realtimepublishers.com®

The Definitive Guide™ To

Enterprise Network Configuration and Change Management

VOYENCE™

Don Jones

Copyright Statement

Chapter 8: Sample Change Management Processes

Chapter 7 introduced some concepts that might be intimidating for smaller organizations: Who has the time or personnel for a CAB or executive committee in a small shop? In this chapter, I'll show you that change and configuration management can be adapted to organizations of any size and tweaked to meet specific business concerns such as speed of response and security control. This chapter will serve as a resource for example change and configuration management processes, which you can adopt outright or (more likely) modify to suit your specific needs.

You'll notice that all of these processes have some common elements, such as a review process to help catch changes that will cause problems, documentation, and disaster recovery procedures. These elements have been covered in nearly every chapter of this guide, and they are the underlying parts that make change and configuration management worthwhile. Even if you decide to create your own process entirely from scratch, be sure that the common elements are included so that your process will offer the value that change and configuration management promises.

Basic Change Management for Smaller Organizations

The most difficult aspect of incorporating change and configuration management into a smaller organization is that most processes are set up for much larger organizations; organizations that have the personnel and other resources to conduct peer reviews, change categorization, and so forth. But change and configuration management can provide just as much benefit to smaller organizations in which the entire "IT staff" might consist of one or two technical professionals and their manager; implementing change management in such organizations is simply a matter of developing a process that acknowledges the limitations of the organization. In fact, smaller organizations, it can be argued, benefit *more* from proper change and configuration management, at least in proportion to the organization's size. Downtime and other IT setbacks create bigger problems for small organizations, as they rarely have the resources—especially personnel and time—to spare for troubleshooting and resolution. By keeping everything up and running smoothly, the organization can focus on beneficial projects rather than simply firefighting.

Processes Incorporating Peer Review

Figure 8.1 shows a basic one-person process, in which the entire change and configuration management activity is handled by a single individual, with input from management.

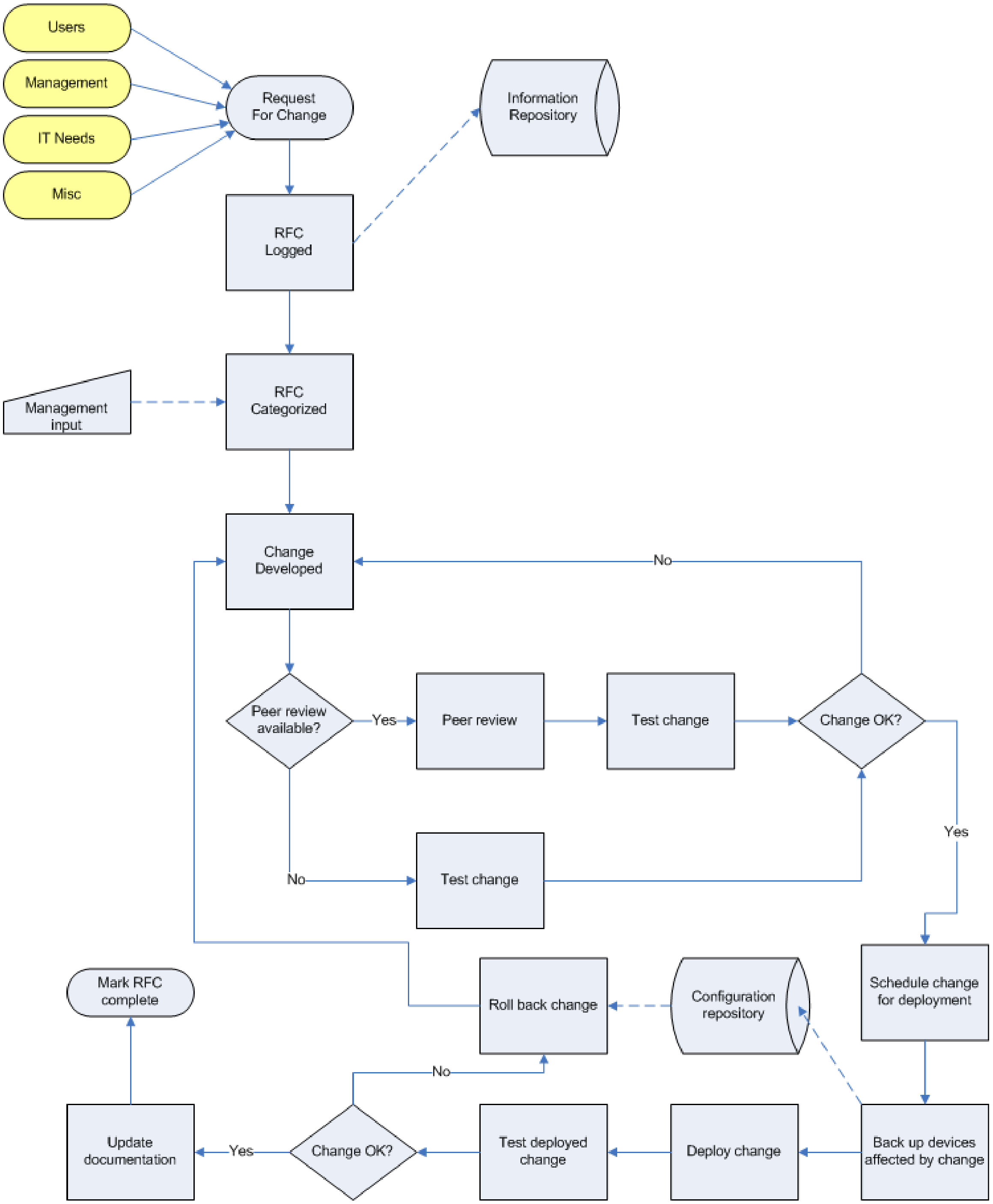

Figure 8.1: Change management process for small organizations.

Let’s walk through the process that this figure illustrates:

1. Changes come in and are logged as Requests for Change (RFCs). This information should be stored in some sort of repository, such as a Help desk ticket tracking system.
2. The RFC should be categorized (and prioritized) with input from management. Doing so will properly involve management in the process of deciding which changes are implemented first; with limited personnel resources, this step is crucial in managing the overall process.
3. When it is time to make a change, the change is developed—a process that includes creating whatever configuration file changes that are necessary on your devices, documenting those changes, and so forth.
4. If you *have* a colleague who can review your change—perhaps another IT staffer or a manager who has technical experience—ask that person to do so. In addition to this review, test the change in a lab environment, on a device emulator, or through some other fashion (such as deploying it late at night in the production environment, perhaps in a limited fashion, with a test recovery plan in place). However, if you *don’t* have the resources for a peer review, you’ll simply have to skip it and go straight to testing.

> ☞ Find out if your area has a user’s group for the network equipment your organization uses. For example, you might find a local Cisco user’s group. Joining the user group will introduce you to other professionals who may be in the same situation as you are; perhaps you can agree to periodically help one another by doing short peer reviews of proposed changes via email or some other means. In this fashion, smaller organizations can team up to help one another out; you’ll also make valuable industry contacts.

5. If the change passes the peer review and/or testing process, you can schedule it for deployment. This deployment scheduling process might include notifying users of any impact to network services.
6. *Always* back up your devices prior to making a change. Even if you have a solution that automatically backs up devices, perform a manual backup so that you’ll know for sure that you are protected.

7. Deploy the change, then immediately test it for accuracy. If any potential problems were noted during the peer review, test for those problems to make sure everything is working.
8. If the change didn't work out as planned, immediately rollback the change using the backup. Go back to the drawing board and redevelop the change.

