



VA

U.S. Department
of Veterans Affairs

VA Salesforce Digital Transformation Center Standard Operating Procedure and Guidelines



Table of Contents

Table of Contents	2
Introduction.....	3
Deployment Management Plan	3
Intended Audience	3
High Level Overview.....	3
Deployment Management Plan	4
Introduction.....	4
VA Salesforce Environment Management.....	4
Deployment Process.....	7
Deployment Technologies	14
Version control / Source Code Management	15
Current Source Code Management Process.....	15
Future-State Source Code Management Processes	15
Regression Testing and Technical Debt	22
Guidelines for Test Classes	24
Apex Guidelines.....	25
Visualforce Guidelines	27
Development Process, Code Reviews, and Exception Requests	27
Appendix	29
Code Review Checklist.....	29
Pre-Deployment	32
Post-Deployment.....	34



Introduction

The purpose of this document is to provide an overview of the different functions performed by the Salesforce Administration at the Department of Veterans Affairs. The document is divided into the following sections:

- Deployment Management Plan
- Defect Management
- Support Models
- Quality Assurance
- Data Management
- Training
- User Account Maintenance
- Issues and Requirements Tracker

Deployment Management Plan

Intended Audience

The primary audience for this document are members of the project teams responsible for developing modules that provide business capabilities on the VA Salesforce platform. Project managers should make this document available to all team members supporting module deployment to ensure they are familiar with the processes described in this document and understand the activities required to ensure a successful deployment.

The secondary audience for this document are the VA Salesforce System Administration team and the VA Salesforce Configuration Management team. These teams will have a working knowledge of the processes described in this plan.

High Level Overview

The Department of Veterans Affairs (VA) Salesforce platform is a shared resource that enables projects across VA to quickly develop robust, flexible and scalable modules in a secure cloud environment. The VA Salesforce Digital Transformation Center (DTC) provides structure and guidance necessary to assist project teams maximize VA's investment in the platform, with the objective of helping project teams deliver quality solutions.

This document outlines the processes and technologies used to deploy VA Salesforce modules from development environments to the Systems Integration Testing (SIT), User Acceptance Testing (UAT), Staging, and Production (PROD) environments.



As use of the VA Salesforce platform grows, it will become increasingly important to manage changes to the configuration of the platform. This document describes the version control processes currently employed by the VA Salesforce platform and future version control processes that uses employ source control software to manage VA Salesforce metadata.

Deployment Management Plan

Introduction

The VA uses a centralized model for Salesforce deployment and release management activities. Responsibility for developing and maintaining deployment management processes lies within the VA Salesforce DTC. Deployment of solutions to the systems integration testing, user acceptance testing, and production environments are executed by the VA Salesforce Administration team.

Project Teams perform a key role in the deployment processes. They ensure that deployment tasks are documented and tested, prerequisite tasks are completed successfully, and they work collaboratively with the VA Salesforce DTC and the Salesforce Administration team to help ensure successful deployments.

This document is organized into the following sections to provide teams with a reference to understand the deployment process and the activities each organization is responsible for performing:

- **VA Salesforce Environment Management:** Provides an overview of the Salesforce environments in use at the VA and the environment management tasks the VA System Administration Team performs.
- **Deployment Process:** Describes the tasks performed by project teams and the VA Salesforce System Administrators to deploy modules on the VA Salesforce Platform.
- **Deployment Technologies:** Describes the technologies used by the VA Salesforce DTC to perform deployments.
- **Version Control / Source Code Management:** Describes current version control management process used by the VA Salesforce Platform and provides an overview of a future state version control management system used to maintain Salesforce code and configuration metadata.
- **Deployment Checklist Template:** Template (found in the appendix) for the deployment checklist that will be created and maintained by VA Project Teams.

VA Salesforce Environment Management

To understand the deployment process, teams must be aware of the various Salesforce environments at the VA and how they are managed. The table below provides an overview of each type environment and the supporting sandbox:



Table 1: Different Environments Management

Environment Type	Purpose	Sandbox Type	Deployments Performed By	Comments
Production	Provides business capabilities to production users	N/A	VA Salesforce Administration Team	Currently there are three production Salesforce organizations operating in the VA environment. Administrator access to production environments is strictly controlled. (Note, efforts are underway to consolidate VA orgs)
Staging	Training and Final pre-prod Test Environment	Full Sandbox	VA Salesforce Administration Team	The Staging sandbox is treated similarly to the Production environment. Accounts with elevated privileges are held only by the VA Salesforce Administration team. Note: this sandbox contains production data and production security model applies
User Acceptance Testing (UAT)	User Acceptance Test, and Staging	Partial Copy Sandbox	VA Salesforce Administration Team	The UAT sandbox is treated similarly to the Production environment. Accounts with elevated privileges are held only by the VA Salesforce Administration team.



VA

U.S. Department
of Veterans Affairs

Environment Type	Purpose	Sandbox Type	Deployments Performed By	Comments
				Note: this sandbox contains production data and production security model applies
System Integration Test (SIT) – Integrated/ Consolidated	Testing of code integrated from multiple project teams	Partial Data / Developer Pro	VA Salesforce Administration Team	Integrated/ Consolidated SIT sandboxes are available to integrate test code created by multiple project teams.
System Integration Test (SIT) – Project Team Owned	System testing of code from a single project team	Developer	Project Teams	SIT sandboxes are used by project teams to conduct system test and to perform the mock deployments necessary to develop and confirm their deployment checklists.
Development	Development and Unit Test	Developer	VA Salesforce Administration Team*	The VA Salesforce Administration team is responsible for development sandbox creation and refresh. Development sandboxes are not typically targets for migration.



The VA Salesforce administration team is responsible for the management of all Sandboxes, including:

- **Sandbox Creation:** To request a new Sandbox, project teams open an issue with the VA Salesforce Help Desk. The link is public facing so you do not need access to Salesforce production organization, however the link is found on the home page of the VA Salesforce org. The VA Salesforce Administration team determines if the Sandbox supports an approved project, validates that an unused sandbox is available, and creates sandboxes for requests approved by the DTC. **NOTE:** No sandboxes are distributed unless there is an Application Enhancement request in progress. The business partner from the VA should create the Application Enhancement Request record in VA's Salesforce production environment to initiate this process.
- **Sandbox Refresh:** To request a Sandbox Refresh, project teams open an issue with the VA Salesforce Help Desk. The VA Salesforce administration team determines if the Sandbox is eligible for refresh and confirms that the current use of the sandbox allows for a refresh to occur.
- **Sandbox Monitoring & Deletion:** The VA Salesforce Administration team monitors sandboxes for inactivity. If a Sandbox has not been used for three months and/or it is not tied to an active App Enhancement Request object, the administration team contacts the project and determines if the sandbox is a candidate for deletion.

Deployment Process

This section describes how code and configuration are deployed in the VA Salesforce environment. It includes two diagrams. Figure 1 illustrates how code progresses from the development and testing environments to production. Figure 2 illustrates the process that is followed to execute deployments.

