servicenow

QUICK REFERENCE GUIDE

Release and Instance Strategy

Governance requires careful planning and design from the start. Defining the governance steps helps with release and instance management. Instance management is about determining instance structure and defining policies to support the platform activities.

Governance steps

Consider the following points when developing governance steps:

- As with all governance, it is important to find the right balance between risk mitigation (control) and value realization (speed)
- There should be a clear process and policy around what requirements must be met before anything (code/data) can move into an environment
- The governance steps must consider the customer's operating model, the scope of the platform usage, the instance structure, and the implementation methodology

Four governance considerations for release and instance management

- Operating model
- Platform scope
- Instances
- Methodology

Instance structure

Define the instance structure by determining the instance stack to successfully support the platform's activities. Examples of instance structures are:

- Three instance stack: development, testing, and production
- Four instance stack: development, quality assurance, user acceptance, production
- Five instance stack: development, quality assurance, user acceptance testing, staging, and production

Benefits of defining Instance management policies

Defining policies to manage each instance included in the structure helps the organization in many ways including:

- Reducing risk by making sure that changes to instances happen using a defined process and within predefined maintenance windows
- Reducing the mean time to resolve (MTTR) incidents or outages to your instance by clearly defining support procedures and teams
- Preventing incidents and outages by defining the process to modify system properties or enable plugins

Considerations for multi-production instances include

There may be criteria that lead customers to implement multiple production instance. It is essential that there is a sound business reason for multi-instance architecture. Some reasons are:

- Instances split by release or product
- Instances sent to external developer
- Instance hosted in another country

Out of the box supported code migration options

- Application repositories
- Source control integrations
- Update sets
- Team development