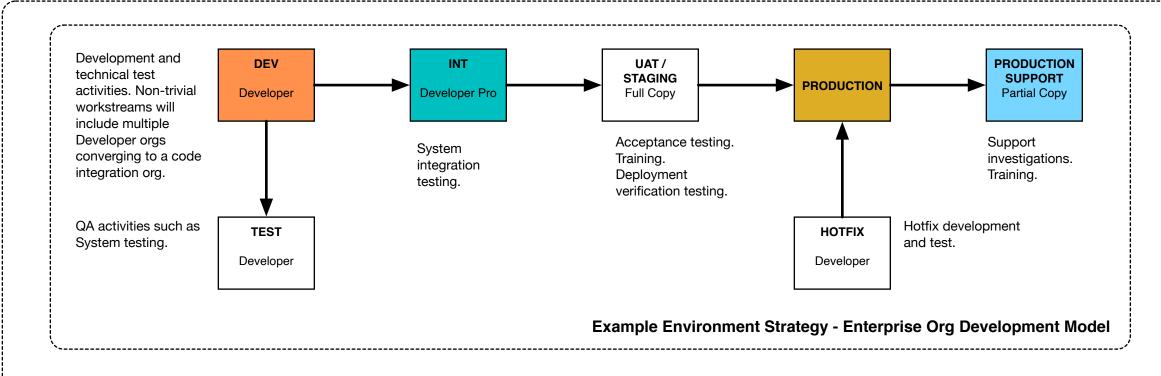
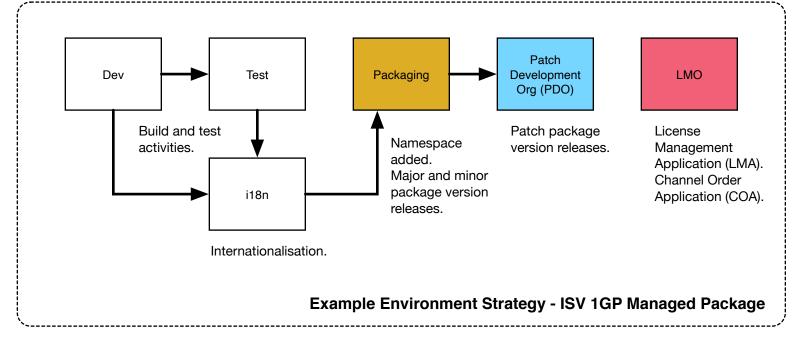


#### Salesforce Core Platform

Org Strategy

#### The org strategy defines how many Salesforce production orgs will be implemented for the Enterprise. Single Org Considerations: (1) Single-org configurations require consideration to separation of concerns. (2) Data synchronisation and consolidated reporting are streamlined, as is user licensing. Multiple Org Considerations: (1) Multi-org architectures can partition configuration, processing and platform limits based on geography, business area or function. (2) License cost will increase where users require multi-org access. Org (3) Platform tools such as Salesforce-to-Salesforce support data synchronisation and hub-and-spoke models.





**Environment Strategy** 

#### Test Strategy

#### Test Strategy drives project teams to define:

(1) Roles and responsibilities.

(2) Tools and process for reporting/defect management. (3) Test methodology (integration into the Agile process etc.);

(3.1) System Testing <- functional tests which validate specific test cases & conditions. (3.2) Regression Testing <- tests executed to guickly validate the impact of change. (3.3) Technical Testing <- tests which validate code behaviour and tests which focus on non-functional test cases and conditions (performance, security, scalability etc.). (3.4) Acceptance Testing <- functional tests with clear acceptance criteria through which user-acceptance is measured.

### Test Practices:

(1) Manual Testing <- manual execution of system test scripts against defined or exploratory conditions.

(2) Test Automation <- automated execution of system test scripts to reduce manual effort or integrated into deployment processes for regression test purposes. (3) Unit Testing <- code-written tests which validate code behaviour. Salesforce requires a 75% coverage level (for deployment) which means 75% of non-test code must be validated by unit tests.

(3) Test Driven Development (TDD) <- The TDD practice advances the principle of testfirst coding; meaning test code is written first.

