


# ARCHITECTURE DECISION RECORDS

REMEMBER THE WHY



# The Problem Case



# The Problem Case

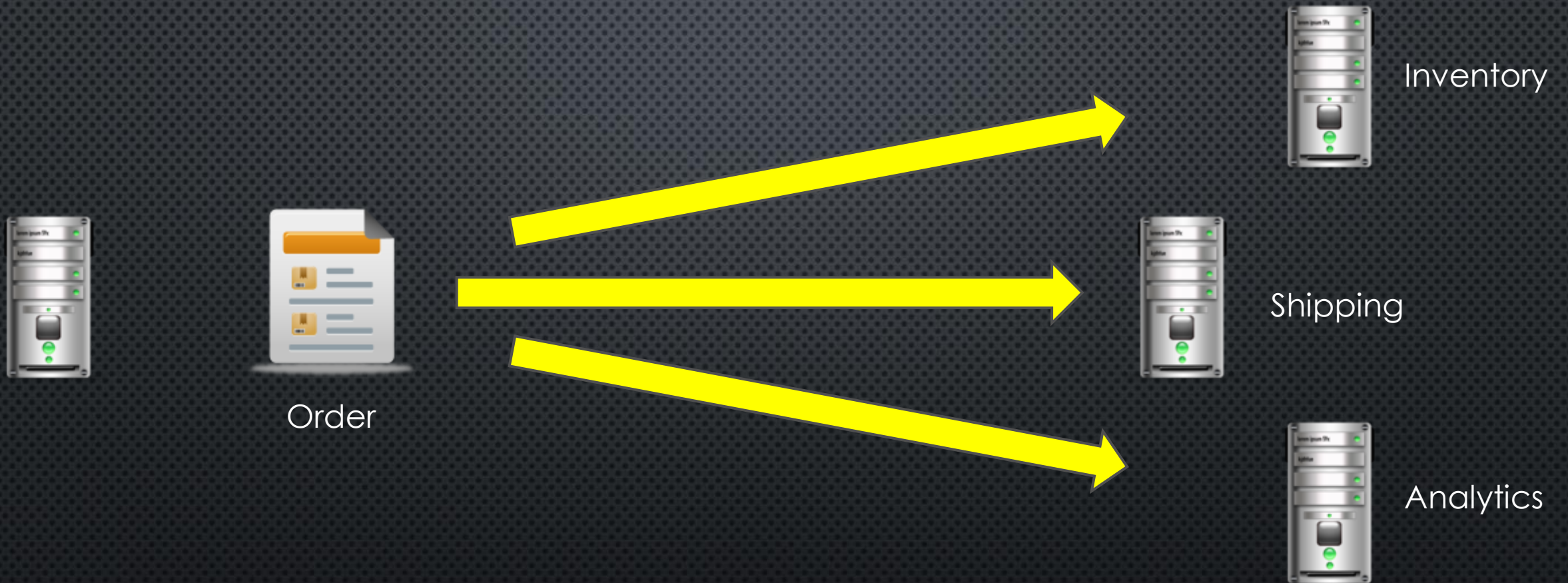


Order



Shipping

# The Problem Case





# The Problem Case



# The Problem Case

Capture changes to architecture

User Stories – new or modified ?

Email ?

Notes ?



<https://unsplash.com/photos/5QgluuBxKwM>



# Architecture Decision Records





# What are Architecture Decision Records?





# What are Architecture Decision Records?

## Documentation

# What are Architecture Decision Records?

Documentation



the “Documentation” word



# What are Architecture Decision Records?

Deploy to production -> Testing -> Writing Unit Tests -> Redeploy to Production -> More testing

Writing of Integration Tests -> Redeploy to Production -> Update Staging of forgotten pieces

Work on next version -> ..... -> ~~Documentation~~ -> Move on to next application



# What are Architecture Decision Records?

Documentation of what changed  
and why



# What are Architecture Decision Records?

Documentation of what changed  
and why

“ADRs” should be simple and easy to use

“ADRs” should be immutable (almost)



What are Architecture Decision Records?

Enough Already!  
Show me an example!



# What are Architecture Decision Records?

## Terminology

Architecture Decision

Architecture Decision Record – record of the decision

Architecture Decision Log – collection of ADRs

# Examples of Changes

Change Identity Provider

ZeroMQ to RabbitMQ

RPC to Messaging

Introduction of a Microservice

Relational DB to Non-Relation DB

Event-Based Trigger to Timer Trigger



# Examples of Changes

Change Identity Provider

Introduction of a Microservice


ZeroMQ to RabbitMQ

Relational DB to Non-Relation DB

RPC to Messaging

Event-Based Trigger to Timer Trigger

Adding another room to house almost completed



# Examples of Changes

to something





# Why use Architecture Decision Records?

# Why use Architecture Decision Records?



<https://unsplash.com/photos/4-EeTnaC1S4>

New to job / team ?

