Arkitektur som kod

Infrastructure as Code i praktiken

Kodarkitektur Bokverkstad

Contents

Chapter 1

Inledning till arkitektur som kod

Arkitektur som kod (Architecture as Code) representerar ett paradigmskifte inom systemutveckling där hela arkitekturen - från applikationer till infrastruktur - definieras, versionshanteras och hanteras genom kod. Detta approach möjliggör samma metodiker som traditionell mjukvaruutveckling för hela IT-landskapet.

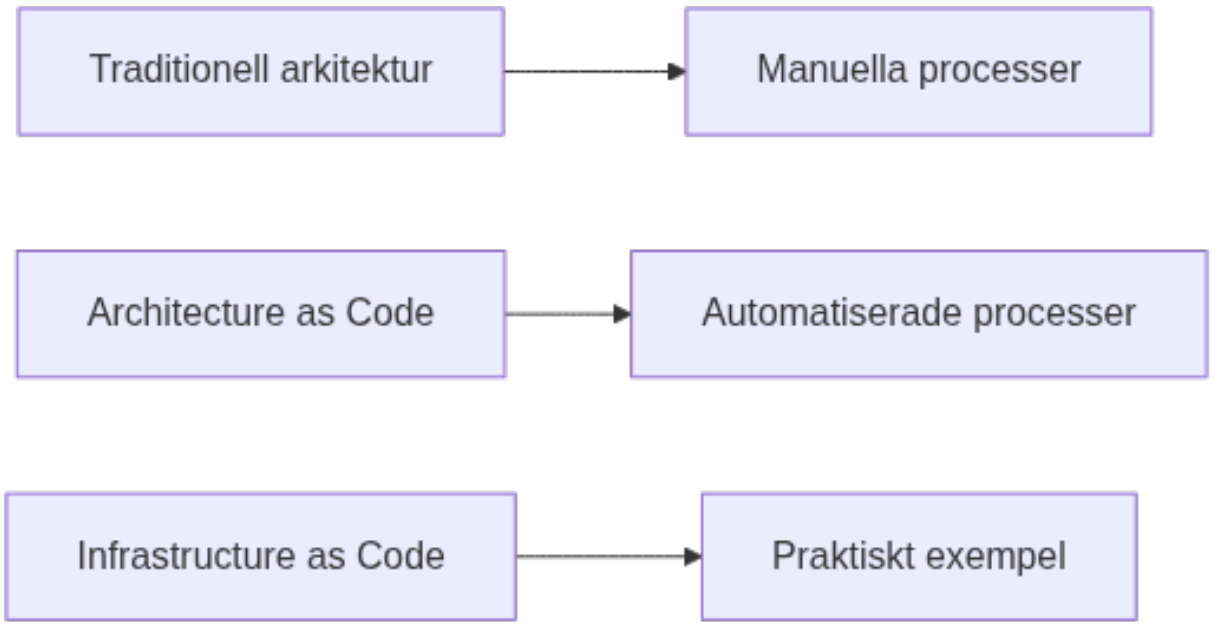

Figure 1.1: Inledning till arkitektur som kod

Diagrammet illustrerar evolutionen från manuella processer via Infrastructure as Code till den omfattande visionen av Architecture as Code, där hela systemarkitekturen kodifieras.

1.1 Från Infrastructure as Code till Architecture as Code

Infrastructure as Code (IaC) var det första steget mot kodifiering av IT-resurser. Genom att behandla infrastruktur som kod uppnåddes automatisering, reproducerbarhet och versionskontroll av serverresurser, nätverk och molnresurser.

Architecture as Code bygger vidare på denna grund men omfattar ett bredare perspektiv. Medan IaC fokuserar på infrastrukturkomponenter, inkluderar Architecture as Code även applikationsarkitektur, dataflöden, säkerhetspolicies, compliance-regler och organisatoriska strukturer - allt definierat som kod.

1.2 Definition och omfattning

Architecture as Code definieras som praktiken att beskriva, versionhantera och automatisera hela systemarkitekturen genom maskinläsbar kod. Detta omfattar inte bara infrastrukturen utan även applikationskomponenter, integrationsmönster, dataarkitektur och organisatoriska processer.

Denna holistiska approach möjliggör end-to-end automatisering där förändringar i krav automatiskt propagerar genom hela arkitekturen - från applikationslogik via infrastruktur till deployment och monitering.

1.3 Bokens syfte och målgrupp

Denna bok vänder sig till systemarkitekter, utvecklare, devops-ingenjörer och projektledare som vill förstå och implementera Architecture as Code i sina organisationer. Infrastructure as Code behandlas som ett viktigt praktiskt exempel och grundpelare, men inte som det enda fokuset.

