

# AWS re:Invent

DECEMBER 2 - 6, 2024 | LAS VEGAS, NV

ARC202-R

# A capability-oriented approach to defining your cloud architecture

**Sushanth Mangalore**

Sr. Solutions Architect  
Amazon Web Services

**Mukund Rao**

Worldwide Business Consulting SA  
Amazon Web Services



© 2024, Amazon Web Services, Inc. or its affiliates. All rights reserved.

A photograph of a large audience of people sitting in rows, facing towards the left. They are all clapping their hands. The lighting is dim, with blue and purple hues, typical of a conference or event setting.

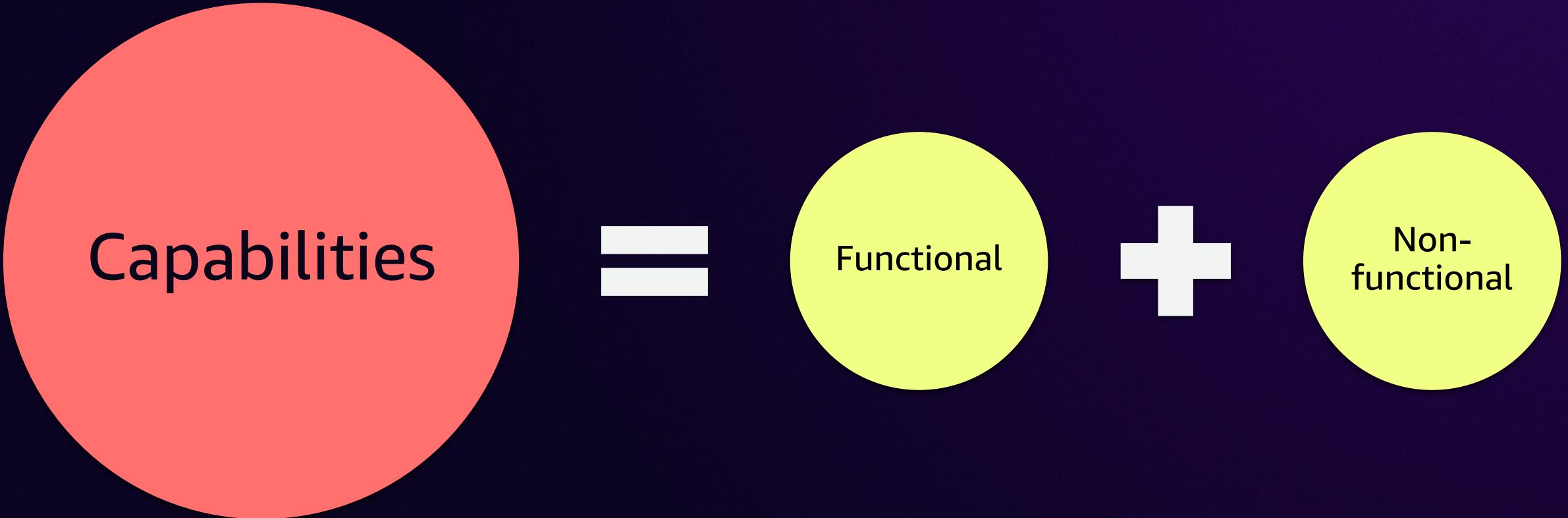
**“Architecting is a series of trade-offs”**