Person with institutional knowledge  
left the company?

No one else to ask?



# Why use Architecture Decision Records?

What was I thinking??

Loss of Context

Change of Context



<https://unsplash.com/photos/1K9T5YiZ2WU>

# Why use Architecture Decision Records?

## New Information

Are the parameters of the decision the same today?



<https://unsplash.com/photos/31OdWLEQ-78>



# Why use Architecture Decision Records?


## Impact ?

Number of technical debt issues [should] decrease

Architecture design quality/competence increase


Quality of communication increase

Frequency of review



# What to capture





# What to capture

## 1. Design Decision

Design Decision:

The decision was made to change from RPC calls to Messaging

# What to capture

1. Design Decision
2. Context

Design Decision:

The decision was made to change from RPC calls to Messaging

Context:

Accounts Receivable module when processing overdue invoices.



# What to capture

1. Design Decision
2. Context
3. Reasoning

Design Decision:

The decision was made to change from RPC calls to Messaging

Context:

Accounts Receivable module when processing overdue invoices.

Reasoning:

Messaging provides ability to have looser coupling between modules and allow for other modules to receive the messages and react independently.

# What to capture

1. Design Decision
2. Context
3. Reasoning
4. Implications

Context:

Accounts Receivable module when processing overdue invoices.

Reasoning:

Messaging provides ability to have looser coupling between modules and allow for other modules to receive the messages and react independently.

Implications:

Messaging system will need to be evaluated and managed upon implementation.



# What to capture

1. Design Decision
2. Context
3. Reasoning

4. Implications
5. Consequences

Reasoning:

Messaging provides ability to have looser coupling between modules and allow for other modules to receive the messages and react independently.

Implications:

Messaging system will need to be evaluated and managed upon implementation.

Consequences:

Code changes to work with defined message format. Code to handle dead-letter queue.

# What to capture

1. Design Decision
2. Context
3. Reasoning
4. Implications
5. Consequences
6. Options not Chosen/Why

Implications:

Messaging system will need to be evaluated and managed upon implementation.

Consequences:

Code changes to work with defined message format. Code to handle dead-letter queue.

Options not Chosen & Why:

RPC incurred too much latency. Had too much coupling between modules and prevented easy adoption of other modules to react to events.



# What to capture

## 7. Tags (optional)

Tags:

RPC; Messaging; Accounts Receivable; Invoice; Overdue; Over due

Include

Sprint number ?

Something else?

Spelling variations?

# Language



Design Decision:

The decision was made

to change from RPC calls to messaging

<https://unsplash.com/photos/POMpXtcVYHo>



# Language



<https://unsplash.com/photos/POMpXtcVYHo>

Design Decision:

The decision was made ...

The product chosen was ...

We will ...

(not the only ways)



# Tools & Templates



# Tools & Templates

Michael Nygard

<https://cognitect.com/blog/2011/11/15/documenting-architecture-decisions>

Title

Context

Decision

Status

Consequences

Proposed, Accepted, Rejected, Deprecated or Superseded



# Tools & Templates

Design Decision

Context

Rationale

Reasoning

Implications

Positive and Negative





# Tools & Templates

1. Design Decision
2. Context
3. Reasoning
4. Implications
5. Consequences
6. Options not Chosen/Why
7. Tags

# Tools & Templates

<https://github.com/joelparkerhenderson/architecture-decision-record>

<https://github.com/npryce/adr-tools>

<https://cognitect.com/blog/2011/11/15/documenting-architecture-decisions>

<https://ardalis.com/getting-started-with-architecture-decision-records/>







# Where to Store ADRs

Where to store the ADRs ?

# Where to Store ADRs

Where to store the ADRs ?

Project or Solution Level?



# Where to Store ADRs

Where to store the ADRs ?

Project or Solution Level?

Solution

- Project 1

Solution

- Project 1
- Project 2
- Project ...
- Project 24



# Where to Store ADRs

Where to store the ADRs ?

What about with microservices?



# Where to Store ADRs

Where to store the ADRs ?

What about with microservices?