Project Teams are responsible for designing and configuring their modules within their respective team development sandbox. Project teams migrate code to their Team SIT sandbox as needed. When testing is complete, the VA Salesforce Administration team then migrates the code to a consolidated SIT, UAT, and Staging environment; then the final migration to production. Figure 1 highlights this flow. The arrows in the figures below represent the sources and targets of the change sets used in this migration.

To request migration between environments, you will need to submit a case using the [Open an Issue with the VA Help Desk](#) link. Cases will be reviewed on a first come first served basis and time to deployment will be dependent upon capacity. Should you have an extenuating circumstance you would like to have reviewed, please attend a DTC integrator call, held at 9:45



am, daily. No deployments will occur without a submitted case, no cases will be approved without proper documentation and customer sign offs in the enhancement request.

For additional information on deadlines for customer sign offs in enhancement requests, you can reference the [2018 release calendar](#), should you have questions on this, please attend the Monday CCB call, hosted weekly at 2 pm by Bea Zimmermann. Staying in accordance with the listed deadlines will prevent deployment delays.

NOTE: In FY2018 this process will be moving out of change sets and into a GitHub/Jenkins solution. Please make sure you attend the Friday Developer CCB calls to stay abreast of the progress.

Change sets are the mechanism by which configuration changes are deployed at the VA. This document will be updated over time as the deployment mechanism matures.

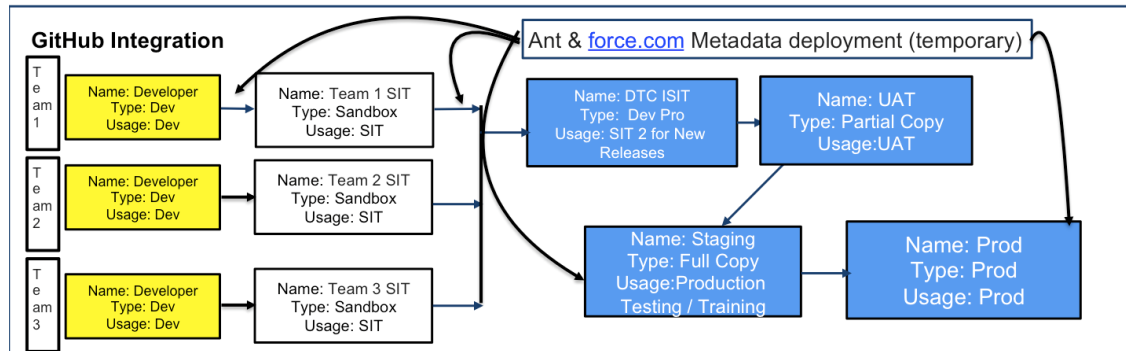


Figure 1: Migration Flow

Note UAT is now a Partial-copy
and Staging is the Full Copy



Figure 2 graphically depicts how deployments are conducted on the VA Salesforce Platform. Deployment checklists created by the project team are executed by the VA Salesforce Administration team to migrate code to the consolidated SIT, UAT, and production organizations.

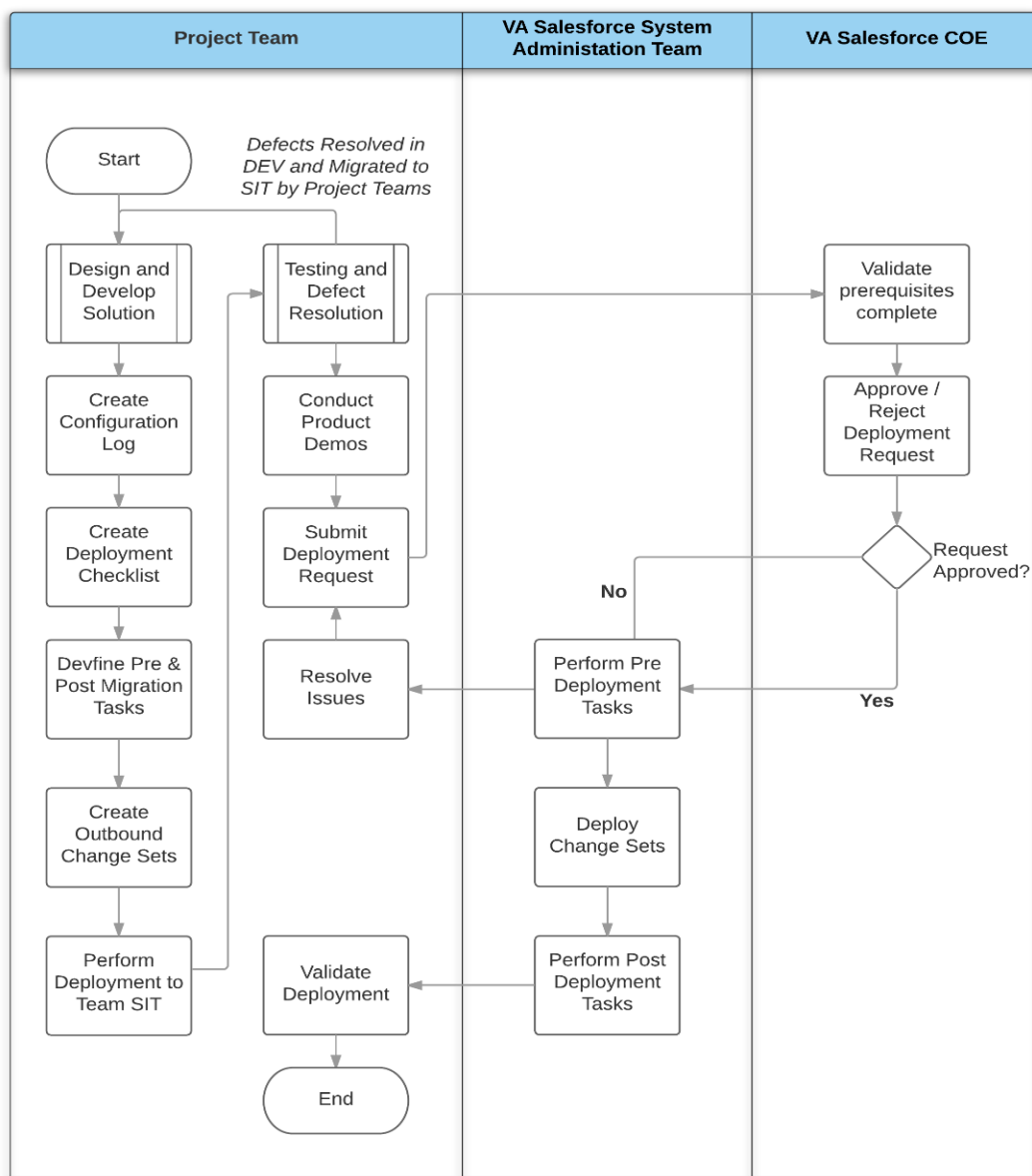


Figure 2: Deployment Process



The tasks involved in the deployment process are described in detail below. Please note that not all tasks in this table are performed for all project deployments. Additionally, rows with a gray background are not deployment tasks, but are listed to provide the context of the deployment task within the overall development process.