**- 3<sup>rd</sup> Law of The Frugal Architect**

**Dr. Werner Vogels**

VP and CTO, Amazon Web Services

# Thinking in capabilities



# Translating business goals to technical needs



Business persona

*"We want to improve our **time to market**"*



Technology persona

*"We must **improve our customer satisfaction**"*

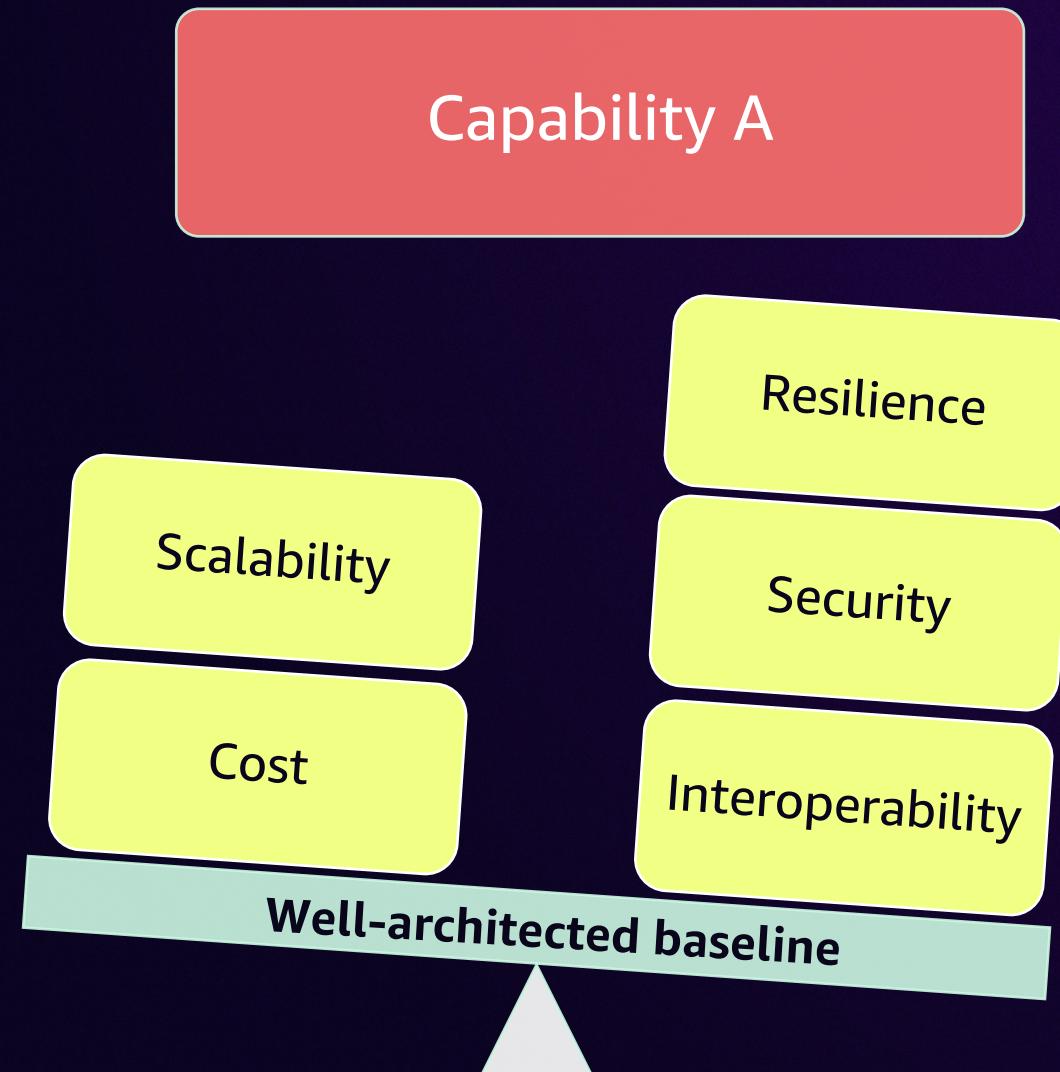
*"We need to **improve our margins**"*

Agility, testability, **deployability**

**Performance**, resiliency, availability

**Cost efficiency**, simplicity, maintainability

# Identifying architectural characteristics



# Architectural characteristics in AWS services

## Amazon DynamoDB

A **fully managed, serverless**, key-value NoSQL database designed to run **high-performance** applications **at any scale**