## Salesforce Data Mask:

Effective testing is reliant upon representative data. Historically (and typically for expediency) this requirement has often resulted in the use of un-obfuscated customer data, particularly where Full Copy or Partial sandboxes are available. This approach to test data has significant issues with PII in relation to Data Protection regulation. Salesforce Data Mask provides a platform-level capability which enables Sandbox data to be automatically obfuscated using flexible rules applied to different types of private data (replace with random characters, similar words, pattern masking, delete).

#### **Development Strategy Sandboxes** Salesforce provides Sandbox orgs (by Edition or add-on) for development and test activities. Sandboxes are created from Production or cloned from another sandbox and refreshed over time. Fully Copy Sandboxes provide a replica of the Production org typically used for UAT, staging, deployment **Full Copy** verification, training or support. Metadata and data. metadata+data Refresh interval: 29 days Storage: as per production **Sandbox Template** Partial Copy Sandboxes are intended for testing purposes. Sample data is included as specified by the Sandbox templates define the **Partial Copy** Sandbox template - up to 5GB. Metadata and data objects (Standard or Custom) metadata+data EE: 1 UE: 1 which are used to populate Full Refresh interval: 5 days Copy and Partial Copy sandboxes. Storage: 5GB data, files as-per production. Sandbox Cloning: Developer Pro Sandboxes provide higher volume A new sandbox can be created development and test environments suited to code as a clone of an existing sandbox i - d Developer Pro integration or data integration testing. Metadata only. metadata rather than a copy of the EE: 0 UE: 5 Production org. The sandbox Refresh interval: 1 day license type must be the same. Storage: 1GB data 1GB files Developer Sandboxes provide isolated Test or **Post Copy Interface:** Development environments. Metadata only. Developer Apex logic can be invoked during the EE: 25 UE: 100 metadata Sandbox copy process via the Refresh interval: 1 day SandboxPostCopy interface. Storage: 200MB data 200MB files

#### **Scratch Orgs DevHub:** A Production org (or ISV Business Org) used Dev Hub for Scratch Org creation. The Namespace Registry Scratch orgs are disposable orgs intended for use allows namespaces to be applied to Scratch Orgs. namespace within a source-driven development process (CI/CD). Limit ActiveScratchOrgs - EE: 40 UE: 100 namespace2 Scratch orgs are created from a DevHub org via the Limit DailyScratchOrgs - EE: 80 UE: 200 Salesforce CLI and are configurable in terms of edition and features enabled. Org Shape for Scratch Orgs Scratch Org: allows a Scratch org to be created with the features, A disposable org configured via Scratch Org Definition File (expires 7 days default, 30 days max). Scratch Scratch limits and settings taken from a Source Org. Scratch Metadata change tracking enables efficient pull and Org Org Orgs are empty and do not copy the metadata from a push of changes between the org and the local file Production org (as Sandboxes do). namespac system. Source shape is finer grained than the metadata API shape. **Development Models** Salesforce supports a variety of development models for both Enterprise and ISV developments.

**Unpackaged (Org development) — Enterprise:** Unlocked Package (Second Generation Packaging 2GP) — Enterprise : Changes are developed directly in the Production Org Unlocked packages are developed in Scratch Orgs and released as applications which or deployed from Sandboxes. Components are support upgrade. Deployed components are visible and editable in the target org. loosely structured by naming convention or Namespaces and version management. association. No version management. The lack of

structure impacts upon maintainability.

Unmanaged Package - Enterprise:

Changes are deployed between environments via

The lack of structure impacts upon maintainability.

unmanaged package. Component association to the

package is lost after installation. No version management.

Managed Package — ISV : 1GP Managed Packages are developed in Developer Edition Orgs and released as IP protected apps which support upgrade. Namespaces and version management.

> Managed Package (Second Generation Packaging 2GP) — ISV: 2GP Managed packages are developed in Scratch Orgs and released as IP protected apps which support upgrade. Deployed components are visible but not editable in the target org. Namespaces and version management.

## Governance

## Governance drives organisations to:

(1) Align to deliver business value.

(2) Identify, prioritise, execute & communicate business goals. (3) Maximise use of people, processes, knowledge & technology. (4) Ensure high value for all users & meet overall business goals.

