TYPES AND LEVELS OF CHANGE MANAGEMENT

Successful IT service operations require proactive management of infrastructure changes associated with configurations, resource provisioning, and service management. These changes are not always planned, often emerging as unforeseen consequences of a service disruption. Technology dependencies in a complex IT infrastructure environment can cause the impact of a small change in the infrastructure to escalate across the IT environment and impact multiple users at scale.

Organizations therefore adopt change management strategies, solutions, and practices that help manage infrastructure changes and mitigate the associated risks. Let's take a look at change management types and levels.

What is change?

ITIL® defines change as "the addition, modification or removal of anything that could have an effect on IT Services". (Change management has undergone its own changes between ITIL v3 and the new ITIL v4.) Though ITSM frameworks vary in best-practice workflows, most follow a standard hierarchy of change:

- **Standard change.** A low-risk change that is pre-authorized and follows documented tasks per a change model, which outlines a repeatable workflow to manage such changes. Examples include an IT service request from the service desk platform.
- **Normal change.** An intermediary-risk change that cannot be categorized as an urgent issue or

a pre-approved change process. A thorough review process is conducted before approving such changes.
- **Emergency change.** Urgent changes that may present high-risk consequences if not addressed promptly. An emergency change may need to be resolved to avoid a major incident or to resume normal operations following the incident occurrence. Examples include security threats and infringements or power outages.

Prioritizing changes levels

Ensuring beneficial changes are realized through minimal service disruptions to the organization is the goal of change management. Changes can be tactical, operational, or strategic. In the context of ITSM, change management should enable business continuity and reduce the risks associated with changes performed on the underlying IT infrastructure.

Many online retail companies freeze changes during the peak shopping season to reduce the risks of IT service outages. However, innovative online service providers follow the DevOps SDLC methodology of continuous improvement, release, and deployment cycles to deliver feature improvements throughout the year. For example, Etsy is known to perform 50 deployments per day through a fully automated and continuous delivery pipeline.

A strategic approval and review process is necessary to realize such goals. The approval process necessary for a change can depend on decision criteria unique for every organization. The decision criteria can define how the changes are prioritized. According to ITIL guidelines, impact and urgency can be used to prioritize change requests.

- **Impact** evaluates the business impact of a proposed change request. It also accounts for potentially damaging consequences resulting from unsuccessful execution of a change that are not previously considered. The ranking may range from minor impact to extensive impact.
- **Urgency** evaluates the necessary time for a change to realize an Impact. A change request that requires quick implementation or one that must be initiated early to account for prolonged implementation duration is ranked with high urgency. The ranking may range from Low Urgency to High Urgency.
- **Priority** indicates the relative importance of a change request is determined by correlating the Impact and Urgency.

Here is an example of a priority matrix:

	Urgency			
Impact	Critical	High	Medium	Low
Extensive	1	1	2	4
Significant	1	2	3	4
Moderate	2	2	3	4
Minor	2	3	3	4

More details on this Priority Matrix and the definition of Urgency and Impact criteria can be found on this BMC documentation.

DevOps and change management levels

The DevOps framework relies on automation and collaboration among development, QA and Ops teams throughout the SDLC pipeline. DevOps principles appear to contradict ITIL guidelines in some respects. DevOps encourages the Fail Fast philosophy whereas ITIL guidelines are designed to avoid failure through rigorous change management controls. In many ways, DevOps is compatible with ITIL and supports risk-free change management processes applicable to changes of all levels and types.

Compare the popular microservices approach in DevOps versus the traditional strategy to deploy and manage applications. Microservices architecture decomposes application or components into multiple single-purpose services. For example, payment processing, inventory catalog, and customer support services can be separate microservices communicating with each other to deliver a full application level functionality. Each microservice offers a unique and segregated functionality that is operated autonomously. The impact and urgency of changes are independent across multiple microservices, thereby allowing for small, frequent and continuous changes to the individual microservices.

These changes are considered 'normal changes' that require approval from the Change Advisory Board (CAB) on a regular basis. Both developers and operations personnel can take key roles in the CAB committee since change management in DevOps encourages minimization of large 'emergency changes' while maximizing the scope of pre-approved 'standard changes'. Automation is leveraged to facilitate implementation of 'standard changes'. Risk mitigation workflows are employed to prevent unforeseen impact of new changes. Automation also expedites the approval, deployment, and control of normal changes. The result is a change management process that relies on objective prioritization criteria with less bureaucracy and high stakeholder satisfaction.

Change management is traditionally followed as a checklist strategy presented as a sufficient process to mitigate potential risks. The responsibility of enterprise IT goes beyond following a checklist that defines IT changes as a high or low priority on a static decision basis. Consider the conflicting goals facing DevOps teams and the CAB, QA, or IT departments. Traditional IT departments prefer a stringent change management process where most requests for new changes

undergo a careful review and approval process. On the other hand, the Agile mindset of progressive product development teams want to push new changes into production faster.

This conflict creates disjointed SDLC processes and prevents members to work as a team. If the change management efforts are limited to specific components in the SDLC pipeline such as QA or testing, the wider impact and urgency of a change type may go under the radar. On the other hand, reevaluating small and frequent changes across the iterative delivery pipeline can help increase the percentage of updates that can be categorized as 'standard changes'.

For a DevOps strategy to be compatible with ITIL guidelines, it is important to re-evaluate the evolving impact and urgency of changes; simplifying and automating the change management process; and observing change management as an enabler to continuous improvement instead of a barrier preventing rapid release cycles.