Many organizations embrace the IT Service Management Forum (itSMF) IT Infrastructure Library (ITIL) v3 framework, which provides guidance on the processes and functions that help implement quality IT services. However, organizations struggle with how to implement ITIL processes such as change management, service asset and configuration management, and release and deployment management. This article shares practical guidance on how to use DevOps practices to implement the processes and functions described in the ITIL v3 Core Transition from Service Design to Service Operation.

View more content in this series

Many organizations, from large international banks to Internet startups, focus on how to ensure the reliability and security of both their critical infrastructure and their core application base. To stay competitive, these organizations need to minimize the risk of service interruptions when they upgrade systems or introduce new features. These interruptions can affect the business and the reputation of the firm. The IT Service Management Forum (itSMF) IT Infrastructure Library (ITIL) v3 framework provides guidance on the process and functions required to deliver quality IT services that are aligned with the business goals and objectives.

Although the guidance in ITIL v3 is highly respected, many organizations struggle with how to implement these preferred industry practices. DevOps provides the exact procedures that are necessary to implement ITIL in a practical and realistic way. This article provides practical guidance on how to use DevOps practices to implement the processes and functions described in the ITIL v3 framework to transition a service from design to operation.

Overview of the configuration management database

One of the most exciting practices described in the ITIL framework is the establishment of the configuration management database (CMDB), which you can use to verify that the code components (known as configuration items) on the production server are the right version and to ensure that there have been no unauthorized changes due to human error or malicious intent.

ITIL describes the use of the CMDB to track changes to the configuration items and to update the configuration management system (CMS). The CMS is an essential part of any effective change
management system and relies upon the CMDB for accurate information. If you implement these practices successfully, your organization can enjoy the benefits of agility, and can deliver value to the business through new features. You can also manage and reduce risk associated with updates to any complex system.

The CMDB and the CMS are part of the larger service knowledge management system (SKMS). These systems provide a wide range of information that supports the entire IT service management process. This information is extremely valuable but it must be kept current and accurate.

The ITIL framework suggests that changes to configuration items and their interfaces need to be tracked but does not provide specific guidance on how to track changes at a technical level. Fortunately, many DevOps practices explain exactly how to track changes to the application build, package, and deployment. When implemented, these DevOps practices help maintain current and accurate information in the CMS and CMDB.

The ability to track code baselines is required to use ITIL effectively. The service asset and configuration management process covers this requirement.

**Introduction to service asset and configuration management process**

The service asset and configuration management (SACM) process is intended to track and record application code baselines. Many organizations have struggled to implement CMDBs and to keep them updated with accurate and useful information as described in the SACM process. ITIL suggests that you track the changes to configuration items. However, it is unrealistic to expect to manually update the CMS and the CMDB. Manual updates are likely to result in less than accurate information.

A better solution is to build and deploy releases that the CMDB can discover automatically. You need to engineer your applications to be discoverable so that the CMS and CMDB systems can be programmatically updated. In addition, build verifiable baselines that can be programmatically discovered by SACM processes and implement configuration management systems that can be kept up to date with correct and validated information with automated procedures. This series of articles explains how to use DevOps to implement the procedures that are described in ITIL release control and validation.

**Overview of the ITIL release control and validation guidance**

The ITIL v3 release control and validation framework describes service management processes. It begins with the need to establish an effective service strategy that leads to a detailed service design, which is described in the service design package. The ITIL v3 framework describes the following release control and validation processes that transition new services from design into operation:

- Change management
• Service asset and configuration management
• Release and deployment management
• Service testing and validation
• Change evaluation
• Request fulfillment
• Knowledge management

This article describes the value that these processes provide and then explains how to implement them using DevOps.

**Change management**

Change management processes evaluate, authorize, and implement changes to services that typically occur when you implement new components or change existing components. Change management processes are designed to:

• Evaluate potential downstream effects from a change
• Reduce risk
• Improve communication to all stakeholders

The Change Advisory Board (CAB) is the governing body that reviews each proposed change. Change management drives the entire release control and validation process. Effective change management balances risk with the need to implement new changes that deliver value to a business and make it competitive. Change management relies on workflow automation and accurate, up-to-date information on assets and configuration management baselines.