Table 2: Deployment Tasks

Task	Performed By	Comments
Create Development Sandbox	VA Salesforce Administration Team	When a project is approved, the VA Salesforce Administration team creates a sandbox and provides access to project team. <i>An approved Application Enhancement Request records must be created by the business owner prior to this step.</i>
Design and develop solution	Project Team	The project team configures Salesforce to meet requirements as detailed in the approved user stories
Create Exception Requests and Obtain Approval	Project Team	Some customizations will require an Exception request to be created and approved by the DTC team. This includes but is not limited to: New fields on a Standard object, New or Update to Profiles, New/Update to Permission set, Change to Community, Any Apex or VisualForce code, AppExchange application. <i>You must attend the Friday CCB call to provide justification for your request to be considered. Requests submitted in a given week will be reviewed during that Friday's call.</i>
Create and Obtain Approval on Architecture Document	Project Team	All new modules and enhancements to existing modules must create an Architecture document (based on VA's approved template) and obtain approval from the DTC team. This should take place



Task	Performed By	Comments
		before code is completion ant it is recommended to create this as soon as possible.
Create and maintain a log of configured items	Project Team	Project teams maintain a configuration log listing all Salesforce components created or updated during development
Create and maintain Deployment Checklist	Project Team	Lists the manual steps performed as part of deployment and the change sets involved in the deployment
Deploy to Team SIT Sandbox	Project Team	Project teams follow the deployment checklist and use change sets to deploy their configuration to their SIT sandbox. During this deployment, the deployment checklist is validated and refined.
Perform System Integration Tests	Project Team	Testing performed in team SIT Sandbox
Remediate defects and deploy to SIT	Project Team	The project team remediates defects in DEV sandbox and migrates to SIT for each successive test cycle. Best practice is for project teams to continuously deploy to their own SIT environment and use the SIT environment for product demonstrations.
Deploy to consolidated SIT environment	VA Salesforce Administration Team	The VA Salesforce Administration team uses the deployment checklist to deploy to the consolidated SIT environment



VA

U.S. Department
of Veterans Affairs

Task	Performed By	Comments
Conduct Integration Testing	Project Team	Testing performed in consolidated SIT sandbox
Remediate defects and deploy	Project Team / VA Salesforce Administration Team	Defects are remediated in the DEV sandbox and migrated to the consolidated SIT environment by the VA Salesforce Administration Team
Validate UAT Requirements met	VA Salesforce DTC	Verifies that code reviews have been complete, SIT complete, and code coverage standards have been met
Conduct UAT Migration	VA Salesforce Team	The VA Salesforce Administration team uses the deployment checklists to migrate the configuration to the UAT environment
Conduct User Acceptance Testing	Project Team	UAT performed full copy sandbox
Remediate defects and deploy to UAT	Project Team / VA Salesforce Administration Team	Defects are remediated in the DEV sandbox and migrated to the UAT environment by the VA Salesforce Administration Team
Verify Production Prerequisites	VA Salesforce Administration Team	Verifies completion of UAT and the Product Owner acceptance of the module prior to production deployment



Task	Performed By	Comments
Conduct Production Migration	VA Salesforce Administration Team	The VA Salesforce Administration team uses the deployment checklists to conduct the Production deployment

Deployment Technologies

Deployments on the VA Salesforce Platform are currently conducted using the following:

Change Sets: Change sets are a native Salesforce technology used for migrating Salesforce components (metadata) between related sandboxes. This is the process currently in use at VA. The process to deploy Salesforce components using change sets is illustrated in the figure below:

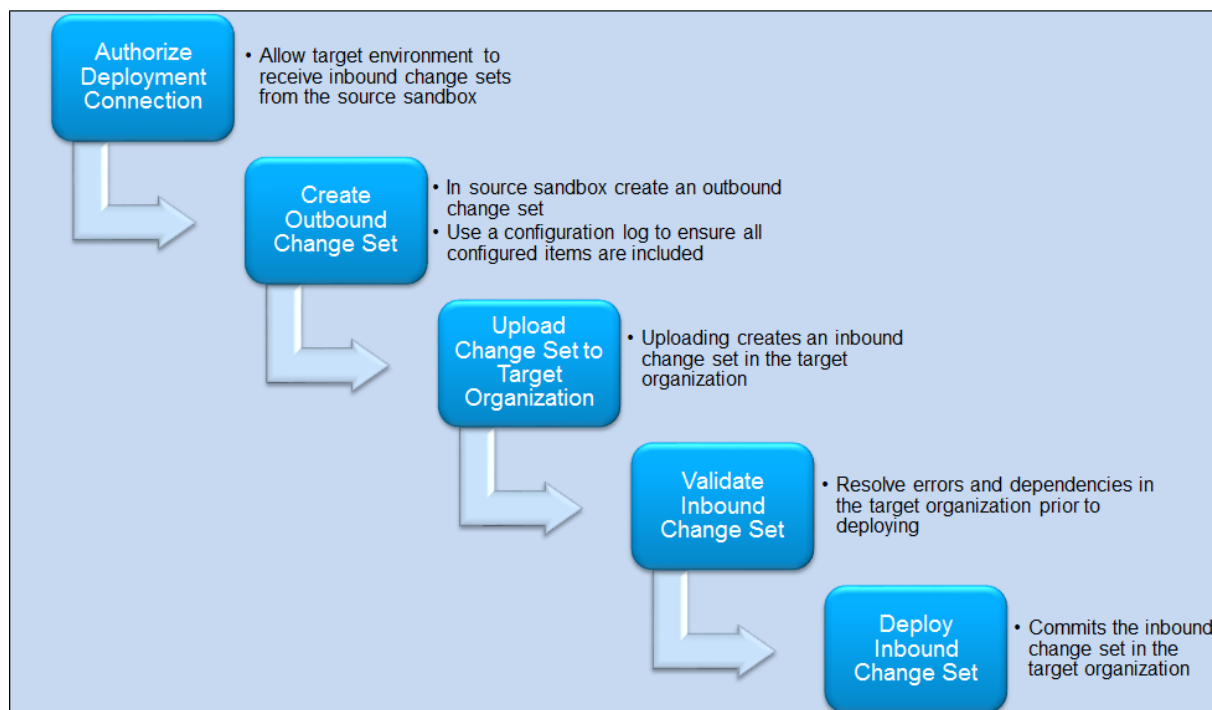


Figure 3: Change Sets

Change sets allow fine-grain control over the items that Salesforce will deploy. Project teams must ensure they have included all configuration items developed or updated in their release



when defining the change set. A configuration log is used for managing the creation of a change set.

Change sets must be recreated manually in each source environment. This is a manual and tedious process. The configuration log is a critical input that project teams must provide to the VA Salesforce Administration to allow them to duplicate change sets in higher environments (SIT - Full, UAT).

Configuration Log: The configuration log is a complete list of all the Salesforce components created or updated in a release. Configuration logs can be maintained in spreadsheets. Project teams must ensure their configuration log is up to date throughout the development lifecycle.