> Do not attempt to repair a failed change deployment on the fly! You'll defeat the entire point of the process, and you stand as much a chance of doing additional harm as actually fixing the problem. When a change doesn't work out as expected, the *only* option is to roll it back completely and start over.

9. If the change did work out as expected, update your network documentation to reflect the change. Mark the RFC as completed in your information repository (such as your Help desk ticket tracking system).

As you can see, this process isn't overly burdensome and can be accomplished with a very small IT staff. It is missing some of the elements that help a larger organization better manage a larger volume of changes, but it incorporates the key elements of categorization and prioritization, peer review and testing, backup and recovery, and documentation.

Processes Incorporating Peer and Supervisory Review

In organizations in which management likes to take a more active role, several of the change and configuration management process steps can be offloaded to management. Management can become involved in a review process that will help them make better change decisions in the future. Figure 8.2 shows a modified small organization change and configuration management process that still requires a minimum number of IT staffers, but provides more steps for management to handle (the green boxes designate management-oriented steps).

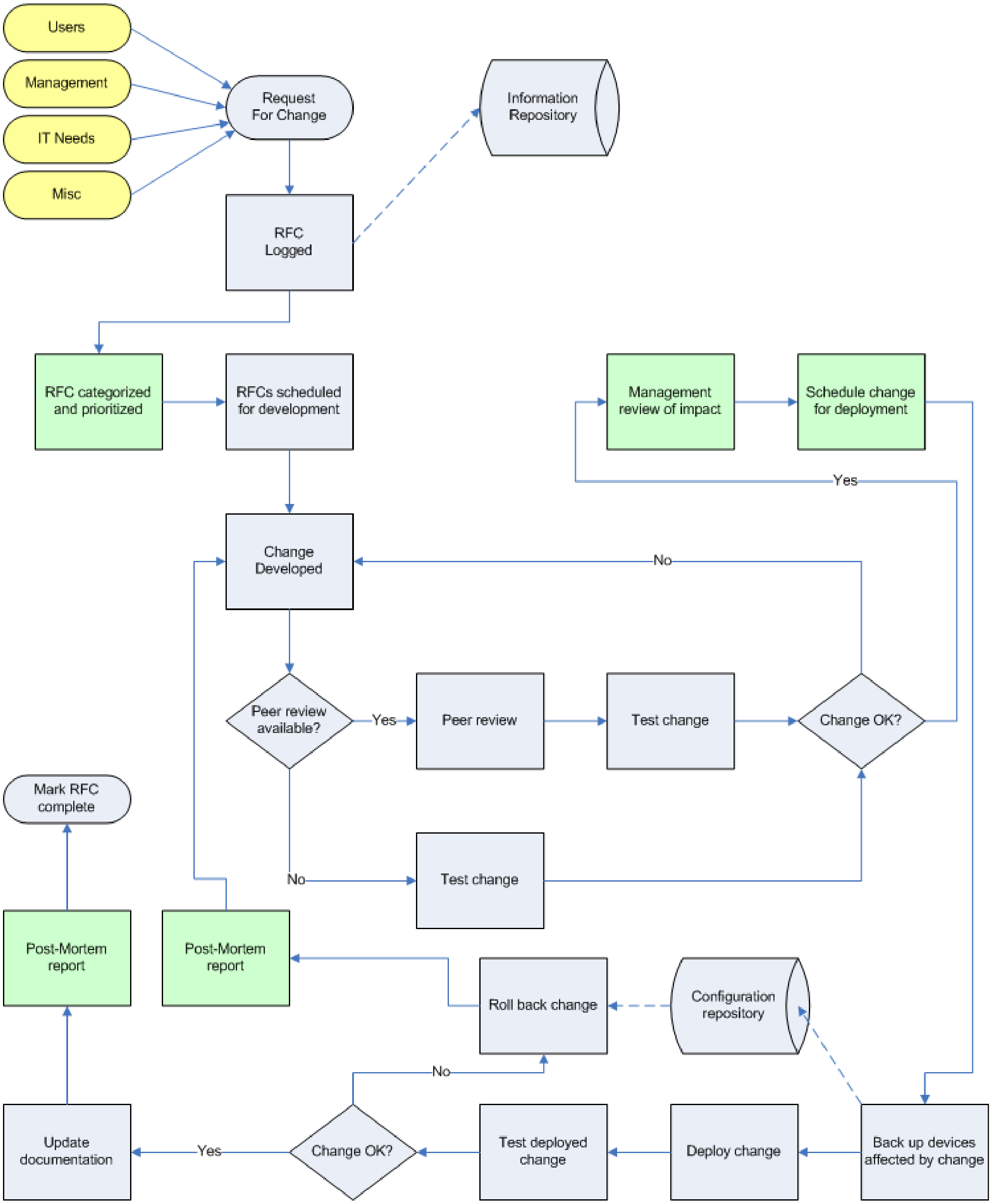

Figure 8.2: Process for small organizations, incorporating supervisory review.

Let’s walk through the process that Figure 8.2 illustrates:

1. As in the previously discussed process, RFCs come in from a variety of sources and are logged.
2. Management steps in to completely take over the categorization and prioritization rather than simply providing input. In fact, as Figure 8.3 shows (which depicts a minor revision to the process), management can also take over the task of accepting changes. This process modification allows management to provide a filter on changes coming in from other sections of management, from users, and so forth—similar to the role that the CAB plays in the ITIL process (which Chapter 7 explored). This filtering role is especially useful in a small environment because it takes the IT staff out of the “line of fire” and allows management to make all the decisions regarding changes to the infrastructure.

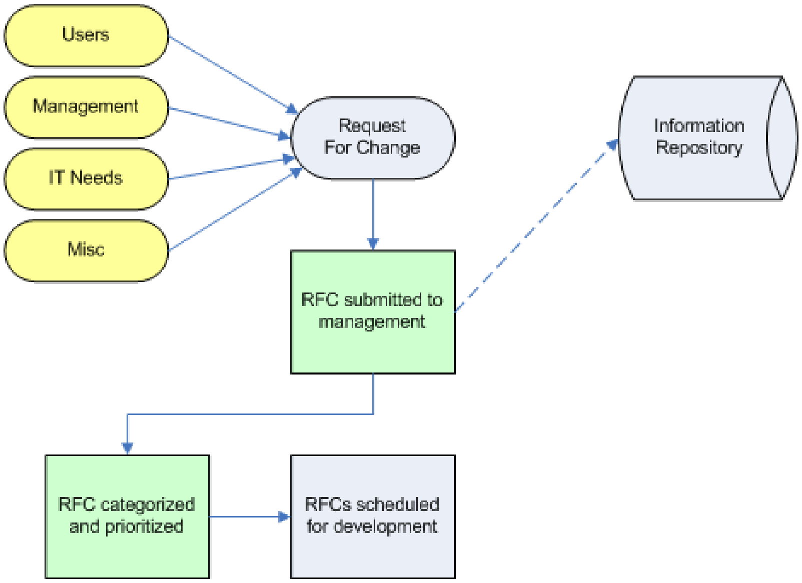
Users
Management
IT Needs
Misc
Request For Change
Information Repository
RFC submitted to management
RFC categorized and prioritized
RFCs scheduled for development

Figure 8.3: Allowing management to become a filter for incoming changes.

3. Once the RFC is approved for implementation, it’s scheduled. Although management will certainly have input regarding the schedule, this area should remain IT-driven because the presence of other projects, changes, and activities will impact the scheduling of any given change.
4. Once scheduled, the change can be developed. This part of the process, being technical in nature, doesn’t change much from the previous process (illustrated in Figure 8.1). There are still dual paths incorporated in the flowchart for when peer review is an option and when it isn’t.