**Service asset and configuration management**

Service asset and configuration management (SACM) processes manage the software assets and maintain accurate information about configuration items. These processes create and manage baselines that track the status of changes to each configuration item.

SACM processes provide all of the other processes and functions, such as the CMDB and the CMS, with accurate and up-to-date information on the status of configuration baselines.

Each software asset needs an owner who is responsible for the asset and can identify the subject matter experts who can accurately assess and evaluate the potential downstream effect of a change.

It is important to understand the interfaces between configuration items. SACM tracks changes to baselines. This tracking ability is essential for traceability. For many organizations, the tracking is required by compliance with federal regulatory and audit controls. With up-to-date information, it is much easier to operate the business, to ensure agility to implement desired changes, to conduct change planning, and to respond to incidents when they do occur. In many ways, SACM is the
"glue" that holds together the other release control and validation functions. The activities in SACM include:

- Management and planning
- Configuration identification
- Configuration change control
- Status accounting
- Verification and audit

The software configuration management plan (SCMP) provides the strategy for how to handle all of the activities required for successful application build, package, and deployment phases of code development. The planning section can simply specify the schedule for the release iterations, or it can specify every aspect of the release and deployment process. The SCMP traditionally consists of four classic functions: configuration identification, status accounting, configuration change control, and configuration audit. Specify the following information for each of these four functions:

- **Configuration identification**: Specifies a naming convention for each configuration item and helps ensure that you select the correct configuration items for each release.
- **Configuration change control**: Includes specific procedures to manage changes to configuration items, to manage new releases, and to retire configuration items.
- **Status accounting**: Documents the path of a configuration item from its initial creation to end-of-life. Status accounting includes the status of configuration baselines and configuration items as they are developed.
- **Verification and audit**: Helps to ensure that the correct configuration items are deployed. The audit information can be independently verified. Organizations depend on well-defined processes to manage release and deployment.

**Release and deployment management**

Release and deployment management (RDM) processes focus on the activities required to build, package, deploy, and test the new services. RDM creates detailed plans that help to ensure that configuration items can be successfully built and tested. The plans emphasize automated procedures that are repeatable and verifiable. RDM makes it easier for stakeholders to understand what is being done and increases the likelihood that they are satisfied with the results as the service transitions from design to operations.

RDM prepares the build, including automated procedures, and helps ensure that testing is conducted and that the test environments are coordinated. RDM processes include tasks related to the initial pilot and early life support of the service that is being transitioned into operation. After the RDM verifies these steps, it reviews and closes the transition effort.

Service validation and testing is also a key aspect when you transition a service from design to operation.

**Service validation and testing**

Service validation and testing helps to ensure the IT service is:
• **Fit for purpose:** It matches the requirements included in the design
• **Fit for use:** It meets the requirements that are needed for its intended use.

The service validation and testing process helps to ensure that the service design package correctly specifies the requirements. It also helps ensure that the service provides value to the business, that it performs well in production, and that it meets the target level of quality.

This process focuses on the activities required to:

• Create and validate test plans
• Manage the test plans and test environments
• Conduct the test
• Verify the results
• Communicate the results to stakeholders
• Create test reports
• Evaluate the test according to exit criteria.

After the test phase, you need to evaluate the change to ensure it delivers value to the business.

**Change evaluation**

Determine whether the change meets the client’s needs in terms of how the updated code is to be used and whether it delivers the expected level of service. Change evaluation ensures that you understand the intended and unintended effects of a change. Change evaluation monitors the predicted performance and manages risk.

**Request fulfillment**

Request fulfillment is part of operations and focuses on the process to complete routine requests. Request fulfillment streamlines the completion of routine requests, to free resources to focus on more demanding and non-routine requests.

**Knowledge management**

The lifecycle processes covered under release control and validation provide the essential knowledge that is necessary to support reliable and effective services. The knowledge management process captures a wide array of information that can help drive the entire process. The configuration management database and the configuration management system are part of the service knowledge management system.