Deployment Checklist: The deployment checklist defines all the activities needed to perform a deployment. It includes the following:

Pre-deployment tasks: An ordered list of activities that must be manually performed prior to deployment

- **Change Sets:** An ordered list of the change sets to be applied during this deployment
- **Post-deployment tasks:** An ordered list of activities that must be manually performed after deployment
- **Verification Steps:** A list of tasks to be executed that confirm the deployment was successful.
- **Roll-back Procedures:** If the deployment of a change set fails, Salesforce will automatically roll-back the metadata changes. However, pre-deployment tasks (documented above) must be manually backed out. The roll-back procedure describes how to back out these tasks. Note, once a change set successfully deploys, the change set cannot be rolled back.

Version control / Source Code Management

Current Source Code Management Process

The VA Salesforce Platform currently uses developer sandboxes to support configuration management. Weekly and prior to the deployment of significant changes to production, a configuration-only sandbox is refreshed with the latest production metadata. This process allows a 7-week rolling history of metadata to be maintained.

Future-State Source Code Management Processes

Reliance on manual deployments via change sets and use of sandboxes to maintain software versions is a good initial solution for the VA Salesforce platform. However, as the use of the



platform grows and projects increase in complexity, the limitations of the tools and processes will become more apparent. Some of the limitations of the existing process include:

- Requires a labor-intensive (manual) process to develop and maintain change sets.
- Deployments that require objects to be deleted or renamed require manual steps. (Change sets cannot delete or rename components).
- Conflict identification and resolution is more difficult without a source code management solution.
- Traditional development techniques including branching and merging code lines is more difficult without a source code management system.

A phased approach, as illustrated below, should be used to incorporate new technologies and processes that address the above limitations.

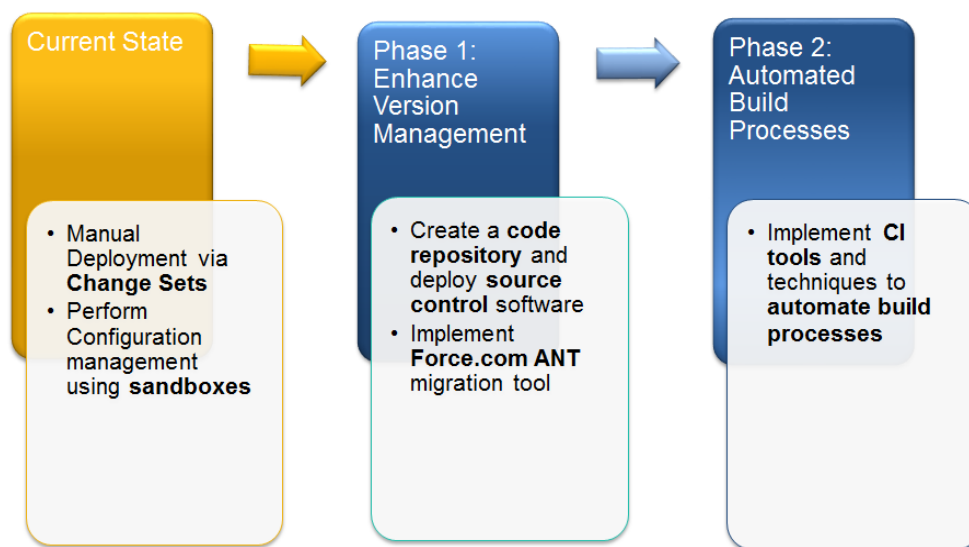


Figure 4: Phased Approach

The first phase in this process is to transition away from using sandboxes for configuration management through the implementation of a code repository and source code management software. This has the added benefit of allowing deployments to be scripted using the Force.com ANT migration tool, reducing the manual effort associated with creating change sets.

It is recommended that a source code management tool should be used because of the following benefits:

- Providing the ability to track changes and easily reverse changes in code as needed



VA | U.S. Department of Veterans Affairs

- Allows multiple developers to work on their own copy of a resource
- Allows conflicts in resources to be identified and managed by developers
- Allows multiple branches from the main code line to be worked simultaneously and merged at various points during the development lifecycle

Below is a notional diagram of how a source code management system can be integrated into the VA Salesforce environment. In this approach, Salesforce project team developers configure their solution in development sandboxes, however they also use the Salesforce Force.com IDE to check their changes into the VA source code repository. Deployment is performed scripted using the Force.com Migration tool.

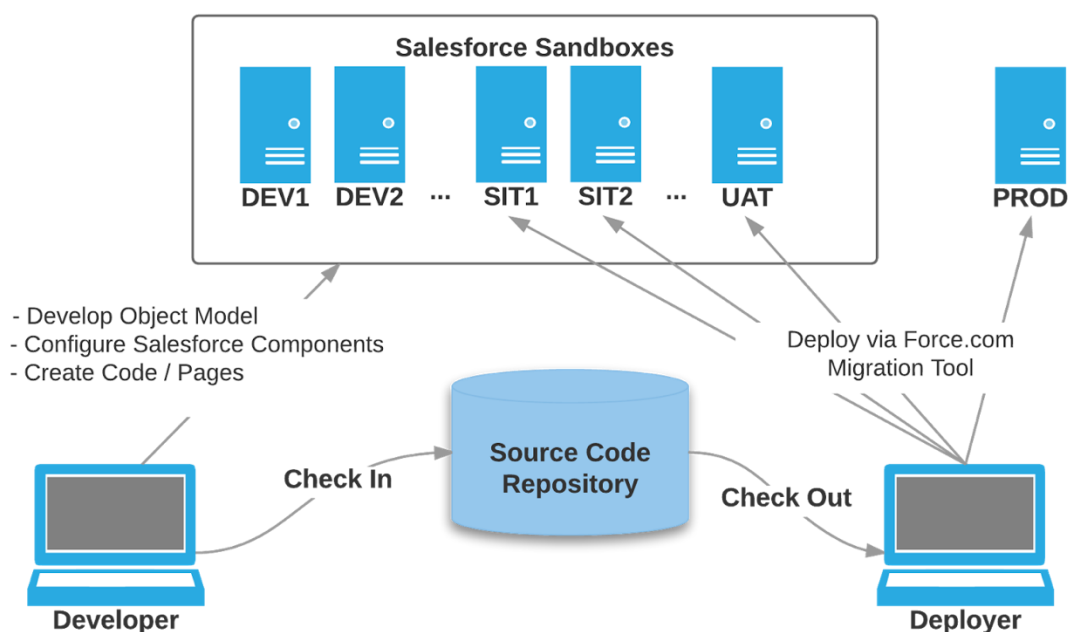


Figure 5: Notional diagram

The Force.com Migration tool is a command line utility that moves metadata between a local file system and a Salesforce organization. Instead of change sets, components for deployment are identified using an XML manifest (package.xml). The scripted retrieval and deployment of components allows for repetitive deployment management processes to be executed more efficiently.



As the VA Salesforce deployment management process matures, the VA build upon the platform established in the first phase by replacing the Force.com Migration Tool with a build automation tool like Jenkins. Jenkins can automatically monitor a Sandbox for metadata changes and check code into the source code repository. Deployments to other sandboxes and Production can be similarly automated.