5. If the change is approved and works properly when tested, the change can go on to final scheduling. At this point, management can again become involved because the actual deployment can have user impact. This process also includes a step in which management reviews the potential impact of the change—this review will directly drive the deployment schedule.
6. As the deployment is technical in nature, this section of the process remains unchanged. Start by backing up your devices, then deploy the change.
7. After deployment, an immediate test should occur, followed by a binary decision: do you leave the change in place or roll back the change?
8. In either event, a management post-mortem review should occur. In the case of a failed change that is rolled back, this review should focus on what failed and how the change can be modified to be successful the next time. Care should be taken to highlight any management decisions (whether made by management or not) that contributed to the failure. For example, perhaps insufficient time was allotted for the change to be deployed, and it had to be rolled back so that the network could be brought online according to schedule. In the case of a successful change, a post-mortem review should focus on areas of the process that proved problematic, and examine the process itself to make sure it performed as efficiently as it could have.

As you can see, even small environments can successfully use the key elements of a change and configuration management process. Figure 8.4 re-examines this process; highlighted in red are the key steps that *every* change and configuration management process should contain. These should help you review your own processes to make sure they contain all the steps necessary to minimally meet industry best practices.

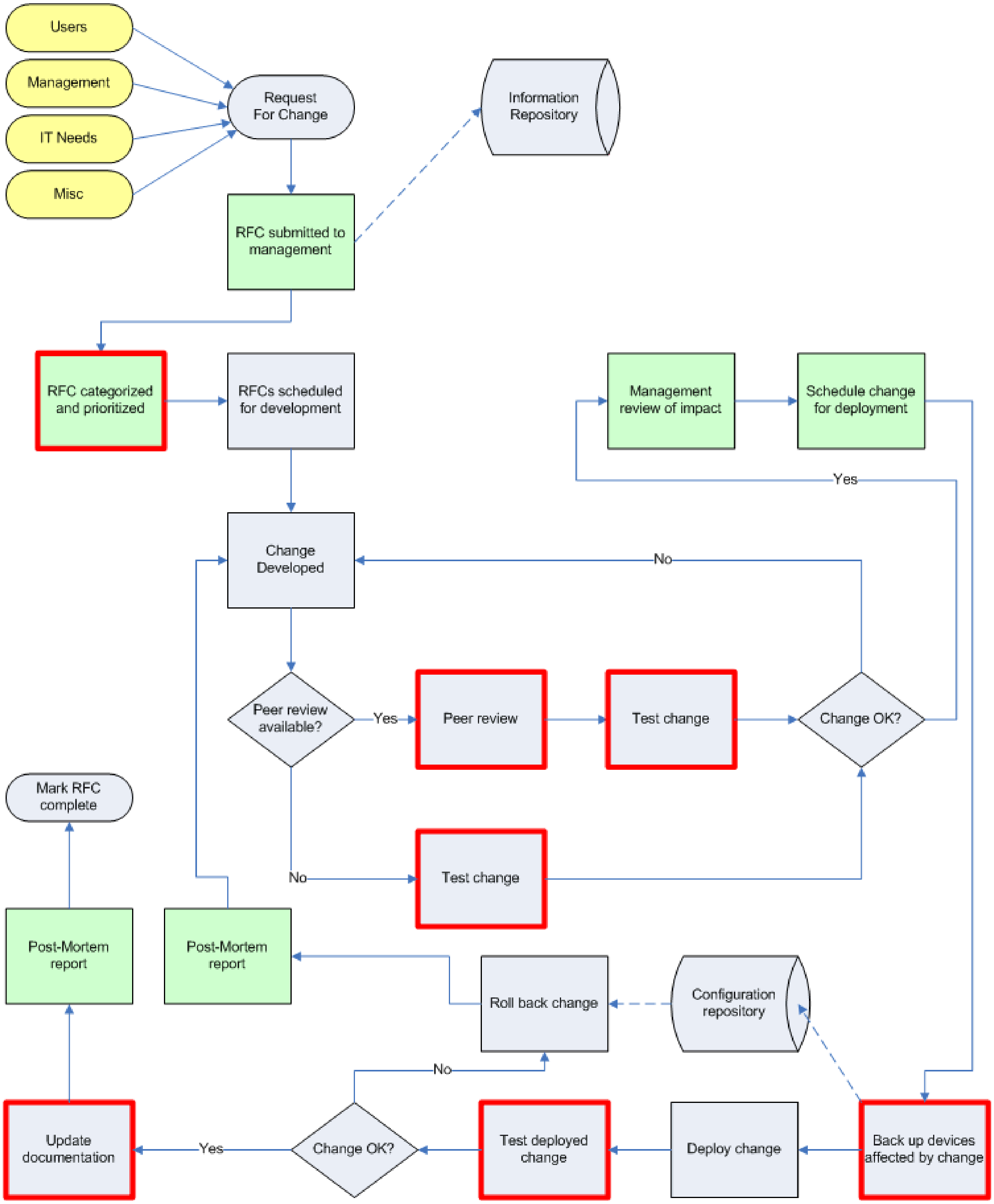

Figure 8.4: Key steps to be included in every change management process.

Shared Management Processes

One situation in which change and configuration management becomes especially complex is when management of the network is shared between a consulting (or outsourcing) firm and its customer. In some cases, the customer retains responsibility for the engineering of their network, with input from the consultant; in others, the customer merely has an oversight role and the consulting firm handles all of the technical details.

Many customers assume that their consulting firm has a change management process in place; customers should never *assume* anything about a consulting relationship and should ask to be made a part of the change management process. The processes presented in the next two sections are intended to help customers and consultants better manage change in the customer's environment, while retaining a streamlined operation.

The Customer Has Engineering Responsibility

Situations in which the customer has engineering responsibility are perhaps more difficult than those in which the consultant maintains this responsibility. The reason is that the customer is essentially creating technical blueprints that the consultant must then implement, manage, and maintain. We'll explore two processes for this situation; Figure 8.5 shows the first.

