10 Key Rules for Using the ITIL Framework Effectively

Debasis Bandyopadhyay, CISA, CISM, CGEIT, PMP, TOGAF9, ITIL v3 (F), has more than 25 years of industry experience in the field of RADAR (Radio Detection and Ranging) technology; satellite communication; VSAT (very small aperture terminal)-based network design for voice, data and video communication; telecommunication switching networks; electronic payment systems; and IT service management. He is the chief information security officer (CISO) at RS Software Ltd. in India, and holds the COBIT® Foundation Certificate.

IT Infrastructure Library (ITIL)[1] is well established as the framework of best practice guidance for IT service management in the industry. This article shares some of the practical challenges in IT service management (ITSM) and how the ITIL framework can be used to overcome those challenges. The following 10 key rules for using the ITIL framework effectively are based on the real-life experience of IT engineers and IT managers:

1. **Do not ignore system-generated alerts**—IT service engineers and managers are generally maintaining large and diverse IT systems with hundreds of applications running in multiple servers, databases and networking equipment. They may find that many alerts are generated every day and auto-generated emails are sent by the monitoring system to concerned groups. The rule of thumb is: Do not ignore alerts. There are many cases in which alerts are ignored, leading to major problems with serious consequences, including massive financial losses.

   The question is how many alerts one needs to address when one is getting hundreds of alerts every day. Considering the potential problems of unaddressed alerts, one should carefully categorize and prioritize alerts that can have the most impact, and address them quickly. To do this, IT service engineers need to understand the business impact and the criticality of the applications they support. If in doubt, they should escalate or inform the application owner without losing any time. This can prevent major catastrophes.
2. **Understand the difference between a service request and incident management**—Service requests (part of the request fulfillment process of ITIL) are simple requests or questions such as "Please unlock my domain password" or "Where do I find the FTP software for downloading?" These requests can be addressed immediately by help-desk professionals or by the requesters, if they are directed to a self-help site where most of the common questions and the corresponding solutions are provided.

   On the other hand, incidents (part of the incident management process of ITIL) are unplanned interruptions to an IT service or reduction in the quality of IT services. Incidents are logged, tracked and analyzed until closed. So, IT engineers should not unnecessarily treat service requests as incidents, thereby increasing the number of incidents that need more effort to resolve and that can result in overhead, such as incident aging calculation, incident categorization, root-cause analysis (RCA), resolution steps and closure.
3. **Write in detail the incident resolution steps and lessons learned**—It is a common practice that after diagnosing and solving an incident, IT service engineers do not detail the steps followed to solve the incident. There is space provided in every incident management tool to elaborate on the resolution steps in plain text. Frequently, the IT service engineers write "fixed," "problem solved" or "closed" in that space. Unless the IT service engineers elaborate on the methods used and steps followed to resolve the incident, it will be of no help for other service engineers to reference in the future when encountering the same or similar types of incidents. Detailing the resolution steps enhances the knowledge repository of the enterprise, improving the effectiveness and efficiency of the knowledge management process.
4. **Understand the problem clearly before jumping in to solve it**—In many situations, the IT service engineer does not understand the problem clearly and tries to solve it immediately. For example, Jenise picks up her phone and tells the IT service engineer that she is unable to log in to a server but needs to

do so urgently. In this case, the IT service engineer should not jump in to solve why Jenise is not able to log in to the server. First, the IT service engineer should ask questions such as "When was the last time that you logged in to this server?" By asking this question, the IT service engineer may discover that Jenise has never logged in to that server and has no reason to log in. The deeper question is: Why did Jenise suddenly require login to a server into which she has never logged in to previously? Probably Jenise was part of a group email ID and, as part of that communication, she got an email (asking her to log in to the server) for which she need not take any action. The email was probably meant for other team members who really needed to log in to find more details, and was not meant for Jenise.

5. **Incident management—go parallel, not serial**—Let us assume that an incident has been reported, and based on that, an incident ticket has been opened and assigned to a service engineer. The service engineer's job is to categorize and prioritize the incident to figure out its severity. In a high- or medium-severity incident, one needs to open a teleconference bridge and ask all concerned parties—the database administrator (DBA), middleware persons, application owners, server and network administrator, and even the supplier's representative—to join. The idea is to bring all the stakeholders onto one common platform and find out what has gone wrong. The advantage of this mode of parallel working is that no one can pass the responsibility to another saying, for example: "It is not my problem; the DBA needs to look into this." The problem can be detected and fixed or a work-around can be found in a short period of time compared to a serial method of calling one individual at a time. So the mantra is: "Go parallel, not serial."