Coding Standards and Development Guidelines

The VA Salesforce platform is a shared resource that enables projects across VA to quickly develop robust, flexible and scalable modules in a secure cloud environment. The VA DTC provides structure and guidance necessary to assist project teams maximize VA's investment in the platform, with the objective of helping project teams deliver quality solutions. Specifically, the coding standards and development guidelines set forth in this document are intended to:

- **Prevent Conflicts:** Standard objects and other Salesforce components are used for different purposes in the various VA Salesforce modules. A key goal of these standards is to help project teams design and implement their changes in a manner so that they do not conflict or negatively impact other projects.
- **Enhance the Supportability of the Platform:** Adherence to design, configuration, and coding standards helps new projects understand and leverage what has been previously developed and enable operations and maintenance (O&M) teams to better support existing modules.
- **Promote Best Practices:** The Salesforce platform is constantly evolving with new features. The guidance set forth in this document will evolve with the platform to ensure consistency with leading practices and configuration techniques.
- **Help Organization Stay within Governor Limits:** Salesforce is a multi-tenant platform and enforces limits to ensure processes do not monopolize shared resources. The reviews and guidelines described in this document help project teams ensure their modules stay within these limits.

Please note, these guidelines are not meant to restrict developer creativity or provide an undue overhead on VA Salesforce projects. Instead, the intent is to enable project teams to quickly get up to speed with Salesforce best practices, avoid common coding mistakes, and accelerate the module development process with a "configure first" approach that leverages out-of-the-box functionality as much as possible, and to maximize the return on investment for the VA Salesforce platform.

No Code Technical Approach

Regardless of your role, you'll need to be able to make wise decisions when it comes to delivering functionality for VA Salesforce Organization. Inevitably, you will encounter scenarios in which a solution can be delivered either with or without the use of code.

There are unique pros and cons in every situation, but do consider a few key points that may support the decision to avoid using code:



VA | U.S. Department of Veterans Affairs

- The need to consider Salesforce.com limits and parameters is significantly reduced or, at times, eliminated completely when building solutions using declarative means.
- Modifications are often more straightforward, as they may only require a change to a configuration setting, not to a line of code.
- No code coverage for testing is required to deploy (Apex test classes, in contrast, must be written for custom Apex Code)
- Knowledge transfer burdens are reduced, since an understanding of the particular feature or function is typically sufficient to quickly determine what a specific module is intended to do
- Future maintenance is simplified, if for example, an individual who built custom modules for you leaves abruptly, picking up the pieces is much simpler if the work was done declaratively.

There are numerous scenarios that may not be able to achieve business requirement with declarative method, which need to do development with code. However, developing with code in Salesforce.com is often done based on need; Many times, want to extend the platform to support functionality that simply does not exist "Out of the box". Make sure you have a solid justification for developing with code if your business requirement cannot be met through configuration.

Readability Guidelines

It is important that teams using the VA Salesforce Platform understand how other teams are using the platform to prevent conflicts, increase supportability, and promote reuse. The guidelines in this section describe naming conventions and standards for use of comments and descriptive text to improve the readability of components configured on the platform.

Naming Conventions for Salesforce Components

The naming convention for Salesforce components along with examples are provided below. The following Salesforce Components are subject to this convention:

Profiles	Permission Sets	Queues
Page Layouts	Apps	Workflow Rules
Approval Processes	Visual Workflow Flows	Process Builder Processes
Tasks	Email Alerts	Outbound Messages
Custom Objects	Apex Classes	Apex Triggers
Visualforce Pages	Visualforce Components	

Component Name / Label: New Salesforce components are named in accordance with the following convention:



[Organization Acronym] - [Module Acronym] - [Meaningful Component Name]:

Example: The custom object name, VBA-CS-Schools indicates the organization this object supports is the Veterans Benefits Administration, it was created to support the Compliance Schedule (Compliance & Liaison) module, and it contains school data.

Note, labels for custom fields are displayed to users of the module and must be user friendly. Field labels are not subject to this requirement.

API Names: API names use Salesforce default naming convention consistently. In this convention API names mirror the object or component name; all words have the first letter capitalized and spaces are replaced by the underscore character.

Example: The API name for the VBA-CS-School object is VBA_CS_Schools__c

Use of Descriptive Text

The Description attribute must always be populated. The description provides a brief but concise description of how the component is used.

Example: The description for the VBA-SSD-GSA Trip Tracking object: “Track all the trips made in GSA Vehicles. Tracks the miles driven and for what purpose.”

Guidelines for Use of Common or Standard Objects

VA Salesforce project teams should consider using standard Salesforce objects wherever practical to take advantage of Salesforce out-of-the-box functionality. This is especially true when there is close alignment between project requirements and the standard Salesforce use case for the object. However, it is critical that project teams research the existing object, identify if and how other teams are using the existing object, and take precautions to ensure that any modifications to the object do not negatively impact current functionality.

The guidelines described below describe tasks that project teams must complete when using standard objects to prevent conflicts with other modules that are currently using the object, or that may do so in the future.

Guideline	Rational
Consider using a new record type, and/or other mechanisms to uniquely identify your project’s records	Allows your project’s data to be distinguished from other data stored in the common object. Note: this is important even if your project is the only module using the object, as other teams may have use for the object in the future.



Guideline	Rational
Review existing sharing settings for the object	Organization-wide defaults and other sharing settings already established may impact how your project utilizes the common object
Review object permissions for existing profiles and permission sets	Provides insight into the how the common object is currently utilized, the types of users that have access to the common object, and any restrictions to field visibility that may be established.
Review existing Workflow Rules, Visual Flows, Process Builder Flows, Assignment Rules, Escalation Rules, and Support / Sales processes for potential conflicts	Helps ensure that updates to the project's data does not trigger another project's workflows unintentionally. Identifies changes to existing components that may be required to exclude data from the project.
Review existing Apex Classes and Triggers	Helps ensure that updates to the project's data do not trigger another project's code unintentionally with unexpected results. Helps identify changes to existing code that may be required to exclude data from the project.
Review existing Reports and Dashboards using the object	Ensure that new data is not inappropriately included on existing reports and dashboards
Review existing List Views	Ensure that new data is appropriate on existing list views

Regression Testing and Technical Debt

Regression Testing: It is important that **ALL** module functionality supported by the object is completely tested prior to release. This includes functional testing of the new capabilities supported by the object and complete regression testing of pre-existing features previously supported by the object. Regression testing is necessary to ensure that changes do not negatively impact existing functionality.

Ideally, regression tests should be performed by the project team that originally developed and tested the capability. If this is not possible, the DTC, product owners, and/or other persons familiar with the original functionality should be engaged to ensure that each affected component is adequately regression tested prior to release.



Technical Debt: It is likely that existing components will need to be refactored when objects are reused. The project team will provide the DTC a list of components that require refactoring for the reused object. Refactoring occurs prior to regression testing and ideally should be performed by the project team that developed the original capability.

