Azure Application Architecture Guide

08/30/2018 • 2 minutes to read • Contributors

In this article

This guide presents a structured approach for designing applications on Azure that are scalable, resilient, and highly available. It is based on proven practices that we have learned from customer engagements.

Introduction

The cloud is changing the way applications are designed. Instead of monoliths, applications are decomposed into smaller, decentralized services. These services communicate through APIs or by using asynchronous messaging or eventing. Applications scale horizontally, adding new instances as demand requires.

These trends bring new challenges. Application state is distributed. Operations are done in parallel and asynchronously. The system as a whole must be resilient when failures occur. Deployments must be automated and predictable. Monitoring and telemetry are critical for gaining insight into the system. The Azure Application Architecture Guide is designed to help you navigate these changes.

Traditional on-premises	Modern cloud
Monolithic, centralized	Decomposed, de-centralized
Design for predictable scalability	Design for elastic scale
Relational database	Polyglot persistence (mix of storage technologies)
Strong consistency	Eventual consistency
Serial and synchronized processing	Parallel and asynchronous processing
Design to avoid failures (MTBF)	Design for failure (MTTR)
Occasional big updates	Frequent small updates
Manual management	Automated self-management
Snowflake servers	Immutable infrastructure

This guide is intended for application architects, developers, and operations teams. It's not a how-to guide for using individual Azure services. After reading this guide, you will understand the architectural patterns and best practices to apply when building on the Azure cloud platform. You can also download an e-book version of the guide.

How this guide is structured

The Azure Application Architecture Guide is organized as a series of steps, from the architecture and design to implementation. For each step, there is supporting guidance that will help you with the design of your application architecture.

Architecture styles

The first decision point is the most fundamental. What kind of architecture are you building? It might be a microservices architecture, a more traditional N-tier application, or a big data solution. We have identified several distinct architecture styles. There are benefits and challenges to each.

Learn more:

- Architecture styles

Technology choices

Two technology choices should be decided early on, because they affect the entire architecture. These are the choice of compute service and data stores. *Compute* refers to the hosting model for the computing resources that your applications runs on. *Data stores* includes databases but also storage for message queues, caches, logs, and anything else that an application might persist to storage.

Learn more:

- Choosing a compute service
- Choosing a data store

Design principles

We have identified ten high-level design principles that will make your application more scalable, resilient, and manageable. These design principles apply to any architecture styles. Throughout the design process, keep these ten high-level design principles in mind. Then consider the set of best practices for specific aspects of the architecture, such as auto-scaling, caching, data partitioning, API design, and others.

Learn more:

- Design principles

Quality pillars

A successful cloud application will focus on five pillars of software quality: Scalability, availability, resiliency, management, and security. Use our design review checklists to review your architecture according to these quality pillars.

- Quality pillars