## **Governance Framework Elements:**

(1) Centre of Excellence (CoE) <- management of governance (2) Change Management <- managing requirements to release, people and business process change. (3) Org Strategy <- design and structure of where capabilities (apps or functions) reside and run.

(4) Technical Governance <- guiding principles and standards intended to deliver continuous improvement and consistent technical excellence.

## **Foundational Processes:**

(1) Vision and Strategy <- outline the reason for the project in business terms and the measures to be applied. (2) Business Backlog <- prioritise requirements based on the business goals and value. (3) Software Development Lifecycle <- apply the established principles and methods of software engineering. (4) Communication Strategy <- structure to drive alignment between all stakeholders and support user adoption. (5) Data Strategy, Architecture and Engagement <- focus on design of the data model and the lifecycle management of the

data record. A robust data architecture is a key project enabler.

#### **Deployment Strategy Metadata API** The Metadata API is built on SOAP and WSDL or REST web service standards to provide read/write access to Salesforce metadata types. The API targets deployment use cases where metadata is moved between Salesforce orgs or between a Salesforce org and a Source Code Control repository via the local file system. **File-Based Declarative Development:** Asynchronous SOAP operations deploy() and retrieve() and the REST API resource POST services/data/vXX.0/metadata/deployRequest Headers; Content-Type: multipart/form-data, Authorization: Bearer token work with zip files containing a folder structure of metadata types and a package.xml manifest which describes the components to be deployed or retrieved. Optionally a destructiveChanges[Pre|Post].xml manifest is added for deletion. A zip file is returned containing the package.xml **CRUD-Development:** metadata component files. Synchronous SOAP API 10K file limit for retrieve/deploy -<Package> operations for CRUD 39MB zip, 400MB uncompressed. <members>\*</members> actions are supported at <name>ApexClass package.zip </types> the metadata type level.

createMetadata()

updateMetadata()

deleteMetadata()

.Apex1.cls

.Apex2.cls

package.xml

.Apex1.cls-meta.xml

.Apex2.cls-meta.xml

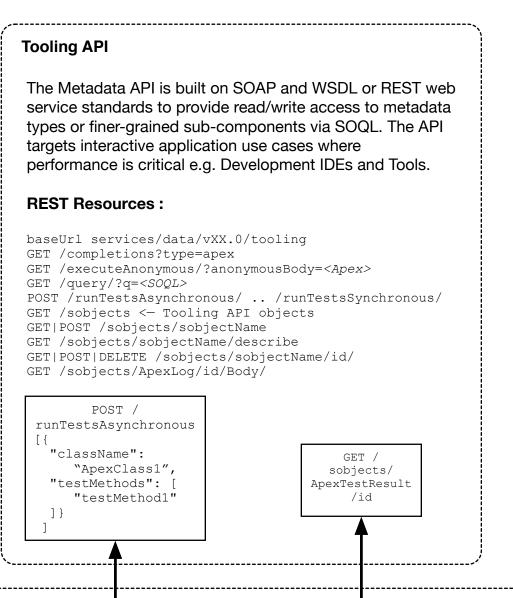
/Package>

**External Consumers** 

retrieve() is called

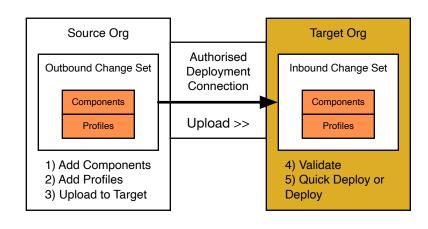
with a package

manifest.



# Required Permission: Modify Metadata Through Metadata API Functions **Change Sets**

Change Sets enable declarative metadata deployment between orgs that share the same parent Production org i.e. Sandbox=>Sandbox, Sandbox=>Production, Production=>Sandbox deployment paths. Each org to org Deployment Connection must be authorised before a Change Set can be uploaded.



## **Change Set Considerations:**

(1) Uploaded Change Sets can't be forwarded to an upstream org; instead the Change Set must be recreated. This limits the use of Change Sets for release management. (2) Change Sets do not support all Metadata types; Setup menu changes only. (3) Change Sets do not support partial deployment or component merge; deployment is

a one-way single transaction limited to 10,000 components. (4) Change Set deployments maintain user references across orgs