Coding Standards

The VA Salesforce DTC considers Apex Classes, Apex Triggers, Visualforce Components, Visualforce Pages, and other components created using a programming language (Apex, Visualforce, HTML, Javascript, etc.) to be “code”, as opposed to declarative development.

- An exception to VA Salesforce Platform DTC standards is required to use code. The exception process is described in the “Code Reviews and Exception Requests” section of this document.
- If declarative techniques can address requirements, use of code on the VA Salesforce Platform is to be avoided.
- The VA Salesforce DTC will work with project teams during design to identify declarative alternatives to code.
- If code is the best or only option to meet a requirement, an exception to use code may be granted by the VA Salesforce Design/Development Configuration Control Board (CCB) as described in the Development Process, Code Reviews and Exception Requests section of this document.
- The standards and guidelines described in this section must be followed if an exception to use code is approved.

Code Structure and Comments

Apex code structure and comments should focus on readability. Classes and methods use standard and meaningful naming conventions. Comments are used to explain the purpose of classes, methods, and code.

Naming Conventions: Apex classes use a similar convention to those described in the Readability Guidelines above, however CamelCase notation is used instead of underscores.

Type	Convention	Rationale /Example
Apex Controllers	[Organization Acronym][Module Acronym][Meaningful Class Name][Controller]	Appending Controller to the end of name indicates that the code is an Apex Controller Example: VBACSSchoolsController



U.S. Department
of Veterans Affairs

Test Classes	[TestedClassName][Test]	Appending Test to the end of the name indicates the code is a Test class Example: VBACSSchoolsControllerTest
Triggers	[ObjectName][Trigger]	Prepending the object name in front of the trigger clearly indicates the object on which the trigger is acting Example: VBACSSchoolsTrigger
Methods, Variables	Meaningful name in camelCase with initial letter lower case	Using camelCase provides a visual distinction from the API names of custom objects and fields (which use the standard Salesforce notation e.g. VBA_CS_School__c)

Comments: Classes and methods contain an information block that briefly describes its purpose, creation date, and a change log. Within methods, more complex sections contain brief comments that describe the code.

Information Block Format:

```

/*****
*****
* Name: [Class Name]
* Description: [Brief description of the class and what it does]
* Change Log:
* -----
-----
* Change ID          | Date          | Author          |
Description
* -----
-----
* 0001               | [Date]       | [Developer Name] | [Brief
description of change]
* -----
-----
*****
*****/

```

Guidelines for Test Classes

The table below describes guidelines for creation of test classes and methods on the VA Salesforce Platform.



Guideline	Rational
Design and develop test classes and methods at the same time the Apex code to be tested is written	Creating test classes and methods simultaneously with development of the tested code helps to ensure effective testing of expected conditions, helps enforce inclusion of error handling within Apex code, and helps prevent the creation of last minute classes solely for the purpose of meeting code coverage requirements.
Create test methods that simulate business functionality, use cases, and error conditions	Test methods should not be created solely for the purpose of meeting code coverage requirements. They should be designed to test the use case that is addressed by the Apex code, as well as to validate error handling code.
Create test data within test classes and methods	Developers should not expect that data within the Salesforce organization will be available for testing purposes. Also, they should not anticipate that organization data will have the necessary values to test all conditions and code branches. All test data should be created with test classes to ensure expected results coincide with conditions in test data.
Use assert statements logically and correctly	Assert statements are used to validate that code works as intended. Without assert statements, code is not truly tested. Meaningless assert statements such as <code>system.assert(true=true)</code> should not be used.
Don't focus solely on code coverage	Project teams keep the Salesforce 85% code coverage requirement in mind; however, they also focus on ensuring all use cases and possible conditions are tested.
Do not use <code>@isTest(SeeAllData=true)</code>	This could allow someone to remove or modify data that they should not have access to.

Apex Guidelines

Salesforce provides a number of excellent guidelines for developing Apex code. The intent of this section is not to rewrite existing guidance, but to provide VA Salesforce project teams with links to existing guidance and best practices and highlight areas where the Salesforce DTC will focus during design and code reviews.

Salesforce Apex Resources



See the links below for Salesforce guidance and best practices for the use of Apex:

- [Apex Best Practices](#): Describes core Apex best practices for writing efficient, scalable code
- [15 Apex Commandments](#): 15 additional guidelines for efficient Apex code
- [Secure Coding Guidelines](#): Describes common security issues Salesforce has identified while auditing modules built on or integrated with Force.com
- [Apex Developer's Guide](#): Provides a comprehensive guide to Apex code and syntax

Focus Areas

The following list highlights areas that project teams focus on during design and development of Apex code. These will be areas of focus during code reviews.

- **Performance and Governor Limits:** Loops, queries, DML statements, recursion, and other code constructs that are likely to have a negative impact on performance and cause governor limits to be exceeded should be designed carefully. Code reviews will focus on ensuring that SOQL and DML are not executed within loops and that queries are efficient and use collections where possible.
- **Bulkification:** Triggers can be invoked via batch updates that impact multiple records at a time. Project teams should ensure that triggers and trigger helper classes are written to handle multiple records once. Code reviews review triggers to ensure they handle bulk update situations appropriately.
- **Triggers:** The sequence of trigger execution cannot be controlled in Salesforce and having multiple triggers on an object reduces code manageability. Project teams should ensure that only one trigger per object is created. Additionally, triggers can be constructed so they invoke themselves. Project teams should validate their code takes appropriate measures to prevent the unwanted, recursive execution of triggers. Code reviews will ensure one trigger is created per object and check recursion.
- **Exception Handling:** Apex code should have Try-Catch-Finally statements in place to handle and recover from exceptions wherever possible. Project teams validate that all Apex code includes appropriate exception handling. Proper use of exception handling is checked during code reviews.
- **Redundant Code:** Blocks of redundant / repeated code make maintenance more difficult. Project teams identify blocks of code that are repeated and are candidates to be condensed into a single class or method.
- **Secure Coding Guidelines:** Project teams review the Secure Coding Guidelines and ensure the Apex code conforms to the described practices.



Visualforce Guidelines

Similar to Apex, Salesforce also provides guidelines and best practices for developing Visualforce. The intent of this section is not to rewrite existing guidance, but to provide VA Salesforce project teams with links to existing guidance and best practices and highlight areas where the Salesforce DTC will focus during design and code reviews.

Visualforce References

- [Visualforce Developer's Guide](#): Provides a comprehensive guide to Visualforce code and syntax.
- [Visualforce Performance Best Practices](#): Provides best practices for optimizing the performance of Visualforce pages.

Focus Areas

The following list highlights areas that project teams focus on during design and development of Visualforce code. These are the areas of focus during code reviews.

- Ensure Javascript and CSS are included as static resources to promote consistency and reuse.
- Ensure that custom Visualforce Components are created and used appropriately to promote consistency and reuse.
- Validate that Javascript is not used for functionality that can be accomplished using native Visualforce features.
- Employ techniques to optimize Visualforce performance, including:
 - Using the cache attribute with the <apex:page> component to take advantage CDN caching (when appropriate).
 - Marking controller variables as “transient” (if not needed between server calls) to enable faster page loads.
 - Using <apex:repeat> to iterate over large collections.
 - Use lazy initialization in Visualforce controllers to improve page performance.
 - Validate that the Visualforce page addresses 508 compliance. For example, ensure that all tables have summary text and graphics have ALT text set appropriately.