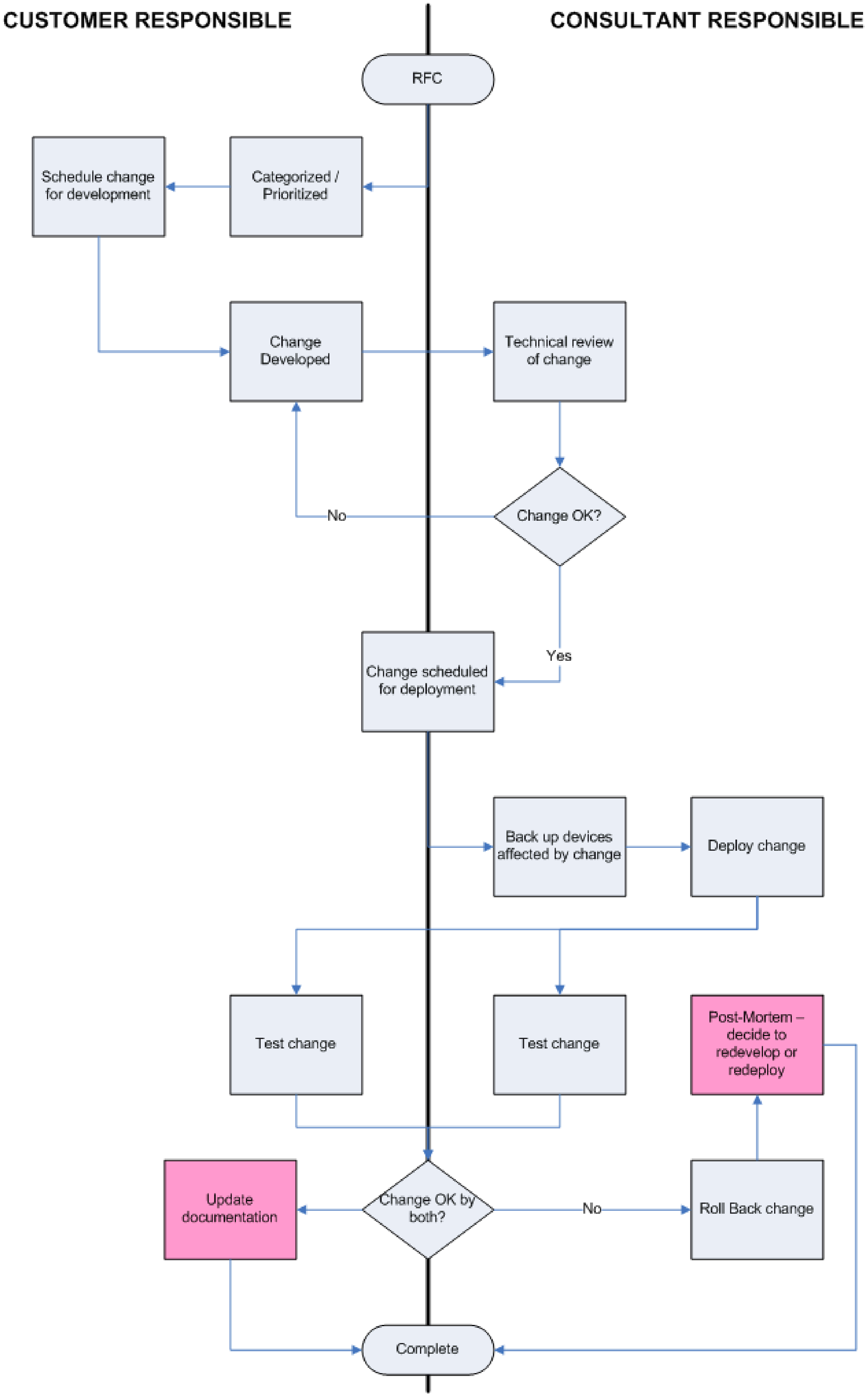
CUSTOMER RESPONSIBLE
CONSULTANT RESPONSIBLE
RFC
Schedule change for development
Categorized / Prioritized
Change Developed
Technical review of change
Change OK?
No
Yes
Change scheduled for deployment
Back up devices affected by change
Deploy change
Test change
Test change
Post-Mortem – decide to redevelop or redeploy
Update documentation
Change OK by both?
No
Roll Back change
Complete

Figure 8.5: A dual-responsibility process in which the customer retains engineering control.

The following steps highlight how this process works:

1. RFCs are submitted by either the customer or consultant (shown by this step straddling the line between the two).
2. The customer will generally categorize and prioritize changes; this responsibility should be the customer's because this is the customer's network.
3. The customer also takes responsibility for scheduling the development of the change, and actually developing it. Of course, as the customer has a business relationship with the consultant, this scheduling process should take into consideration any factors that the consultant raises.
4. The consultant should be responsible for the peer review of developed changes, because the consultant is the one who will have to work with the change. However, as Figure 8.6 shows (which shows the customer's tasks on the left side), it is not a bad idea for the customer to conduct an internal peer review first (if the customer has the resources to do so).

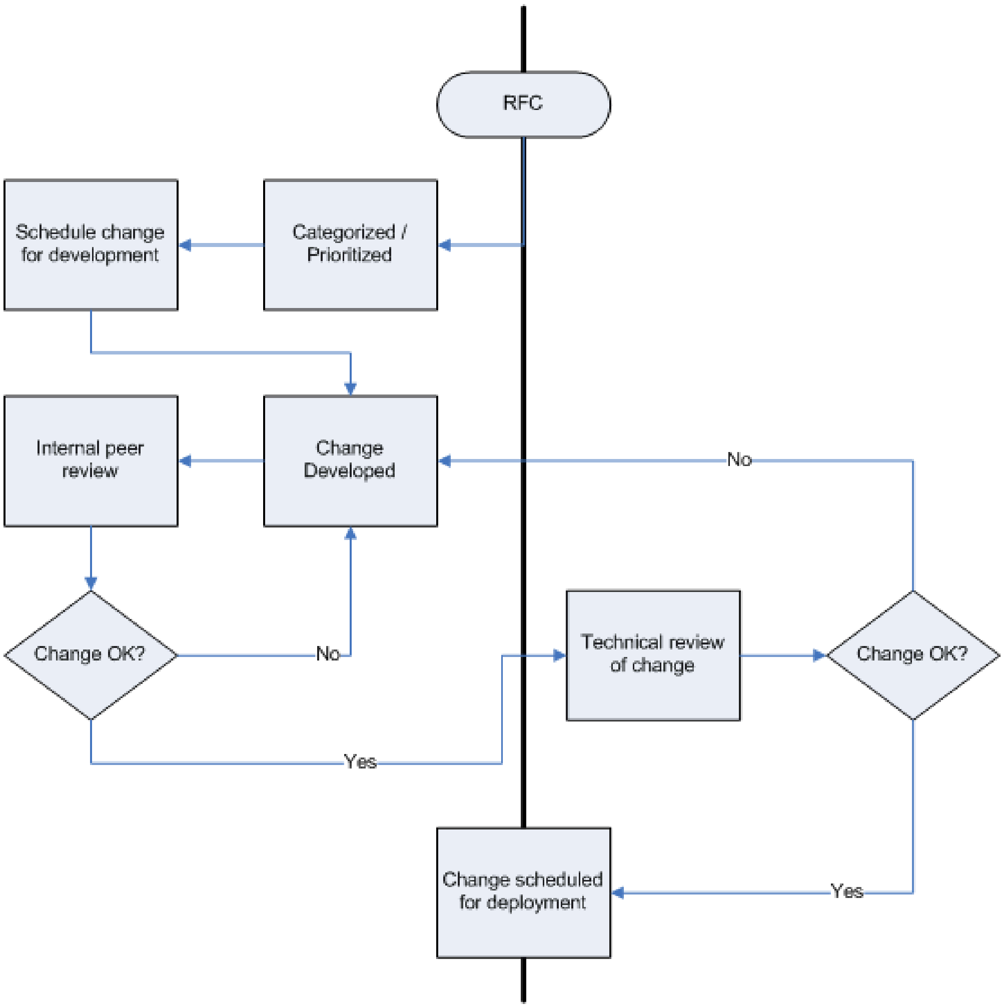
RFC
Schedule change for development
Categorized / Prioritized
Internal peer review
Change Developed
No
Change OK?
No
Technical review of change
Change OK?
Yes
Change scheduled for deployment
Yes

Figure 8.6: Adding an internal peer review to the dual-responsibility process.

5. Once the full gamut of peer reviews is over, the change is scheduled for deployment. This task is a joint task for the customer and consultant. The consultant needs to position resources to perform the change, while the customer needs to schedule changes to minimize business impact.
6. The consultant has responsibility for backing up devices and deploying the change.
7. The consultant and customer should both conduct independent tests of the completed change. A unanimous decision to keep the change deployed is required; otherwise, the consultant is responsible for rolling back the change.
8. If the change is rolled back, the consultant and customer should conduct a post-mortem to decide what to do next. The problem might lie in the technical aspects of the change, requiring redevelopment; the problem might also lie in the deployment, requiring the consultant to reschedule it.
9. If the change is retained, network documentation should be updated. This task may be performed by either party, but should be verified by both, as both the customer and consultant will rely on this documentation in the future.

