Armaan Shamsaasef

Technical Change Management

Notes: Technical Change Management (SY0-701 - 1.3)

Core Concept: Technical Change Management is the practical implementation of the change management process, using technology to enforce and automate changes. It focuses on the *how* of making changes in a controlled and consistent manner.

Key Technical Methods & Processes

1. Version Control

- **Definition**: A system that tracks and manages changes to files and code over time.
- **Purpose**: To maintain a history of all changes, allowing teams to collaborate, see who made what change, and revert to a previous version if a new change causes problems.
- **Common Tools**: Git, Subversion (SVN).
- Example: Storing all firewall configuration files in a Git repository. Every time a rule is added or modified, it's logged with a comment explaining the change.

2. Change Management Platform / ITSM

- **Definition**: A formal software platform used to manage the entire change management workflow.
- **Purpose**: To provide a centralized system for submitting, reviewing, approving, and documenting all change requests.
- **Common Platforms**: ServiceNow, Jira Service Management.
- Process in the Platform:
    1. Request is submitted as a ticket.
    2. Ticket is routed to the Change Control Board (CCB) for approval.
    3. Upon approval, the ticket is assigned for implementation.

4. All documentation, rollback plans, and post-implementation notes are attached to the ticket, creating a complete audit trail.

3. Automated Deployment (CI/CD)

- **Definition**: The use of automated pipelines to test and deploy changes.
- **Purpose**: To ensure changes are applied consistently and without human error. Automation is fast, repeatable, and provides a clear success/failure status.
- Key Concepts:
    - CI/CD Pipeline: An automated process that builds, tests, and deploys code.
    - Sandbox/Staging Environment: A isolated copy of the production environment where changes are tested automatically before being deployed.
- Example: A developer commits a code change to a version control system. The CI/CD pipeline automatically deploys that change to a staging server, runs tests, and if all tests pass, deploys it to production.

4. Checklists & Documentation

- **Definition**: Formalized lists of required steps and documentation that must be completed for every change.
- **Purpose**: To ensure no critical step is missed and that the process is followed consistently every time.
- What it Includes:
    - Purpose of the change.
    - Exact steps for implementation.
    - Detailed Rollback Plan.
    - List of impacted systems.
    - Post-change validation steps.

The Technical Workflow

1. A change is proposed and documented in a Change Management Platform.
2. The change (e.g., a new script or config file) is stored and versioned in a Version Control system.

3. After approval, the change is deployed using an Automated Deployment pipeline to a sandbox for testing.
4. If tests pass, the change is deployed to production, with all steps tracked against the original ticket.
5. The entire process is guided and verified using a Checklist.

Key Takeaway

Technical Change Management brings rigor, consistency, and automation to the change management process. It replaces error-prone manual changes with a traceable, repeatable, and reversible workflow. This is critical for maintaining system stability, security, and a clear audit trail for compliance.