# Architectural characteristics in AWS services

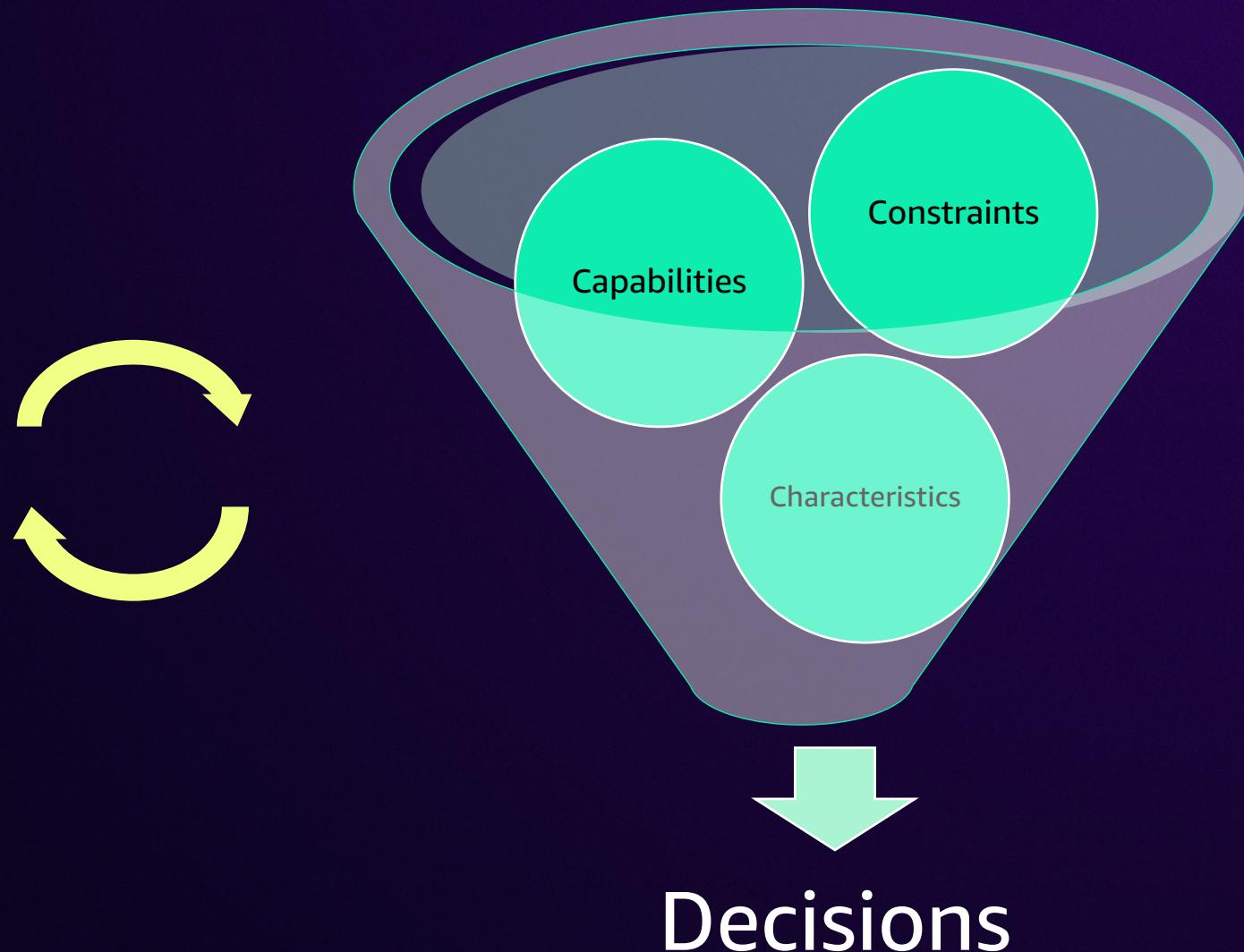
## Amazon ECS

A **fully managed** container orchestration service that **simplifies** your **deployment, management**, and **scaling** of containerized applications

# Documenting architectural decisions

## Example guidelines

- Avoid vendor lock-in
- No serverless
- Windows shop
- Build over buy



# Architectural Decision Records (ADR)

An *architectural decision record* (ADR) is a document that describes a choice the team makes about a significant aspect of the software architecture they're planning to build. Each ADR describes the *architectural decision*, its *context*, and its *consequences*. ADRs have states and therefore follow a lifecycle.

# Documenting architectural decisions

## ADR Template

Title: Architectural Decision Record (ADR #1)