A well thought-out process can maintain all the key features of a good change and configuration management process, even when responsibilities are divided between two organizations. It is important that both organizations agree to the process, and that they both follow the process. Communications are, of course, important, and means should be in place to allow free communications across organization lines during key parts of the process. Email might be insufficient because it is often difficult to keep email messages in a long-term form that is suitable for recordkeeping; instead, a ticket tracking system or other database is a more efficient and reliable means of communicating during the process. Figure 8.7 illustrates the key points of communication during the process, along with suggestions for how to best implement those communications.

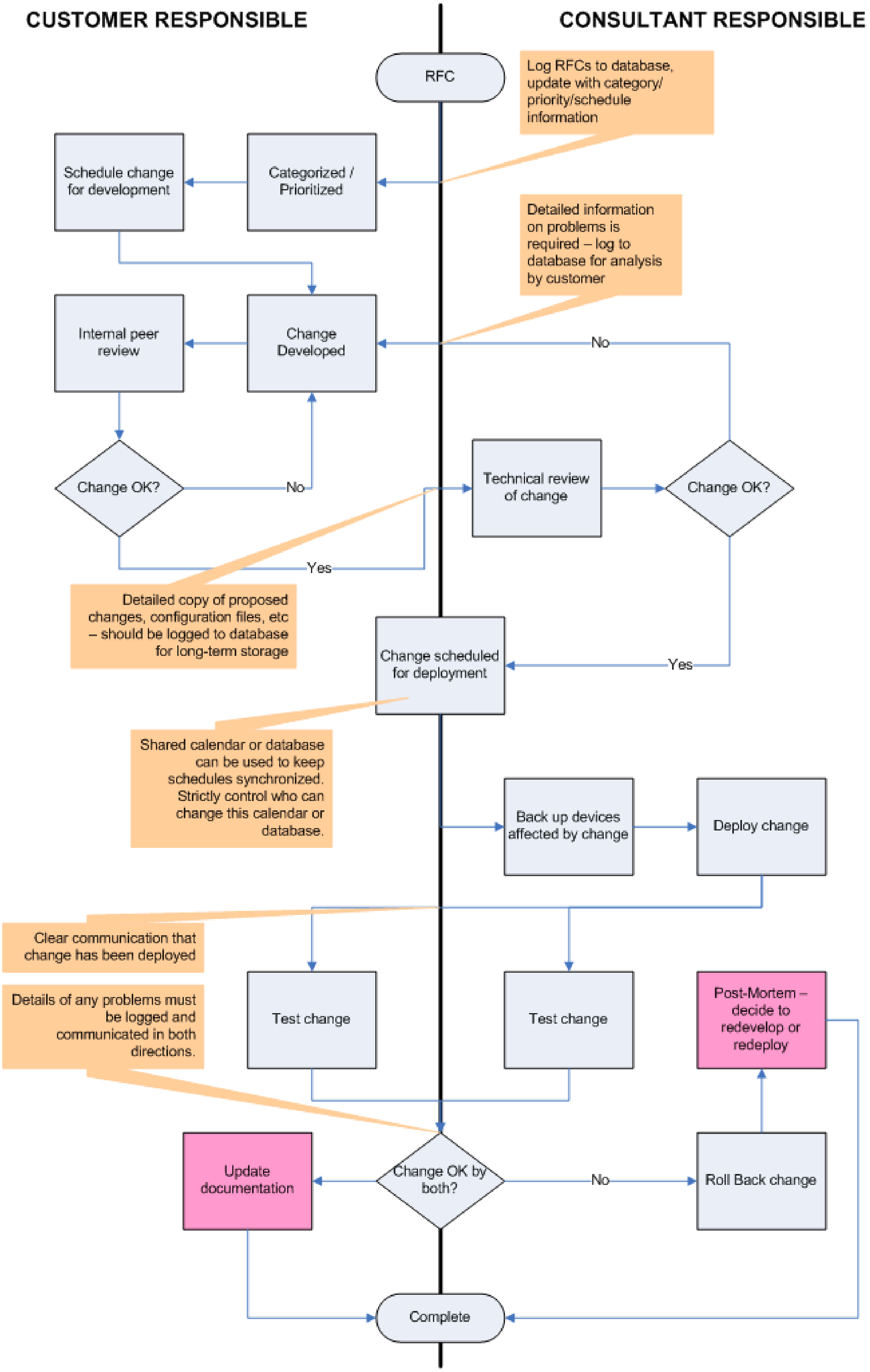
CUSTOMER RESPONSIBLE
CONSULTANT RESPONSIBLE
RFC
Log RFCs to database, update with category/ priority/schedule information
Schedule change for development
Categorized / Prioritized
Detailed information on problems is required – log to database for analysis by customer
Internal peer review
Change Developed
No
Technical review of change
Change OK?
Change OK?
No
Yes
Detailed copy of proposed changes, configuration files, etc – should be logged to database for long-term storage
Change scheduled for deployment
Yes
Shared calendar or database can be used to keep schedules synchronized. Strictly control who can change this calendar or database.
Back up devices affected by change
Deploy change
Clear communication that change has been deployed
Details of any problems must be logged and communicated in both directions.
Test change
Test change
Post-Mortem – decide to redevelop or redeploy
Update documentation
Change OK by both?
No
Roll Back change
Complete

Figure 8.7: Lines of communication in a dual-responsibility process.

The Customer Has Oversight Responsibility

In the process that Figure 8.8 shows, the customer has significantly less responsibility. The customer primarily approves general scheduling issues and has an oversight capacity.

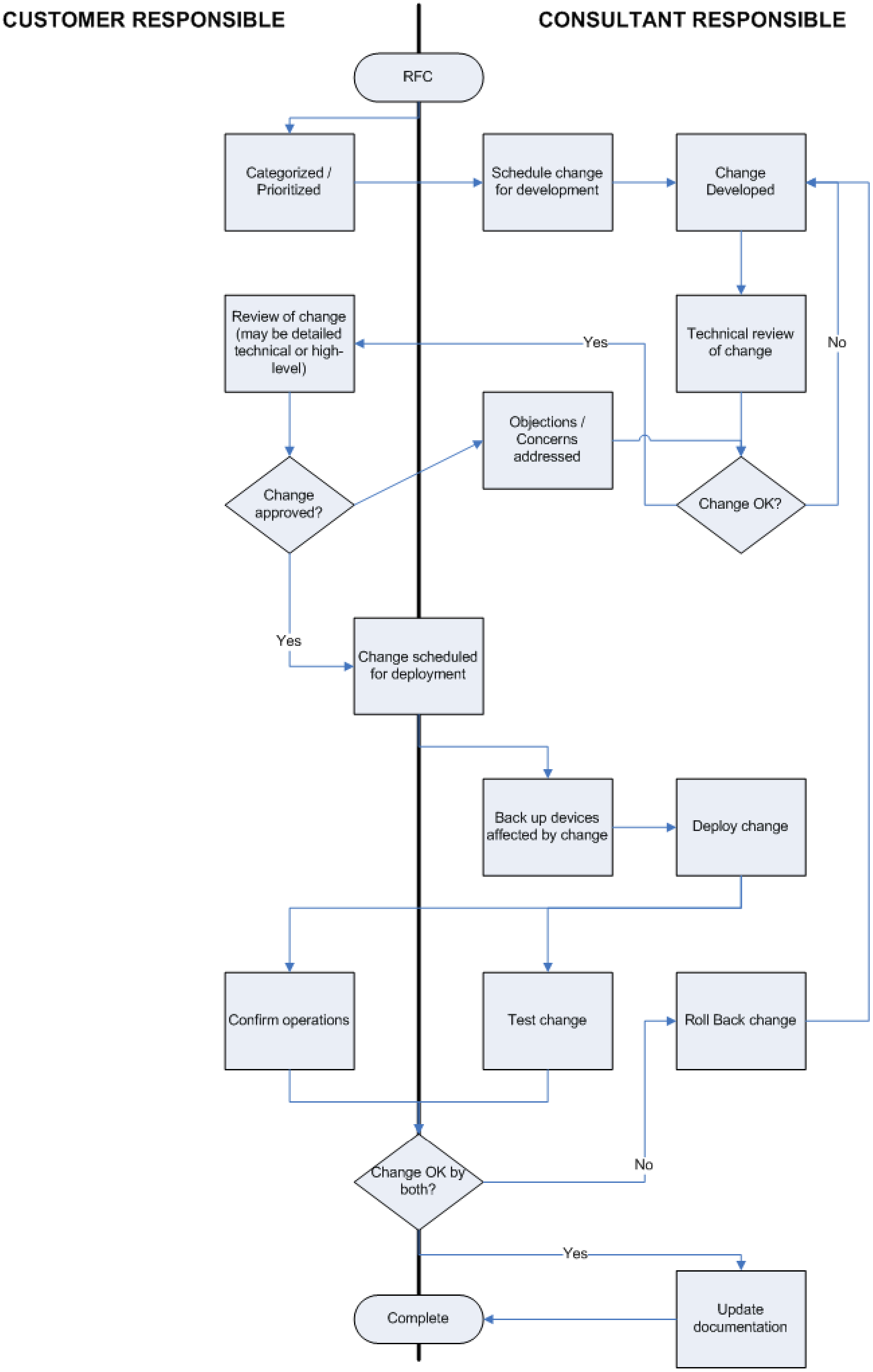
CUSTOMER RESPONSIBLE
CONSULTANT RESPONSIBLE
RFC
Categorized / Prioritized
Schedule change for development
Change Developed
Review of change (may be detailed technical or high-level)
Yes
Technical review of change
No
Objections / Concerns addressed
Change approved?
Change OK?
Yes
Change scheduled for deployment
Back up devices affected by change
Deploy change
Confirm operations
Test change
Roll Back change
No
Change OK by both?
Yes
Complete
Update documentation

