Decomposing Monoliths CSSE6400

Richard Thomas

May 19, 2025



What are the benefits of a monolith architecture?

What are the benefits of a monolith architecture?

Answer

• Simple deployment

What are the benefits of a monolith architecture?

- Simple deployment
- Simple communication between modules

What are the benefits of a monolith architecture?

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

Why do monoliths have a bad name?

Why do monoliths have a bad name?

Answer

• Many legacy system nightmares were monoliths

Why do monoliths have a bad name?

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

Why do monoliths have a bad name?

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

Why do monoliths have a bad name?

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

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

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

Answer

• Greenfield replacement

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

- Greenfield replacement
- Migrate to another architecture

How do I migrate a monolith to a new architecture?

How do I migrate a monolith to a new architecture?

Answer

Decompose the monolith into services.



• Develop API for application's UI



- Develop API for application's UI
- Proxy intercepts API calls
 - Proxy directs calls to application or new services



- 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



- 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



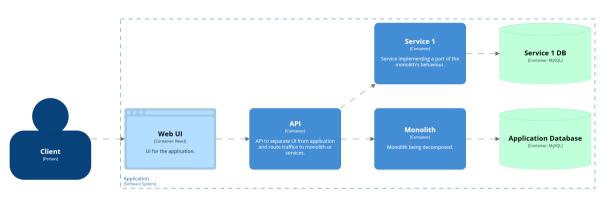
- Develop API for application's UI
- Proxy intercepts API calls
 - Proxy directs calls to application or new services
- Implement a service
 - Redirect calls to service
- Progressively replace monolith
- Shadow & Blue-Green Deployment



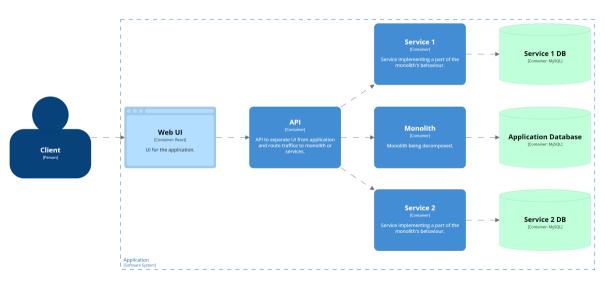
Monolith Deployment



Monolith Decompose: Step 1



Monolith Decompose: Step 2



Identify bounded-contexts

- Identify bounded-contexts
- Simple first service
 - e.g. Authentication

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

- Reduce coupling between bounded-contexts
 - e.g. Customer account management
 - Profile, Wish List, Payment Preferences
 - Separate Services

- 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



- Focus on pain points
 - Bottlenecks
 - Frequently changing behaviour

- 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 0. Macroservice

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

Definition 0. Nanoservice

deployed independently – its context is too small.

Service that depends on other services and cannot be

References

[Fowler, 2024] Fowler, M. (2024).

Strangler fig.

https://martinfowler.com/bliki/StranglerFigApplication.html.