Status: Proposed / Under Review / Accepted

Context: Problem statement / Alternatives

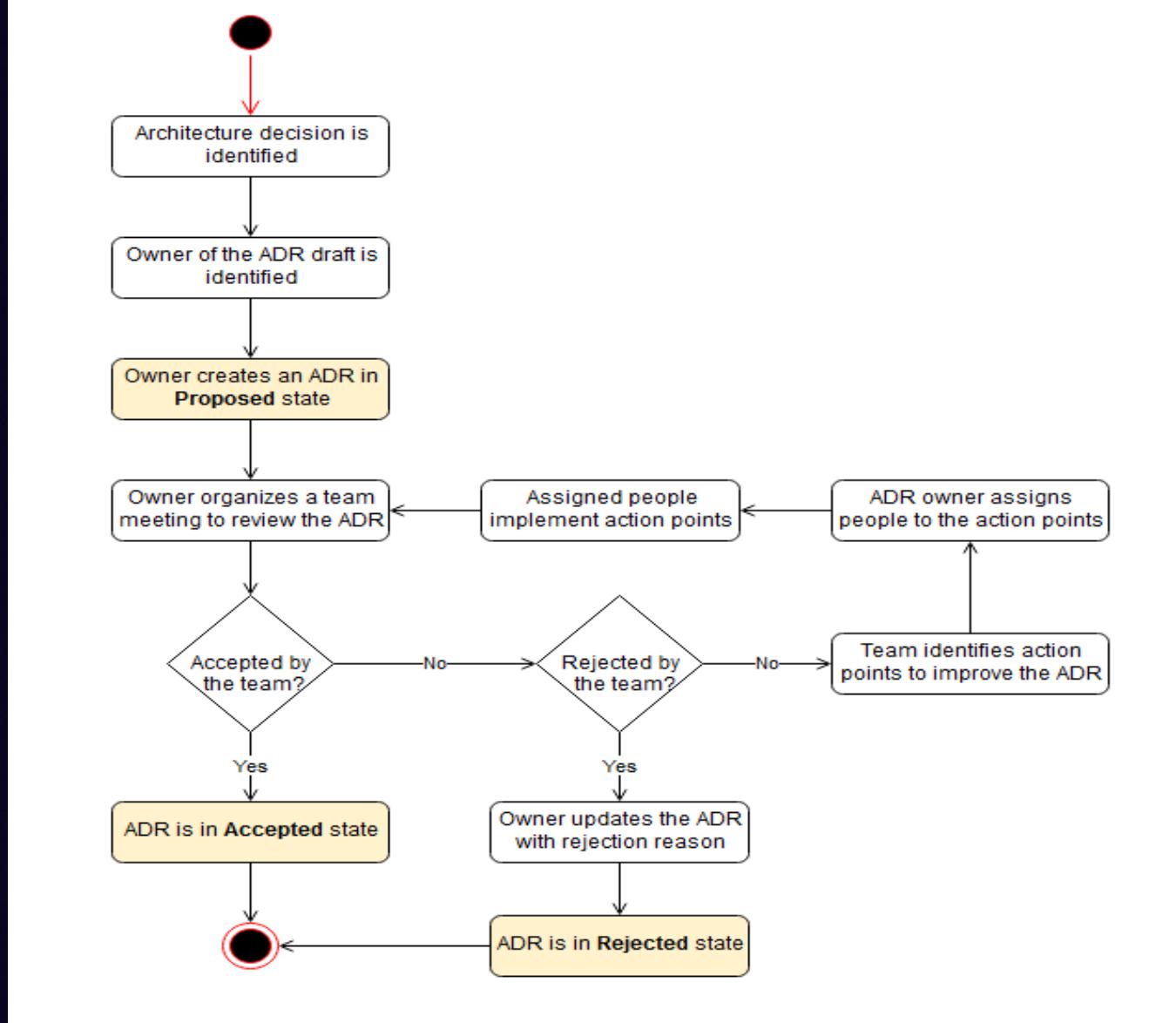
<Stakeholders>: Architect, Lead Engineer, Product Manager - Names

<Characteristics>: List the characteristics influencing the decision

Decision: We will ....