Figure 8.8: A dual-responsibility process in which the customer only exercises oversight.

Let's explore how this process works:

1. RFCs come from both sides of the relationship. The customer should still have the opportunity to categorize and prioritize them, as it is the customer's business. Figure 8.8 doesn't show a logging responsibility, but it's likely that the consultant will be the one logging the RFCs and maintaining all other information and data used within the process.
2. The consultant gets to schedule the change's development cycle, keeping in mind the customer's priorities.
3. The consultant develops the change and technically reviews it.
4. Once the change is technically ready, the customer has a review opportunity. This review might be a high-level review that examines the change's potential impact, or it might be a second technical review, depending on the customer's involvement and level of technical expertise. The level of the customer's review should be an agreed-upon, stable component of the business relationship between the two organizations; in other words, the customer should *always* conduct the same type of review.
5. Any objections to the change are addressed by the consultant and, if necessary, the change is redeveloped to accommodate the customer's concerns. Another technical review should always occur on redeveloped changes, and the customer should get another look to ensure that the original concerns were properly addressed.
6. Once the change is ready, the customer and consultant jointly schedule it for deployment.
7. The consultant takes responsibility for backing up devices that will be affected and for deploying the change.
8. The consultant tests the deployed change, and the customer must confirm that the change and the entire network are operating as desired. Once the change is approved by both organizations, documentation is updated and the change is complete. However, if either party is unsatisfied with the results of the change, the change should be rolled back and redeveloped. Changes should *not* be "tweaked" in the live production environment to address technical or business concerns!

This process provides the customer with less responsibility but enables the customer to maintain oversight capacity to ensure that the network continues to meet the customer's business requirements. The key elements of a good configuration and change management process are present: peer review, disaster recovery plans, change prioritization, and testing.

Effective communications are just as important in this process. Whenever the process crosses the vertical line separating customer and consultant responsibilities, communications must be clear, and should be logged in a formal record keeping system, such as a ticket tracking database.

Change Management for Larger Organizations

Larger organizations, with their greater rate and scope of change, need a more complex, comprehensive process to ensure that changes can be evaluated and controlled centrally. Otherwise, the organization's changes will quickly get out of hand. The need for peer review and testing of changes is also increased, as the impact of an improperly made change is much greater than in small organizations that have, if nothing else, fewer users who will be affected. Figures 8.9 and 8.10 provide a split view of a process that addresses the needs of a larger organization. This process is based on ITIL best practices.

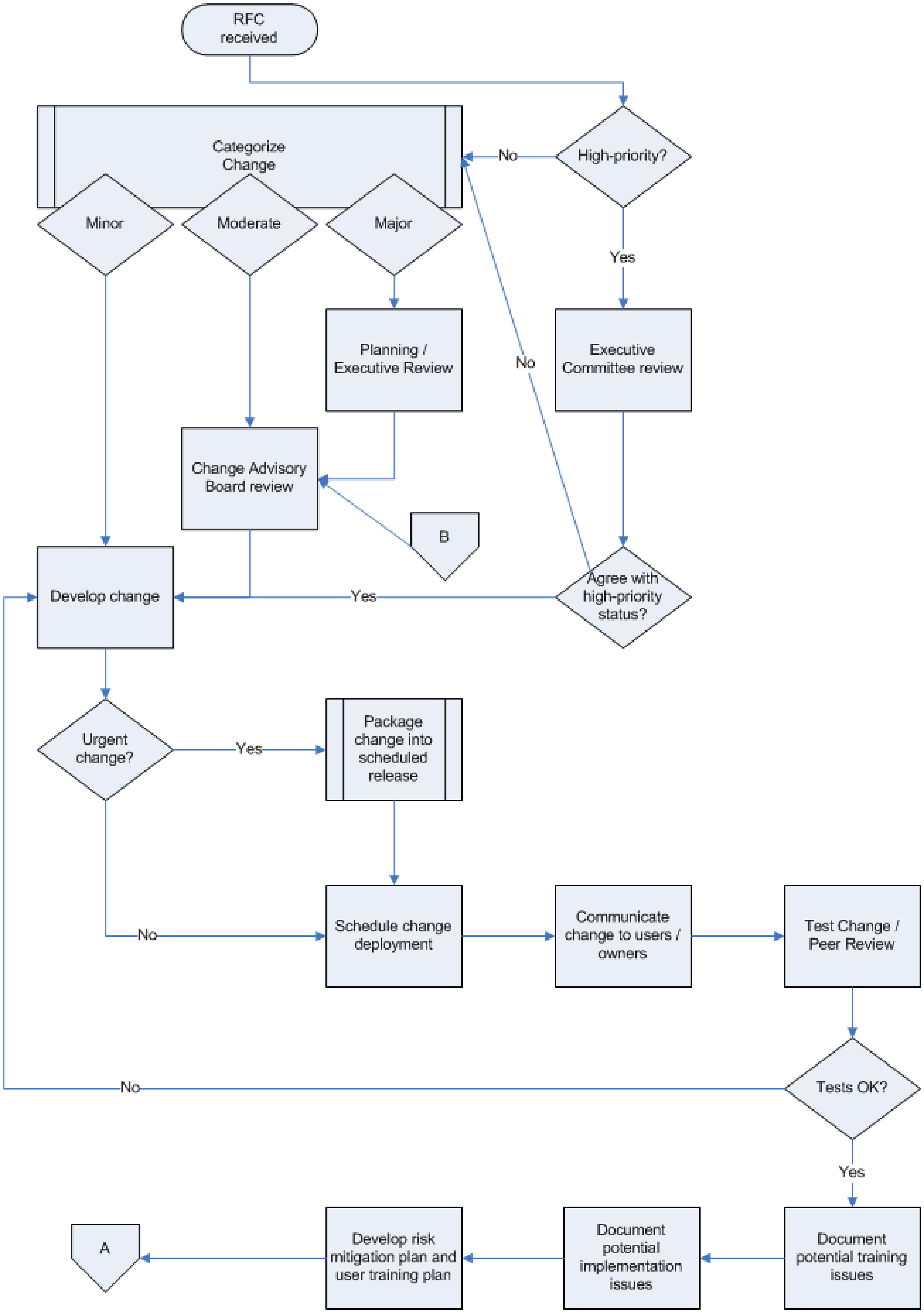
RFC received
High-priority?
No
Categorize Change
Minor
Moderate
Major
Yes
Planning / Executive Review
No
Executive Committee review
Change Advisory Board review
B
Develop change
Yes
Agree with high-priority status?
Urgent change?
Yes
Package change into scheduled release
No
Schedule change deployment
Communicate change to users / owners
Test Change / Peer Review
No
Tests OK?
Yes
A
Develop risk mitigation plan and user training plan
Document potential implementation issues
Document potential training issues

