Software Engineering Principles

Get Feedback Early and Often

Implemented on (date)	
Approved by (name & role)	Head of Software Practices (Fahad Anwar) In consultation with TAG (Technical Advisory Group).
Last review on (date)	13/03/2024
Reviewed by	TAG (Technical Advisory Group)
Next review due on (date)	
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	Final Draft
Published version link	

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

Get Feedback Early and Often

Gain feedback by frequent and early releases of functionality, rather than Big Bang releases.

Rationale

Whilst this represents a wider cultural shift, the engineering benefits are clear. Early and frequent releases through to Production provide feedback sooner, make code level integration easier, allow course corrections to be made earlier and provide business value more quickly. Problems are easier to identify and resolve in smaller changesets.

Implications

- Encourage all team members to help slice work into smaller units that can be tested and released independently.
- We encourage trunk-based development. Teams should adopt patterns such as Feature Flags and Branch by Abstraction to enable frequent deployments without resorting to long-lived branches.
- Prioritise keeping the software deployable over working on new features through constant investment.
- Requires regular assessment of feature flags and removal of the obsolete ones.
- Requires an automated, version-controlled configuration management strategy and delivery pipeline.
- No long-term development branches (and consequential complex merges) are needed.
- A common approach to controlling, measuring, and evaluating outcomes of experiments is needed (within the context of the project), particularly when systems are broken down into Small and Simple pieces.

Questions to be considered

- Are you implementing actual or anticipated requirements?
- Did you design using a BDUF (Big Design Up Front) approach or MVP?
- Do your service interfaces and protocols rely on well-known standards or are they tightly coupled to a proprietary 3rd party tool/protocol?
- Does your software rely on well-known standards or are they tightly coupled to a proprietary 3rd party tool/protocol?
- What are the high-level goals and objectives of the solution and has a MVP been defined?
- Have you identified the initial release candidate functionality and understand what iterative releases may need to contain to meet the minimum requirements of the user?
- Can the solution be split into modules or independent component pieces of functionality to speed up development and delivery?

- How will frequent releases be demonstrated to the end user and how will user feedback be collected, prioritised and actioned?

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

Principle Enforcement

G6 Program Managers through Technical Leads or Sr. member of respective software development team is responsible for monitoring and enforcing principle compliance.

Exception process

It is acknowledged that situations arise in which the Principal Detail as above may not be able to be met. Where this is the case, a documented Principal exception must be sought from the Principal Owner (specified in the table on the title page).