Consequences: Trade-offs / fall backs / 2-way doors

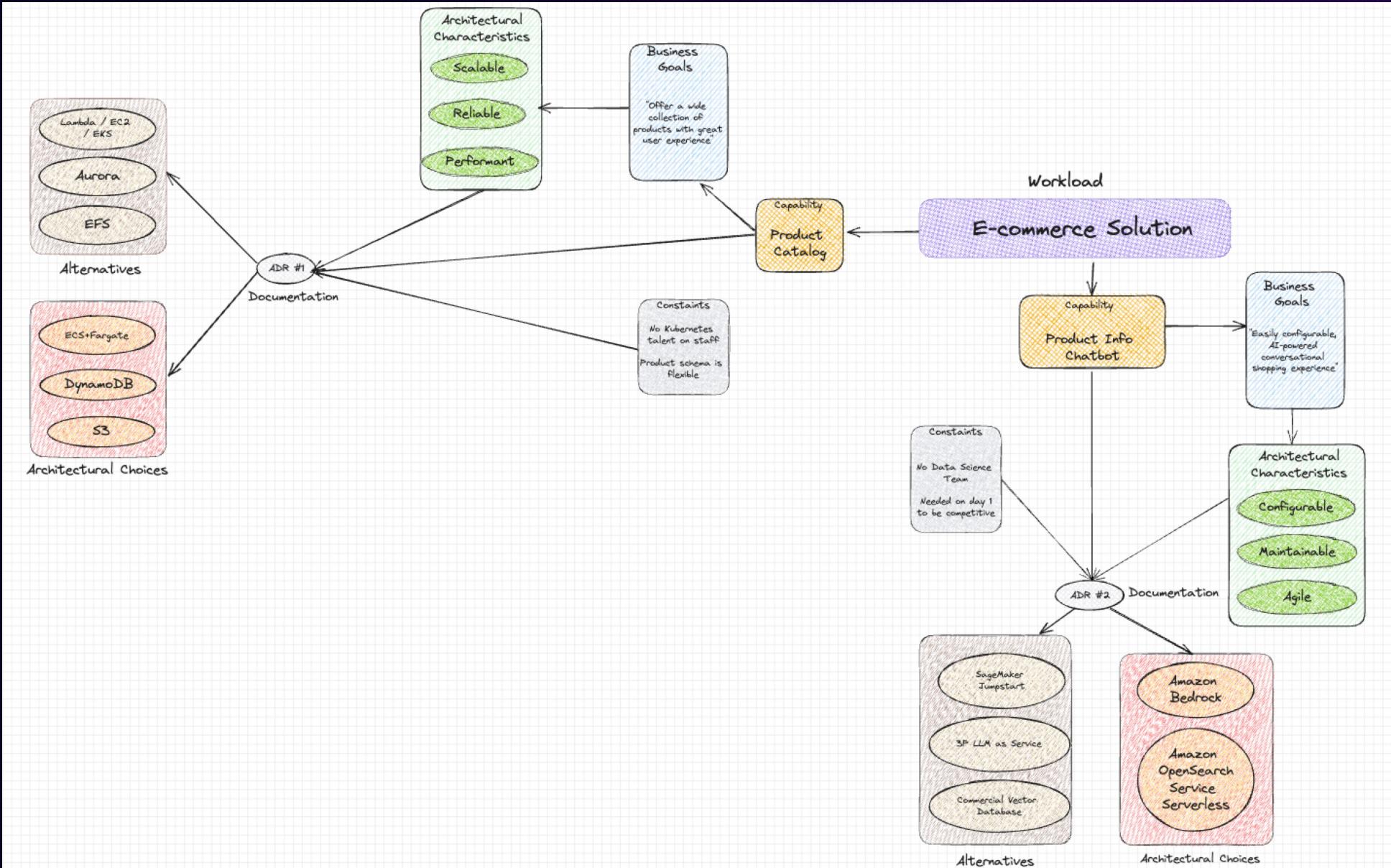
# ADR Process



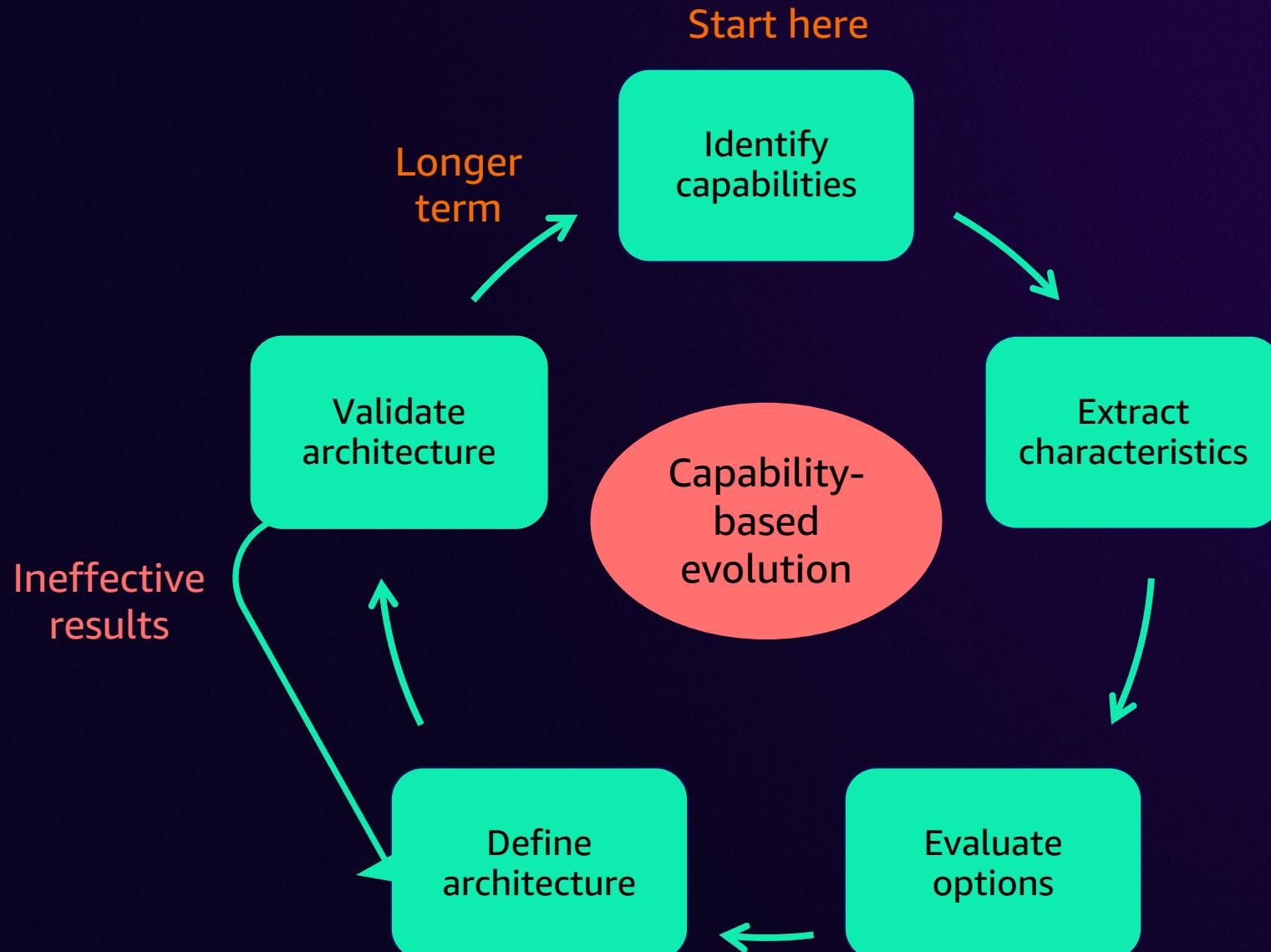
# ADR Sample

Title:	ADR #1 - Product Catalog
Status:	Accepted
Context:	We need to select the Compute, Database and Storage options to implement the Product Catalog that aligns with our business goals.
Stakeholders:	Bob - Solutions Architect, Sandy - Lead Engineer, Ravi - Product Manager
Decision:	We have decided to select Amazon ECS with Fargate as our compute option, DynamoDB as our database and Amazon S3 as the storage. This is in line with required architectural characteristics of scalability, durability and performance. These characteristics were chosen based on our business goals of rapid customer growth through wide variety of product selection and excellent user experience at all times, including periodic flash sales
Alternatives Considered:	EKS/EC2/Lambda - We expect consistent site usage, so Lambdas may not suit. We do not have Kubernetes expertise in-house, so we ruled out EKS. We prefer to use containers for better resource utilization, portability and dependency management, so we ruled out deploying on EC2 directly. With ECS also, Fargate was preferred for easier maintenance. Aurora - Ruled out as DynamoDB will scale better for our needs and we want to allow flexible schema for products. EFS - Ruled out as S3 offers better durability and is cost effective for static assets
Consequences:	Using Fargate will mean that we will have always-on compute, but we will use autoscaling to scale out and scale in based on the demand. DynamoDB means we will need to ensure we understand our data access patterns well, so we can design our tables and indices for efficient and performant querying. Picking S3 means we are expecting product catalog to only need static assets. In the future, if we require file-system type of access, we may need to consider other options like EFS.
Characteristics:	Scalability, reliability, performance

# Capability to solution tracing



# Mental Model



# Thank you!

**Sushanth Mangalore**

mangalos@amazon.com



sushanthmangalore

**Mukund Rao**

mukundpr@amazon.com



mukund1989



Please complete the session  
survey in the mobile app



© 2024, Amazon Web Services, Inc. or its affiliates. All rights reserved.