Figure 8.9: Change and configuration management for large organizations (part 1 or 2).

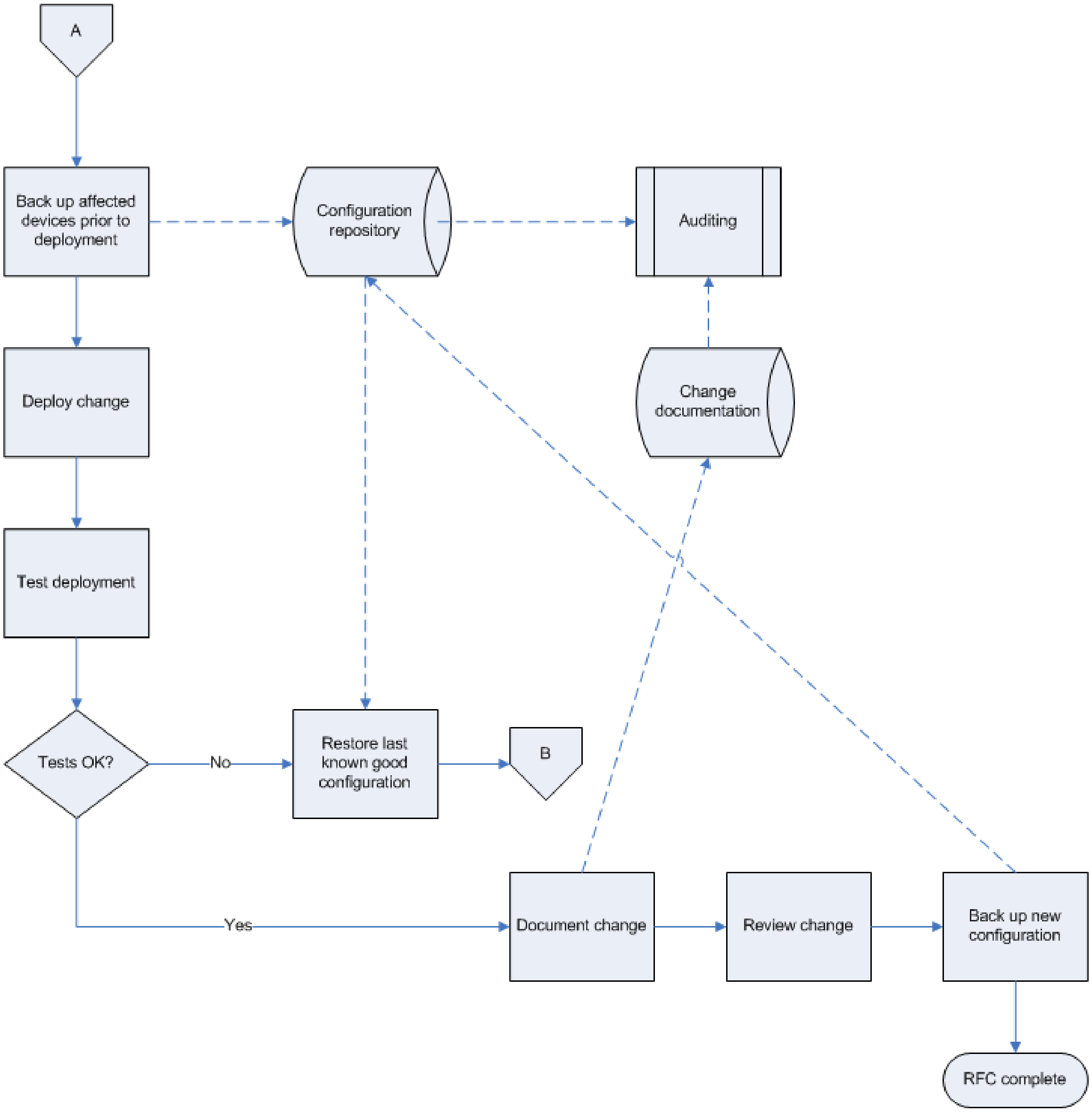

Figure 8.10: Change and configuration management for large organizations (part 2 of 2).

Let's walk through how this process works:

1. RFCs are categorized upon receipt into minor, moderate, or major-grade changes. This categorization allows the review process to be customized. Minor changes involving less risk can be built immediately; moderate changes are scheduled for review by the CAB; major changes are sent through a planning step with the executive committee before being sent to the CAB.

> 🕮 The CAB, executive committee, and other ITIL-related terms are defined and explained in Chapter 7.

2. For high-priority changes, a path is provided that bypasses the regular categorization process and provides for an immediate review by the executive committee. The committee's decision on priority either sends the change through the regular process (if the committee determines that the change is not high-priority), while genuinely urgent changes are sent directly for development.
3. Once developed, urgent changes are scheduled for deployment, while less urgent changes are scheduled for deployment in a package of changes. Although not all organizations choose to use the release-packaging technique recommended in ITIL, changes need to at least be scheduled in a sensible fashion. For example, you wouldn't want to roll out major configuration changes to a set of devices at the same time that you're rolling out a firmware upgrade.
4. The process then requires upcoming changes to be communicated to users and/or process owners so that they are aware of an upcoming change.
5. The change is tested and peer reviewed. If it fails testing, it's sent back for further development. If the change passes its testing and review, it moves into a documentation phase.
6. Any training issues are documented, along with any potential implementation issues. For a minor change, there probably won't be any issues; major changes will almost definitely involve potential implementation risks that should be understood.
7. A risk mitigation plan—what to do if something goes wrong during implementation—along with a training plan are developed. These are used to make the final preparations for the change.
8. The process moves to Figure 8.10 at this point, and picks up with a backup of all devices that will be affected by the change.
9. The change is deployed and immediately tested. As in prior processes, a binary decision to keep or roll back the change is made based on the results of the testing. There should be no decision to modify the change at this point.
10. If the change is retained, it is documented and reviewed. Affected devices are backed up. The documentation generated at this point is used in the organization's change and configuration management auditing processes, which ensure that the overall process is being followed for all changes.

11. If the change is not retained, devices are rolled back to their last-known-good configuration. Moving back to Figure 8.9, you see that the change is not sent back for further development, but rather to the CAB. The CAB needs to review the reasons for the change's failure, and either decide to redevelop the change, re-categorize it, or take some other action (such as deciding to deploy the change by itself rather than in a package or to bring in additional resources to help deploy the change successfully). Feeding a failed change back to the CAB ensures that the change is handled properly and that the CAB has information that may contribute to other pending changes.

