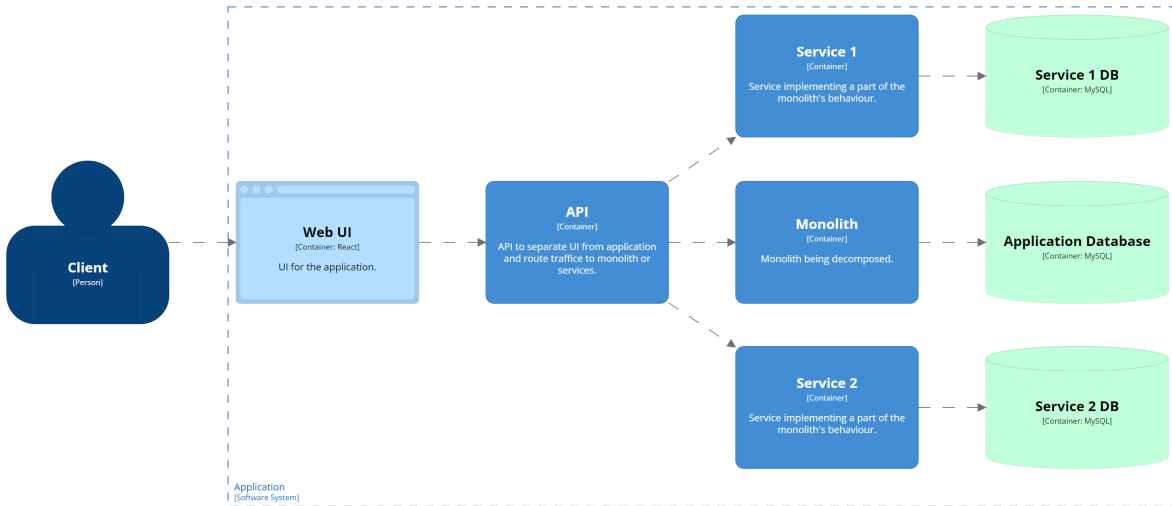
Monolith Deployment



Monolith Decompose: Step 1



Monolith Decompose: Step 2



Decomposition Process

- Identify bounded-contexts
- Simple first service
 - e.g. Authentication
- Minimise dependency from services to monolith
 - Monolith may use services

Decomposition Process

- Reduce coupling between bounded-contexts
 - e.g. Customer account management
 - Profile, Wish List, Payment Preferences – separate services
- Decouple vertically
 - Service delivers entire bounded-context
 - Data is decoupled from monolith

Decomposition Process

- Focus on pain points
 - Bottlenecks
 - Frequently changing behaviour
- Rewrite, don't reuse
 - Redesign for new infrastructure
 - Reuse complex logic
 - e.g. Discounts based on customer loyalty and behaviour, bundle offers, ...

Atomic Decomposition

- Refactor monolith
 - Use service to deliver application functionality
 - Monolith may need to invoke service
 - Remove service logic from monolith

Stepwise Decomposition

Replace application functionality one service at a time.

Definition 1. *Microservice*

Separate service, but may span more than one domain or share a database with the monolith or other services.

Definition 2. Nanoservice

Service that depends on other services and cannot be deployed independently – its context is too small.

Definition 3. Conway's Law

Organisations design systems whose structure is inevitably a copy of the organisation's communication structure [2] [3].

Conway's Law Consequences

- Business Process Management
- Microservices to reflect organisation structure
- Teams formed around services

Conway's Law Consequences

Team insularity – more loyal to team than organisation.

Conway's Law Issues

- Cross-cutting concerns
 - e.g. Security
- Organisation structure should align with market structure
- Physical location of teams

Evidenced-Based Software Engineering

Don't follow fads, seek evidence for good practice.

Let's hear from an expert

Software Engineering's Greatest Hits

**what we actually know about software development
and why we believe it's true**



Greg Wilson

<http://third-bit.com/talks/greatest-hits/>



1 / 47

References

- [1] Mark Richards and Neal Ford.
Fundamentals of Software Architecture: An Engineering Approach.
O'Reilly Media, Inc., January 2020.
- [2] Melvin E. Conway.
How do committees invent?
Datamation, April 1968.
- [3] Alan MacCormack, Carliss Baldwin, and John Rusnak.
Exploring the duality between product and organizational architectures: A test of the “mirroring” hypothesis.
Research Policy, 41(8):1309–1324, 2012.