   Another important point is that the incident ticket should be closed by the person who has raised the incident and not the person to whom it was assigned. Having the service engineer to whom the ticket is assigned close it is a common mistake. The service engineer can say that the problem has been resolved, but not closed. The incident owner, when he/she is satisfied with the solution, should close the incident ticket.
6. **Keep a close watch on how the new or modified system is working; the importance of post-implementation review (PIR)**—PIR is a very tricky aspect of service management. When the IT service developer has developed a new service or has modified the existing service and moved it to production after proper testing, as per the test plan, the first point to remember is to call all the stakeholders (e.g., the DBA, middleware, application owner, server and network maintenance folks) before moving a new code to the production environment. A quick smoke test[2] must be conducted to verify that the main functionalities of the service are working fine, and then the code can be moved to production. IT service developers cannot take their attention from a new or modified service even after successful movement of the code to the production environment. Problems can begin to surface in the production environment even after three months. Therefore, it is important to set up a PIR process and keep checking the service input and output results, database and error logs. The results may show, for example, that a field of a database table is not getting populated due to a coding error—a problem that would only surface in a report due every three months. Rather than waiting three months for the problem to surface, the IT service developer must keep checking the system and do smoke tests periodically to check major functionalities and the back-end results. It is important to include the PIR effort in the project effort estimation process.
7. **Automate the processes that are repetitive in nature**—It is always advisable to automate processes that are repetitive in nature. This can be achieved by writing a small script or even a full-fledged system. Automation brings higher productivity and reduces manual errors.

A word of caution for processes automation: Simplify the process and clarify the flow of activities before automating them. One should not be in hurry to automate tasks that are neither simple nor routine.

8. **Understand the problem from the customer's perspective and provide solutions accordingly**—It is important to understand the problem from customers' perspectives and the context in which they are looking for solutions. For example, an IT service engineer gets a call from a customer who says that her printer is not working and she has to print a document to circulate in a meeting that is taking place in 15 minutes. The service engineer can judge from the customer's voice that she is extremely anxious. In this situation, the service engineer should not try to fix the printer, but rather give the customer an alternate solution, such as printing to a different printer. Considering the situation and the urgency, this might be a better solution than asking the customer to take the time for routine checks to fix the printer.
9. **Put 'known solutions' in the intranet (S-help, FAQs) to reduce the number of calls**—The IT service manager should analyze the service calls and determine the most common problems reported by various users. If the solutions are known, they can be placed in the organization's intranet in the form of self-help (S-help) or frequently asked questions (FAQs). This will save a lot of time and effort and will improve process efficiency.
10. **Check before making any changes to ensure that no one else has initiated those changes**—IT service developers have to deal with frequently changing management requests. IT service developers may encounter a need to change code, database or other parameters to accomplish the desired outcome. This is part of the change management process of ITIL. Before planning any change, it is critical to make sure that the same changes have not already been taken up by other groups within the enterprise. This is a typical problem of large, global organizations. It is important to avoid duplication of work and improve service effectiveness and efficiency.

CONCLUSION

These 10 key rules are based on the real-life experiences of IT professionals who work in IT service management. These rules can help the IT professional effectively use the ITIL framework.

It was once a headline in all leading newspapers of Singapore when an alert was ignored while conducting a routine repair job and the oversight caused a major impact to a chain of banks in Singapore. As a result, the banking operation was shut down for three hours. So what went wrong? It might be that the key first rule was not followed: Do not ignore system-generated alerts. What actually went wrong is not publicly known, but there are numerous examples like this of disasters that could have been averted if simple key rules had been followed.

> "Simple automation can improve productivity significantly."

Additionally, simple automation can improve productivity significantly. The automation of running shell scripts or other routine jobs can avoid problems with manual errors. Manually running a wrong shell script may bring down the wrong server or even a complete data center at the wrong time.

Implementing ITIL practices in an organization with a disciplined approach, which can be done more effectively with the 10 key rules provided here, can help organizations reach higher maturity levels.

ENDNOTES

[1] Information Technology Infrastructure Library (ITIL), *Best Practice Guidance*, 2011, *www.best-management-practice.com/Knowledge-Centre/Best-Practice-Guidance/ITIL/*

[2] This term was probably coined from hardware testing in which one ensures that no smoke comes out when the new hardware system is plugged in to the power socket.

The *ISACA Journal* is published by ISACA. Membership in the association, a voluntary organization serving IT governance professionals, entitles one to receive an annual subscription to the *ISACA Journal*.

Opinions expressed in the *ISACA Journal* represent the views of the authors and advertisers. They may differ from policies and official statements of ISACA and/or the IT Governance Institute® and their committees, and from opinions endorsed by authors' employers, or the editors of this *Journal*. *ISACA Journal* does not attest to the originality of authors' content.

www.isaca.org