Development Process, Code Reviews, and Exception Requests

The *VA Salesforce Configuration and Change Management Plan* (July 2016) establishes and documents the VA Salesforce Configuration Management Team and the VA Salesforce Configuration Management Boards (CCBs). These organizations are responsible for the integrity of the Salesforce platform. It is through the CCBs that the development process is managed, code is reviewed, and exception requests are processed. Each CCB is described in more detail below:

VA Salesforce Project CCB



The VA Salesforce Project CCB is responsible for monitoring and managing existing Salesforce projects and reviewing and approving new projects and future enhancements. Project teams will interact with the VA Salesforce Project CCB as needed for issues regarding Salesforce licenses, sandbox management, and configuration management.

VA Salesforce Design/Development CCB

The VA Salesforce Design / Development CCB helps maintain the integrity of the Salesforce platform by reviewing and approving project designs, code and configurations. This CCB also provides a forum to assist with design decisions, and answer questions regarding technology and the Salesforce product. Project teams interact with the VA Salesforce Design/Development CCB weekly.

Exception Requests: The VA Salesforce Design and Development CCB is responsible for approving exceptions to allow the use of code, modify or create profiles, changes to standard objects, changes to roles, community changes, use of AppExchange apps, use of lightning components, VIEWS shared features, modify or create integrations, and modify or create permission sets. Exception requests should be submitted through the [form](#) in Salesforce. If you need assistance with submitting, please see the [How to Open an Exception Request](#) Guide. Exception requests will be reviewed during your design review and weekly thereafter on the CCB calls at 10 am on Fridays.

Design Reviews: Project teams must have their designs reviewed and approved prior to the start of development. Designs will be reviewed at the weekly CCB meetings on Mondays at 2 pm or Fridays at 10 am. Prior to your review you must complete an Architecture Document using the [template](#) found in the VA System Integrators Chatter group.

Code Reviews: Prior to deployment to the System Integration Test (SIT) or User Acceptance Testing (UAT) environments, all code must be reviewed by the CCB. Prior to deployment to production, all issues noted in code reviews must be resolved.



Appendix

Code Review Checklist

This section provides a checklist for evaluating project teams' code; items on this checklist are verified during code reviews.

Code Review Checklist

DATE: [DATE COMPLETED]

MODULE REVIEWED: [MODULE NAME]

REVIEWER: [REVIEWER NAME]

Item #	Description
1	Is all code well-structured, consistent in style, and consistently formatted (Is TABS used in formatting)?
2	Are there any uncalled/unneeded classes or triggers?
3	Are there any blocks of repeated code that can be condensed into a single class?
4	Does the code implement naming and other standards as defined in these coding standards?
5	Are there any redundant or unused variables?
6	Do all test classes and methods run/execute successfully?
7	Is the code clearly and adequately commented and is the code consistent with the comments?
8	Are all variables properly initialized?
9	Are triggers appropriately bulkified?
10	Are collections and loops used appropriately to process records?



Item #	Description
11	Are SOQL and SOSL queries avoided inside of loops?
12	Are DML statements avoided inside of loops?
13	Is there no more than one trigger per object?
14	Are unwanted trigger recursions avoided?
15	Are any record IDs hard coded?
16	Does all code include appropriate exception handling?
17	Are static queries, bind variables and escapeSingleQuotes method used to protect against SOQL and SOSL injection attacks
18	Are any classes excessively complex and should be restructured or split into multiple routines?
19	Do triggers contain complex logic that is better maintained in a separate Apex class?
20	Are controllers used to define picklist values? (instead of including them in the Visualforce pages)
21	Are controller variables or wrapper classes used to hold data referenced in Visualforce pages? (this will avoid “insufficient privilege” error if user does not have access to the data)
22	Is “With Sharing” used in Apex Classes to honor sharing settings and rules that prevent unauthorized access to data?
23	Are JavaScript and CSS included as static resources instead of on Visualforce pages?
24	Are CSS references located at the top of, and JavaScript references located at the bottom of Visualforce pages to provide faster page load?
25	Is Javascript used that can be replaced by Visualforce conventions?



VA | U.S. Department
of Veterans Affairs

Item #	Description
26	Have Visualforce pages accounted for 508 compliance? (e.g. Do all graphic components have an alt text?
27	Do test methods simulate business functionality and use cases?
28	Are assert statements used logically, and correctly?
29	Is test data created appropriately in test methods and classes?

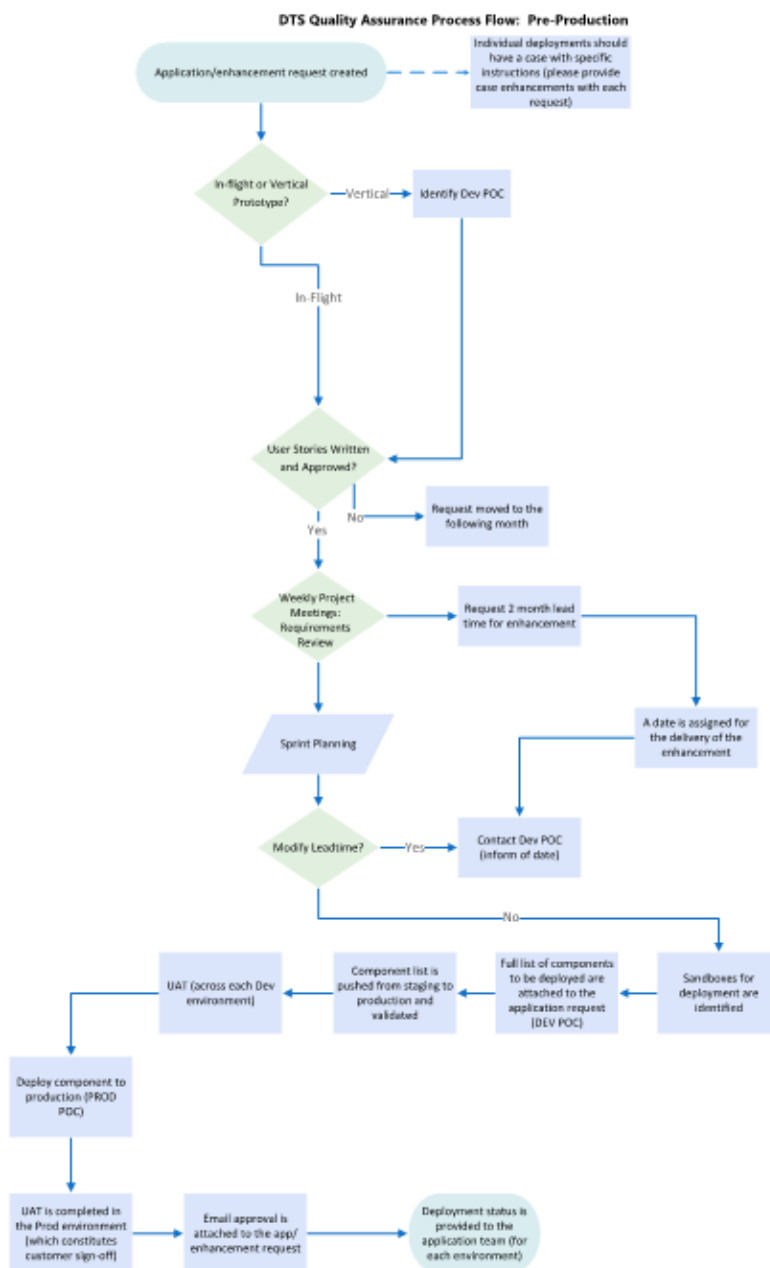


VA | U.S. Department of Veterans Affairs

Pre-Deployment

Pre-Deployment QA Process

Please see the diagram below regarding the proper pre-deployment QA process. Please utilize this as a guideline in moving through your development cycle.





