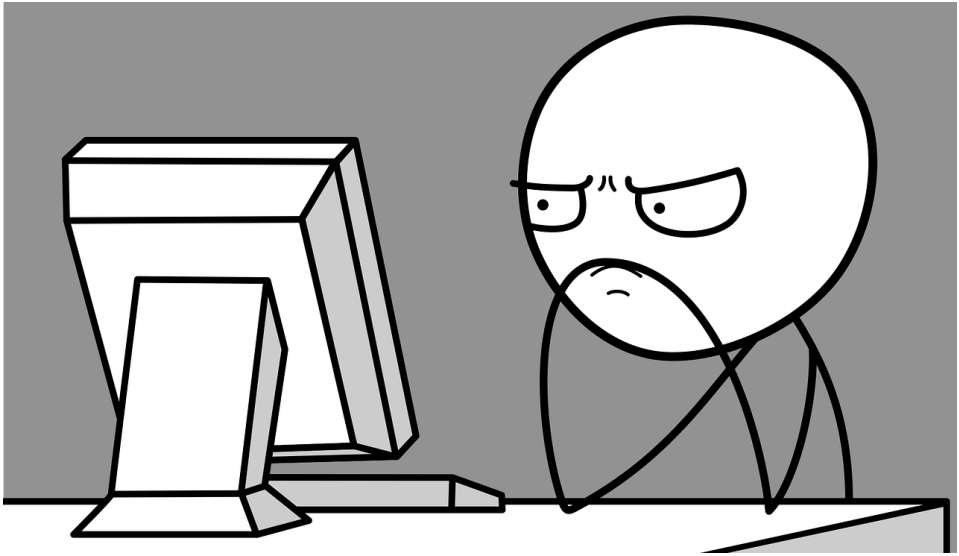
Architectural Decision Records

Software Architecture

Richard Thomas & Guangdong Bai

March 3, 2025

Developer Reaction to Reading Software Architecture Documentation



Question

How do you know why certain decisions were made in the architectural design?

Question

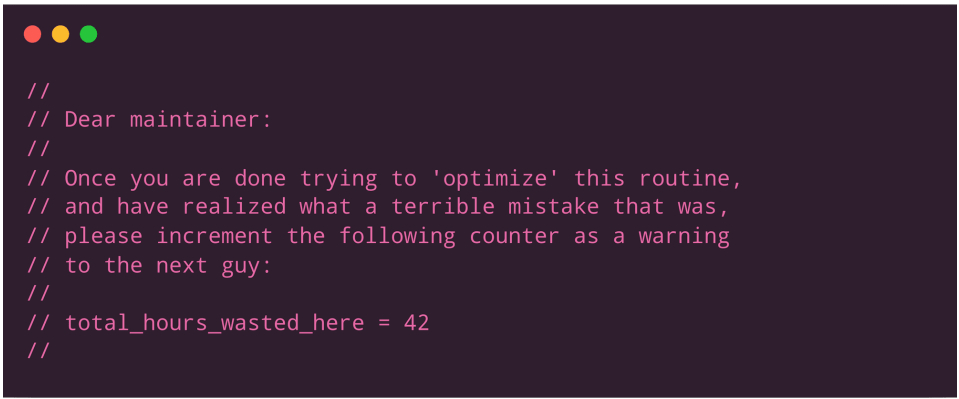
How do you know why certain decisions were made in the architectural design?

Answer

Architectural Decision Records (ADRs)

Record Decisions that Influence

- *Structure* of the architecture
- Delivery of *quality attributes*
- *Dependencies* between important parts of the architecture
- *Interfaces* between important parts of the architecture
  - or external interfaces
- *Principles* about implementation techniques or platforms
- More ”why certain decisions were made” than ”what are in structure, interfaces, etc.”


//
// Dear maintainer:
//
// Once you are done trying to 'optimize' this routine,
// and have realized what a terrible mistake that was,
// please increment the following counter as a warning
// to the next guy:
//
// total_hours_wasted_here = 42
//

Question

Why ADRs?

Question

Why ADRs?

Answer

My code will defeat the architectural design,
if I do not know why it was designed that way.

Why ADRs

- *Team* alignment to ensure all stakeholders are on the same page regarding architectural choice
  - Onboarding team members sync up on past decisions.
- *Traceability* to allow the team to track how architecture *evolves*
- Decision *justification* to defend decisions when questioned later
- Preserve *knowledge* for future reference

ADRs

Record a *single* decision

ADRs

Are *never* deleted

Mark as *superseded* and link to new decision

Question

Where are ADRs documented?

Question

Where are ADRs documented?

Answer

Each decision is a separate file in the project repository[1]

[1] See the `adrs` directory in the C4 Model on the course website.

ADR Management Tools

- Log4brains
  - Generates ADRs from Markdown files.
  - Provides a web-based UI for browsing ADRs.
- adr-tools
  - A light-weight command-line tool.
  - Generates ADR templates automatically.

ADR Template [Nygard, 2011]

Title Short phrase describing the decision

Date When the decision was made

Status Current status of the decision

- proposed, accepted, deprecated, superseded, rejected

Summary Summarise the decision and its rationale

Context Describe the facts that influence the decision

Decision Explain how the decision will solve the problem

Consequences Impact of the decision

- what's easier to do
- what's harder to do

ADR Example

1. Independent Business Logic

Date: 2022-01-06

Status

Accepted

Summary

In the context of delivering an application with multiple platform interfaces,
facing budget constraints on development costs,
we decided to implement all business logic in an independent tier of the software architecture,
to achieve consistent logical behaviour across platforms,
accepting potential complexity of interfaces to different platforms.

Context

- The system is to have both mobile and web application frontends.
- Marketing department wants a similar user experience across platforms.
- Delivering functional requirements requires complex processing and database transactions.
  - Product recommendations based on both a customer's history and on purchasing behaviour of similar customers.
  - Recording all customer interactions in the application.
- Sales department wants customers to be able to change between using mobile and web applications without interrupting their sales experience.
- Development team has experience using Java.

Decision

All business logic will be implemented in its own tier of the software architecture. Web and mobile applications will implement the interaction tier. They will communicate with the backend to perform all logic processing. This provides clear separation of concerns and ensures consistency of business logic across frontend applications. It means the business logic only needs to be implemented once. This follows good design practices and common user interface design patterns.

The business logic will be implemented in Java. This suits the current development team's experience and is a common environment. Java has good performance characteristices. Java has good support for interacting with databases, to deliver the data storage and transaction processing requirements.

Consequences

Advantages

- Separation of concerns, keeping application logic and interface logic separate.
- Ensures consistency, if business logic is only implemented in one place.
- Business logic can execute in a computing environment optimised for processing and transactions.
  - Also makes load balancing easier to implement.

Neutral

Reading...

"Architectural Decision Records" Notes *[Thomas, 2023]*

References

[Nygard, 2011] Nygard, M. (2011).
Documenting architecture decisions.
https://cognitect.com/blog/2011/11/15/documenting-architecture-decisions.
Accessed: 2022-01-27.

[Thomas, 2023] Thomas, R. (2023).
Architectural decision records.
https://csse6400.uqcloud.net/handouts/adr.pdf.