Principper

De centrale principper følger Fællesoffentlig Digital Arkitektur og OS2's værdier for at sikre en fleksibel, robust og bæredygtig løsning.

- Åbenhed: Brug åbne standarder og OSI-godkendte licenser.
- Genbrug: Komponenter udvikles, pakketeres og leveres som uafhængige løskoblede komponenter for at sikre genbrugspotentiale på tværs af projekter
- Portabilitet: Løsningen skal kunne flyttes og driftes på tværs af driftscentre og leverandører uden andet end tilpasning af konfiguration.
- Digital Suverænitet: Fælles ejerskab af løsningen skal kontinuerligt sikres ved at arbejde aktivt med kildekode, dokumentation, bygge og pakketerings pipelines og udrulnings skabeloner i os2 ejede versionstyrede repositorier.
- Skalerbarhed: Skalerbarhed, resilliens og sikkerhed implementeres på platformss-niveau via åbne moderne Clouod Native standarder.
- Interoperabilitet: Funktionalitet eksponeres via åbne og veldokumenterede API'er, så integration med andre systemer er enkel og robust.

Operationalisering og implementering af principper

Dokumentation

Projektet skal levere dokumentation for design, drift, test og brug for at sikre ensartethed, læsbarhed, relevans og konsistens gennem hele projektets levetid

- Genbrugelig og læsbar: Dokumentation skal være ensartet, struktureret og let at forstå.
- Versionsstyret: Dokumentation for design, drift, test og brug placeres som Markdown i et Git-baseret repository og følger samme principper som kode (jf. OpenGitOps principper).
- Relevant: Generering og validering af dokumentation indgår i CI/CD-pipelines for at sikre relevans, konsistens og reduktion af manuelle fejl.

Udrulning og Drift

For at fremmer transparens, reducere risikoen for manuelle fejl og sikre, at driftsmiljøet altid er i overensstemmelse med den ønskede tilstand, skal løsningen sikre sporbarhed og historik på alle ændringer i udrulning og drift.

- Sporbarhed: Historik og transparens sikres via deklarativt defineret og versionsstyret drift (jf. OpenGitOps principper). Alle driftsændringer håndteres via pull requests og review.

- Kontinuitet: Software-agenter trækker automatisk den ønskede tilstand fra et udrulningsspecifikt repository ejet af OS2. Systemet bringes løbende i overensstemmelse med den ønskede tilstand uden manuel indgriben.

- Ensartethed: Løsningen pakketeres som OCI-compliant containere og tagges med release-tags. Pakketering og udrulning foregår automatiseret via CI/CD-pipeline i OS2's repository for at reducere manuelle fejl og sikre ensartet kvalitet.

- Skalerbarhed og portabilitet: Platformen der udrulles på, understøtter dynamisk skalering og drift på tværs af driftscentre, hvilket sikrer den rette fleksibilitet og afbøder risici for fastlåsning til en enkelt leverandør.
- Synlighed og kontrol: Overvågning, sporbarhed og rapportering etableres som en del af infrastrukturen via åbne standarder. Formålet er at skabe tillid til stabilitet, compliance og kvalitet uden at belaste applikationsudviklingen.

Governance og Open Source

For at sikre åbenhed og kompatibilitet accepteres kun OSI-godkendte licenser og der stilles krav om åben udvikling og styring direkte i den OS2 anviste versionsstyring.

- Ejerskab: Al kode udvikles aktivt i et åbent Git-baseret repository ejet af OS2 for at sikre kontinuitet, fælles ansvar og mulighed for genbrug. Leverancer af kode, dokumentation og konfigurationer sker via den indbyggede issue-tracker i git, pull/merge requests, og en dokumenteret reviewproces.
- Transparens: Historik på leverancerne sikres med fx Conventional Commits, så ændringer er ensartede, sporbare, forståelige og nemme at auditere.

Anvendte metoder og best practice:

OpenGitOps

OpenGitOps er en tilgang til drift, der fokuserer på deklarativ konfiguration og versionsstyring. Det sikrer, at alle ændringer i driftsmiljøet er dokumenteret og sporbare. Ved at bruge Git som en enkelt kilde til sandhed (Single Source of Truth) sikres det, at alle teammedlemmer har adgang til den samme information og kan følge ændringerne over tid. Dette fremmer transparens, reducerer risikoen for manuelle fejl og sikrer, at driftsmiljøet altid er i overensstemmelse med den ønskede tilstand.

Læs mere:

> - OpenGitOps: https://opengitops.dev/
> - GitOps-principper: https://www.gitops.tech/

Everything as Code

Everything as Code er en praksis, hvor al konfiguration af systemer og applikationer håndteres som kode. Dette betyder, at konfigurationer versionsstyres, testes og udruleres på samme måde som applikationer. Det sikres, at konfigurationer er reproducerbare, konsistente og nemme at vedligeholde. Ved at behandle konfiguration som kode reduceres risikoen for manuelle fejl, og det bliver lettere at rollbacke til tidligere versioner, hvis der opstår problemer. Dette fremmer også samarbejdet mellem udviklere og driftsansvarlige ved at skabe en fælles forståelse af systemets konfiguration.

Læs mere:

> - What is Everything as Code?: https://octopus.com/blog/what-is-everything-as-code
> - Open Practice Library: Everything as Code: https://openpracticelibrary.com/practice/everything-as-code/

> - Martin Fowler on Infrastructure as Code: https://martinfowler.com/bliki/InfrastructureAsCode.html
> - The New Stack: Automate DevOps with an Everything-as-Code Approach: https://thenewstack.io/automate-devops-with-an-everything-as-code-approach/
> - HashiCorp: Everything as Code - The Future of Ops Tools: https://www.hashicorp.com/en/resources/everything-as-code-the-future-of-ops-tools

12/15 Factor App

12/15 Factor App-principperne er en samling af best practices for at opbygge robuste, skalerbare og vedligeholdelige applikationer. Disse principper inkluderer løskobling, konfiguration, skalerbarhed, resiliens og observabilitet. De sikrer, at applikationer er fleksible, pålidelige og nemme at vedligeholde.

Læs mere:

> - Evolving Twelve-Factor: Applications to Modern Cloud-Native Platforms - https://12factor.net/blog/evolving-twelve-factor
> - IBM Developer: 15-Factor Applications: https://developer.ibm.com/articles/15-factor-applications/
> - The Twelve-Factor App: https://12factor.net/
> - The Fifteen-Factor App explained: https://domenicoluciani.com/2021/10/30/15-factor-app.html

Cloud Native

Cloud Native er en tilgang til at udvikle og drifte applikationer, der er designet til at udnytte fordelene ved cloud-infrastruktur. Ved at levere i cluster baserede driftcentre med Cloud Native Kubernetes, kan applikationer skalere dynamisk, sikre høj tilgængelighed, automatisere driften og fremme løskobling. Denne tilgang sikrer, at applikationer er robuste, skalerbare og nemme at vedligeholde og da en Cloud Native platform automatisk tilpasser ressourcer efter behov, sikres optimal performance og betydelig reduktion af omkostninger.

Læs mere:

> - IBM Think - What is cloud native? https://www.ibm.com/think/topics/cloud-native
> - What is cloud native architecture? - https://intercept.cloud/en-gb/blogs/what-is-cloud-native-architecture