Application 1  
Solution  
- Project 1

Application 3  
Solution  
- Project 1

Application 4  
Solution  
- Project 1

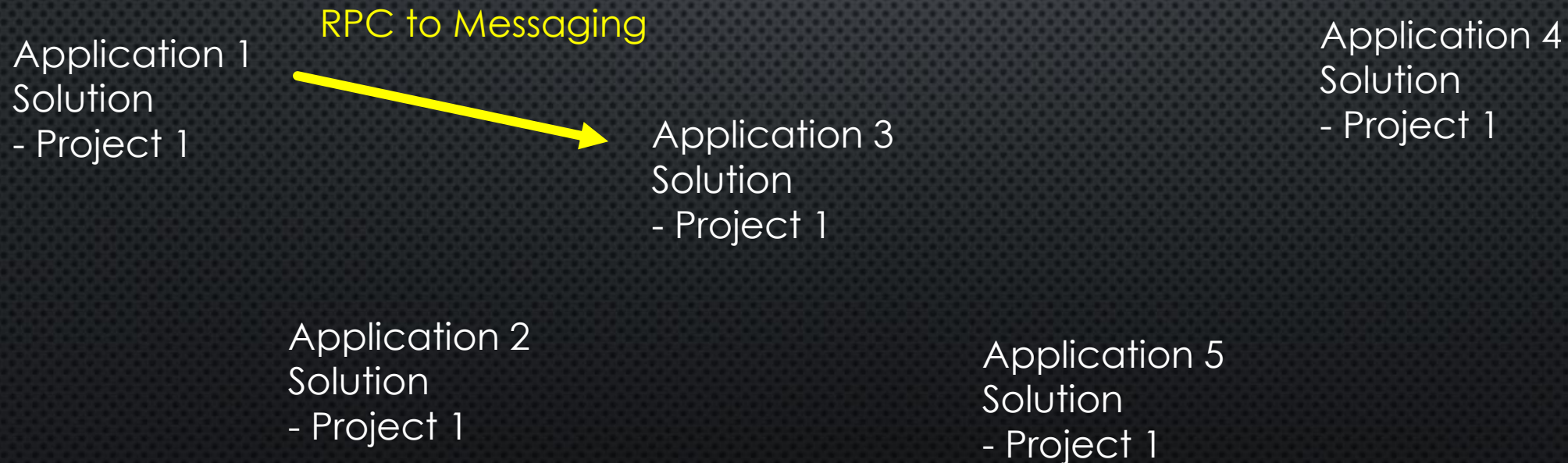
Application 2  
Solution  
- Project 1

Application 5  
Solution  
- Project 1

# Where to Store ADRs

Where to store the ADRs ?

What about with microservices?





# Where to Store ADRs

Where to store the ADRs ?

What about with microservices?



# Review the ADRs

When time to review ADRs in proposed state:

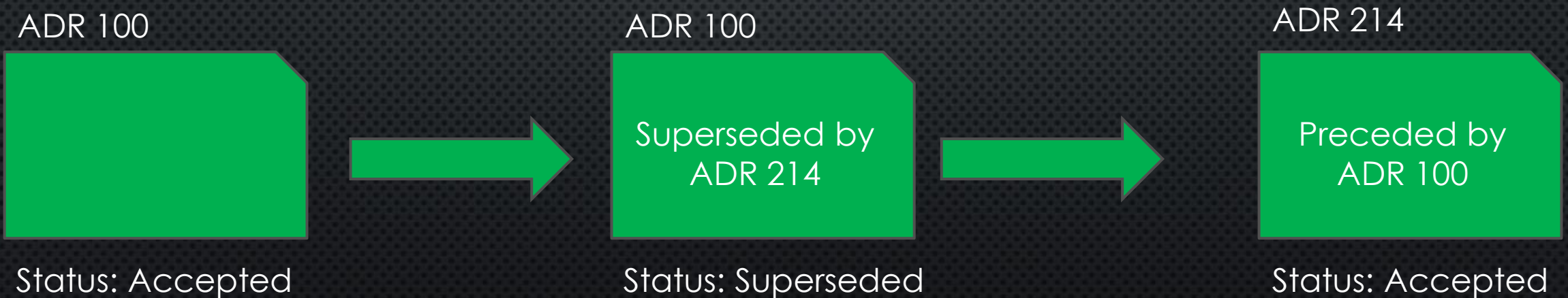
1. Visit often until resolved
2. Remember to keep ADR immutable (except to add reference to superseding ADR or status change)



# Review the ADRs

When time to review ADRs in proposed state:

1. Visit often until resolved
2. Remember to keep ADR immutable (except to add reference to superseding ADR or status change)





# Guidance

Tweak to make work for your team

Just enough information to capture why

Make them searchable (review, amending, history)

Tool should be easy to adopt and use

Peer review – someone from another team – does it make sense to them?





# Guidance

Consider reviewing ADR's touched at sprint review

Consider reviewing relative ADRs at sprint planning

Architecture Review Board meetings – review ADRs or at least have the ADRs to fuel discussions

**THANK YOU !**

**SEAN WHITESELL**

**SR. CLOUD ARCHITECT AT ARCHITECTNOW**

**MICROSOFT MVP**

**TULSA .NET USER GROUP LEADER**

**GitHub: seanw122/presentations**

**Web: seanwhitesell.com**

**Twitch: codewithsean**

**Twitter: codewithseanw**

