Software Engineering Principles

Consistent Environment

Implemented on (date)	20/05/2024
Approved by (name & role)	Head of Software Practices (Fahad Anwar) In consultation with TAG (Technical Advisory Group).
Last review on (date)	16/05/2024
Reviewed by	TAG (Technical Advisory Group)
Next review due on (date)	16/05/2025
Principle owner (name)	Head of Software Practices In consultation with Technical Advisory Group (TAG) representing Software Engineers, Cloud Division, TISS and Security Division.
Principle owner (division)	Digital Services and Technology (DST)
Main point of contact (name)	Fahad Anwar Software Engineering Head of Practice
Status	Approved

Principle Review Record

This Review Record is to be completed on each time a review is conducted. Its purpose is to maintain a record of reviews, recording who conducted the review, the date of the review and the outcome of the review (fit for purpose, amendment required, principle no longer required, etc).

This principle is to be reviewed annually.

Review No	Review Conducted By	Review Date	Review Outcome
01			

Amendment Details

Date	Amendment Summary	Amended by	Version

RASCI (For detail please – RASCI Information document)

Responsible	G6 Program Managers through Technical Leads, G7 and SEO's
Accountable	Head of Software Practices
Supportive	Head of Cloud Functions (Amazon, GCP, Azure) SAIM SIRA Software Engineering Community of Practice (SE-CoP)
Consulted	Technical Advisory Group (TAG) representing Software Engineers, Cloud Division, TISS and Security Division.
Informed	Senior Leadership Team

	Software Engineering Community SAIM SIRA Design Authority Chair

Consistent Environments

Environments of the given system should have homogeneous application configuration, software, operating system, infrastructure and data (where appropriate).

Rationale

Consistency between environments reduces the time wasted investigating issues due to discrepancies, which improves the speed of delivery for the business. This consistency also helps increase our confidence in what we build and its safe delivery in production environment.

Implications

- Configuration will need to be captured in version-controlled code.
- Production, Test/QA and Development environments should be configured consistently.
- Software packages that are not amenable to being consistently configured will be less desirable.
- Software that is free, or provides cheaper licenses for development and testing, will be preferable.
- We can perform push-button deployments of any version of the software to any environment on demand.
- A single trunk-based CI build at the beginning of the pipeline is strongly recommended (for example https://cloud.google.com/architecture/devops/devops-tech-trunk-based-development).
- We will need to invest heavily in making data available and consistent across environments (if possible)
- Creating (and destroying) new environments, including appropriate data sets, will need to be as simple and as fast as possible. Rely heavily on the principle of "Automate by Default".
- Environments should be idempotent on creations, updating and deletion; particularly where an "environment" is required to include a large set of (distributed) applications in order to facilitate end-to-end testing, the creation and maintenance of an environment can be very expensive. Reducing/limiting the number of environments may act as an enabling constraint – the fewer environments we have, the easier and cheaper it will be to keep them consistent.
- We will require consistent configuration of infrastructure.

Questions to be considered.

- If application is not in cloud, can we still have consistent environment (keeping in view the cost implications).
- You have considered the elements of environment which should be consistence such as encryption keys, certificates, passwords used in the production environment should NOT be used in any other environment.

- Similarly, access to environments should not blindly be consistent. The personnel that have privileged application access in test and dev environments should likely not have this same level of access in the PROD environment etc.
- Can we create temporary environments for feature level testing?
- Where consistent data is not possible to the production level, have you ensured that the data is a close representation of real-world data? Can we use Infrastructure as Code to deploy our environments?
- Can we scale up and down environments for better FinOps management?

Motivation for the Principles

- https://www.gov.uk/guidance/the-technology-code-of-practice
- https://engineering-principles.jlp.engineering/
- https://github.com/otto-de/tech_manifest
- https://www.gov.uk/government/publications/security-policy-framework/hmg-security-policy-framework