Pre-Deployment Checklist

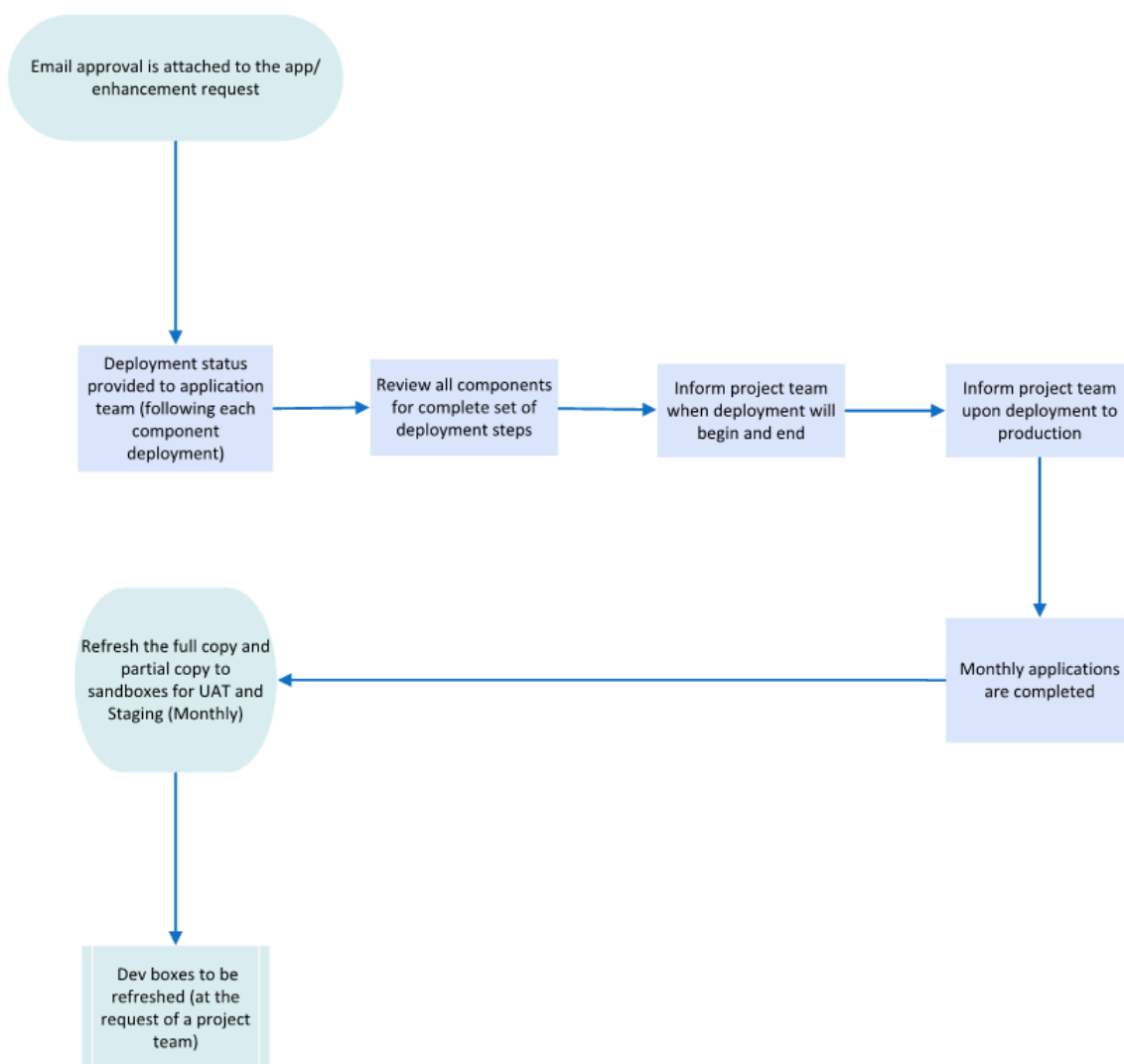
In addition to the guidelines you have received above, we have a standardized DTC pre-deployment checklist that we would like you to ensure has been completed prior to each deployment, please find the list below.

DTC Deployment Checklist	Yes/No	If Yes please elaborate
Has a Application/Enhancement request been submitted?		
Is the Application/Enhancement approved?		
Has a business sponsor been identified?		
Have user stories been written and approved?		
Has a requestor submitted a minimum 60 days notice (application/enhancement completion date)?		
Has an exception request been submitted?		
Have development sandboxes identified?		
Does the enhancement require custom code?		
If yes, has a code review been scheduled with the DTC team?		
Does the enhancement require a Security review?		
Does the request alter the VAs current Security settings (Org Wide Defaults)		
Have Sandboxes been connected (via deployment settings)?		
Have test scripts been written?		
Has a deployment point of contact been identified?		
Will data need to be loaded into Production? (if YES, list the Objects in column C)		
Have specific deployment instructions been prepared for each case across each Environment (and Production)?		
Will the enhancement require a Custom Profile?		
Will the enhancement require the creation and assignment of specific permission sets?		
Has DTC support been trained and signed off on how to administer the new functionality?		
Has the Business Point of Contact Signed off testing in DEV/UAT/Staging?		
Has the Business Point of Contact signed off in Production?		
Were all issues/errors resolved successfully during the deployment to DEV/UAT/Staging?		
Does the Customer support intake form capture enough information to assist with first touch resolution?		



Please see the diagram below regarding the proper post-deployment QA process. Please utilize this as a guideline following your development cycle.

DTS Quality Assurance Process Flow: Post-Production





Post-Deployment Checklist

In addition to the guidelines you have received above, we have a standardized DTC post-deployment checklist that we would like you to ensure has been completed following each deployment, please find the list below.

DTC Deployment Checklist	Yes/No	If Yes please elaborate
Will data need to be loaded into Production? (if YES, list the Objects in column C)		
Will the enhancement require a Custom Profile?		
Will the enhancement require the creation and assignment of specific permission sets?		
Has DTC support been trained and signed off on how to administer the new functionality?		
Has the Business Point of Contact signed off in Production?		