Läsaren kommer att få omfattande kunskap om hur hela systemarkitekturen kan kodifieras, från grundläggande IaC-principer till avancerade arkitekturmönster som omfattar hela organisationens digitala ekosystem.

Källor: - ThoughtWorks. “Architecture as Code: The Next Evolution.” Technology Radar, 2024. - AWS. “Infrastructure as Code Best Practices.” Amazon Web Services Documentation. - Morris, K. “Infrastructure as Code: Managing Servers in the Cloud.” O’Reilly Media, 2020. - Martin, R. “Clean Architecture: A Craftsman’s Guide to Software Structure.” Prentice Hall, 2017.

Chapter 2

Grundläggande principer för Architecture as Code

Architecture as Code bygger på fundamentala principer som säkerställer framgångsrik implementation av kodifierad systemarkitektur. Dessa principer omfattar och bygger vidare på Infrastructure as Code (IaC) men sträcker sig till hela systemlandskapet.

Figure 2.1: Grundläggande principer diagram

Diagrammet visar det naturliga flödet från deklarativ kod genom versionskontroll och automatisering till reproducerbarhet och skalbarhet - de fem grundpelarna inom Architecture as Code.

2.1 Deklarativ arkitekturdefinition

Den deklarativa approachen inom Architecture as Code innebär att beskriva önskat systemtillstånd på alla nivåer - från applikationskomponenter till infrastruktur. Detta skiljer sig från imperativ programmering där varje steg måste specificeras explicit.

Infrastructure as Code är ett praktiskt exempel på deklarativ definition, där verktyg som Terraform eller CloudFormation beskriver infrastrukturens önskade tillstånd. Architecture as Code utvidgar detta till att omfatta applikationsarkitektur, API-kontrakt och organisatoriska strukturer.

2.2 Helhetsperspektiv på kodifiering

Medan Infrastructure as Code fokuserar på infrastrukturresurser, omfattar Architecture as Code hela systemekosystemet. Detta inkluderar applikationslogik, dataflöden, säkerhetspolicies, compliance-regler och till och med organisationsstrukturer.

Ett praktiskt exempel är hur en förändring i en applikations API automatiskt kan propagera genom infrastrukturdefinitioner, säkerhetskonfigurationer och dokumentation - allt eftersom det är definierat som kod.

2.3 Immutable architecture patterns

Principen om immutable arkitektur bygger vidare på Infrastructure as Code:s immutable infrastruktur men applicerar det på hela systemarkitekturen. Istället för att modifiera befintliga komponenter skapas nya versioner som ersätter gamla på alla nivåer.

Detta skapar förutsägbarhet och eliminerar architectural drift - där system gradvis divergerar från sin avsedda design över tid.

2.4 Testbarhet på arkitekturnivå

Architecture as Code möjliggör testning av hela systemarkitekturen, inte bara enskilda komponenter. Detta inkluderar validering av arkitekturmönster, compliance med designprinciper och verifiering av end-to-end-flöden.

Infrastructure as Code-testning utgör en viktig del av denna helhetssyn, men kompletteras med arkitekturtester som validerar designbeslut och systemkomplexitet.

Källor: - Fowler, M. “Infrastructure as Code: Patterns and Practices.” Martin Fowler Blog. - Red Hat. “Architecture as Code Principles and Best Practices.” Red Hat Developer. - Google Cloud. “Infrastructure as Code on Google Cloud.” Google Cloud Architecture Center.

Chapter 3

Versionhantering och kodstruktur

Effektiv versionhantering utgör ryggraden i Infrastructure as Code-implementationer. Genom att tillämpa samma metoder som mjukvaruutveckling på infrastrukturdefinitioner skapas spårbarhet, samarbetsmöjligheter och kvalitetskontroll.

Figure 3.1: Versionhantering och kodstruktur

Diagrammet illustrerar det typiska flödet från Git repository genom branching strategy och code review till slutlig deployment, vilket säkerställer kontrollerad och spårbar infrastrukturutveckling.

3.1 Git-baserad arbetsflöde för infrastruktur

Git utgör standarden för versionhantering av IaC-kod och möjliggör distribuerat samarbete mellan team-medlemmar. Varje förändring dokumenteras med commit-meddelanden som beskriver vad som ändrats och varför, vilket skapar en komplett historik över infrastrukturutvecklingen.

3.2 Kodorganisation och modulstruktur

Välorganiserad kodstruktur är avgörande för maintainability och collaboration i större IaC-projekt. Modulär design möjliggör återanvändning av infrastrukturkomponenter across olika projekt och miljöer.

Källor: - Atlassian. “Git Workflows for Infrastructure as Code.” Atlassian Git Documentation.