**DevOps practices that implement ITIL**

To help implement the guidance contained in the ITIL v3 framework, DevOps practices:

• **Focus on better communication**
• **Build software for discovery**
• **Establish baseline releases**
• Implement continuous delivery
• Automate the workflow
• Amplify feedback loops
• Improve process and automation iteratively

**Focus on better communication**

DevOps practices emphasize the goal to improve communication and collaboration between development and operations. The need for structures that enhance communication is described throughout the ITIL framework. DevOps practices enhance communication among development teams, operations teams, and other key stakeholders, including information security (InfoSec) and quality assurance and testing. These practices make it possible to share knowledge and preferred practices across groups, facilitate cross-functional teams, and reduce the influence of silos.

IBM products and solutions support the DevOps approach and accelerate adoption. They are built on open standards and designed to work with each other and with open source solutions.

**Build software for discovery**

DevOps practices automate the procedures to build, package, and deploy application software. Continuous integration builds the code with verifiable source code baselines that are secured in the development version control libraries. For milestone releases, the code baselines are stored in the definitive media library (DML).

These procedures match those used by SACM to programmatically discover and verify baselines. Automated discovery of configuration changes makes it possible to keep the CMDB and the CMS updated. Automated discovery depends on automated build procedures. To implement SACM and to include accurate, up-to-date and verifiable information in CMS and in the CMDBs, you need to build quality into the build process at the beginning. You need to embed version IDs in configuration items so that the version IDs can be used in a physical configuration audit. Many of these tools and techniques also make use of cryptographic hashes to verify the validity and integrity of a configuration item.

Consider using **Rational Team Concert** to enable you to collaborate with software development professionals across many disciplines (development, operations, test, and so on.)

Once the configuration items are built using automated and verifiable procedures, the next step is to package them into release baselines.

**Establish baseline releases**

Release and deployment management processes ensure that applications can be packaged and deployed. Create releases in verifiable packages with automated build and continuous integration practices. Ensure that these packages include a list of the configuration items contained in the package with a manifest that is cryptographically signed. The cryptographic hash serves two important functions. It verifies that the package comes from the intended source (non-repudiation) and it ensures that nothing has tampered with the package.
Continuous delivery

Continuous delivery depends upon the creation of automated scripts to support the deployment pipeline. To achieve continuous delivery, you need to script the entire application build, package, and deployment tasks. The deployment pipeline provides the ideal structure to ensure that information is automatically updated in the configuration management system, configuration management database, and the definitive media library. The deployment pipeline also provides information to the service knowledge management system. This information can support the continuous improvement of IT services.

Automate the workflow

The processes described in the ITIL v3 framework rely on workflow automation to ensure that stakeholders understand and complete the tasks assigned to them. Automated workflow makes it possible to trace requirements as they become implemented, and improves the ability to see changes, the effects of changes, approvals, and verifications.

Amplify feedback loops

DevOps focuses on ways to amplify feedback to all stakeholders. Feedback improves processes and increases effective communication. Online virtual communities help to facilitate effective communication overall process improvement.

Improve process and automation iteratively

The ITIL v3 framework relies on the iterative development process that is inherent in the DevOps agile culture. DevOps teams need to pilot these processes to gauge whether the implementation is effective and fit for use. With the help of the agile retrospective, teams evaluate each process to determine what went well, what did not go well, and what can be improved. With this approach, processes improve with each iteration.

Conclusion

The ITIL v3 framework provides excellent guidance on IT service management. It includes guidance for processes across the release control and validation lifecycle. DevOps provides the procedures that make ITIL effective and practical. With the right tools and processes, organizations can realize the value of reliable IT services that help facilitate the business and that improve quality and productivity.
Related topics

- ITIL
- IT Service Management Forum
- itSMF, US focused
- Out of the Crisis, W. Edwards Deming, MIT, 1986
- ITIL Service Transition 2011 Edition (Best Management Practices), Stuart Rance
- IBM DevOps
- Rational Quality Manager for task-based development
- Rational Team Concert for task-based development
- IBM UrbanCode Deploy