Once again, the major facets of a robust change and configuration management process are present, including testing and peer review, categorization of changes, rollback capability, and so forth. Additional steps are provided to help ensure that changes from across the enterprise are brought into centralized control, ensuring that changes receive sufficient review and management attention to be successful, and that all changes align with larger business and operational needs.

Security-Centric Processes

With the current increased awareness of IT security issues, it is useful to consider sample support processes that focus on security. For example, consider the process that Figure 8.11 shows. This process focuses heavily on the design and development portion of change management and can, in fact, be incorporated into almost any of the processes that have already been covered in this chapter. The idea is that security is maintained through a well thought-out design, often in the form of a template, which creates a carefully-designed security baseline.

> ✎ A *template* is a predesigned configuration file that is created according to your company's standards. Each device's configuration should start with the appropriate template and be filled in by using values specific and unique to that device (IP address, name, and so forth).

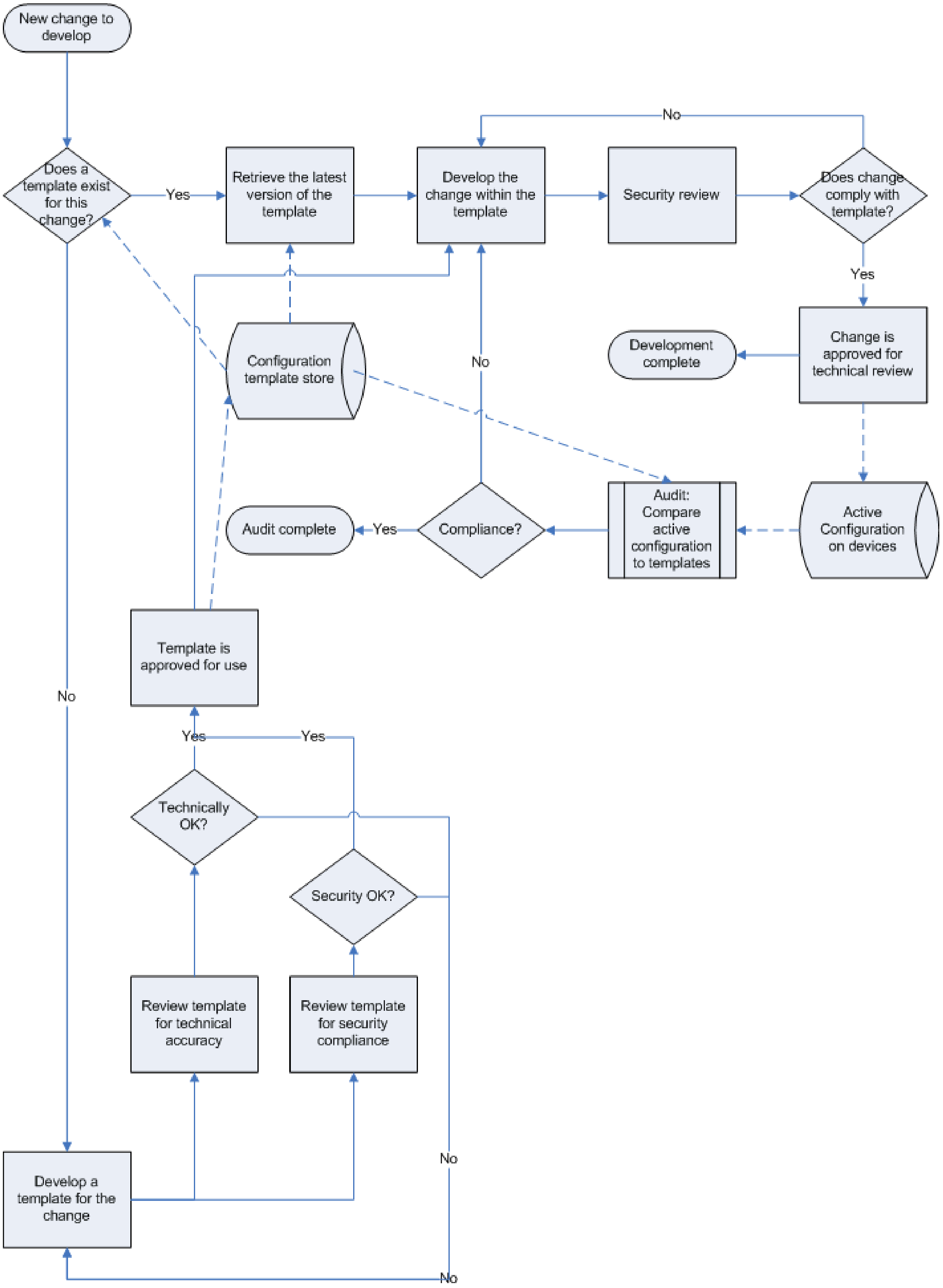
New change to develop
Does a template exist for this change?
Yes
Retrieve the latest version of the template
Develop the change within the template
Security review
Does change comply with template?
No
Yes
Change is approved for technical review
Development complete
Configuration template store
No
Active Configuration on devices
Audit: Compare active configuration to templates
Compliance?
Yes
Audit complete
Template is approved for use
No
Yes
Yes
Technically OK?
Security OK?
Review template for technical accuracy
Review template for security compliance
No
Develop a template for the change
No

Figure 8.11: Change development process incorporating security best practices.

The idea behind this process is that individual security reviews of each and every change will become tedious and are error-prone. Instead, develop a security-compliant template for changes, then audit changes to ensure that they adhere to the template—this process is more efficient. In fact, complete change and configuration management solutions will provide built-in template repositories and provide the ability to develop changes in those templates. The following steps explore how the process in Figure 8.11 works:

1. When a new change needs to be developed, check for a template. For example, if you've just implemented a new router in your organization, you might not yet have templates that determine how the device should be configured (SNMP community strings, access permissions, and so forth).
2. If a template does exist:
    a. Retrieve the template.
    b. Develop the change within the template. The template should have an area that cannot be changed (such as access permissions settings) and areas that can (such as per-device IP addresses and so forth).
    c. A brief security review ensures that the template hasn't been modified and that the change was developed within the scope of the template.

☞ Don't use templates that don't provide needed functionality; a new template, which *does* provide the necessary configuration capabilities, can be developed.

    d. Once the change is reviewed, it is approved for the next step in the overall process, which is generally a technical review.
3. If a template does not exist:
    a. A new template is developed. Typically, existing templates will provide a starting point for the development of a new template, especially for configuration parameters that are security-related.
    b. The new template should pass both a security and a technical review. Any changes to the template based on these reviews must be resubmitted to *both* reviews so that both the technical and security reviews are considering the template that will actually be used in the production environment.
    c. Once approved, the template is stored in a central repository and used to kick off the development of the change that was originally requested.

Notice that this process also provides for an audit, which tests active device configurations for compliance with templates. Some change and configuration management solutions can actually conduct this review automatically, retrieving device configurations and comparing them with an approved template. Devices whose active configurations don't match the approved templates are flagged with an alert, and the process then calls for those devices' configurations to be redeveloped according to the approved templates.

✎ The entire process that Figure 8.11 shows is intended to "snap in" to the "Develop Change" task shown in most of the other processes. This process is simply an expansion of that one task into a more formal process.

A major security and operational benefit of template-based configuration is consistency. Device configuration options that aren't changed frequently (such as permissions) aren't always known to every member of the IT staff; by requiring device configurations to be template-based, these critical areas of the configuration can be determined ahead of time and more easily enforced in new and updated device configurations. Administrators no longer need to remember the correct security permissions for a device; the template simply has them built-in so that they're always applied correctly.

Figure 8.12 shows a sample audit process. This process is intended to be conducted apart from the primary change and configuration management process, and serves as a form of double-check to ensure that the process is being followed.

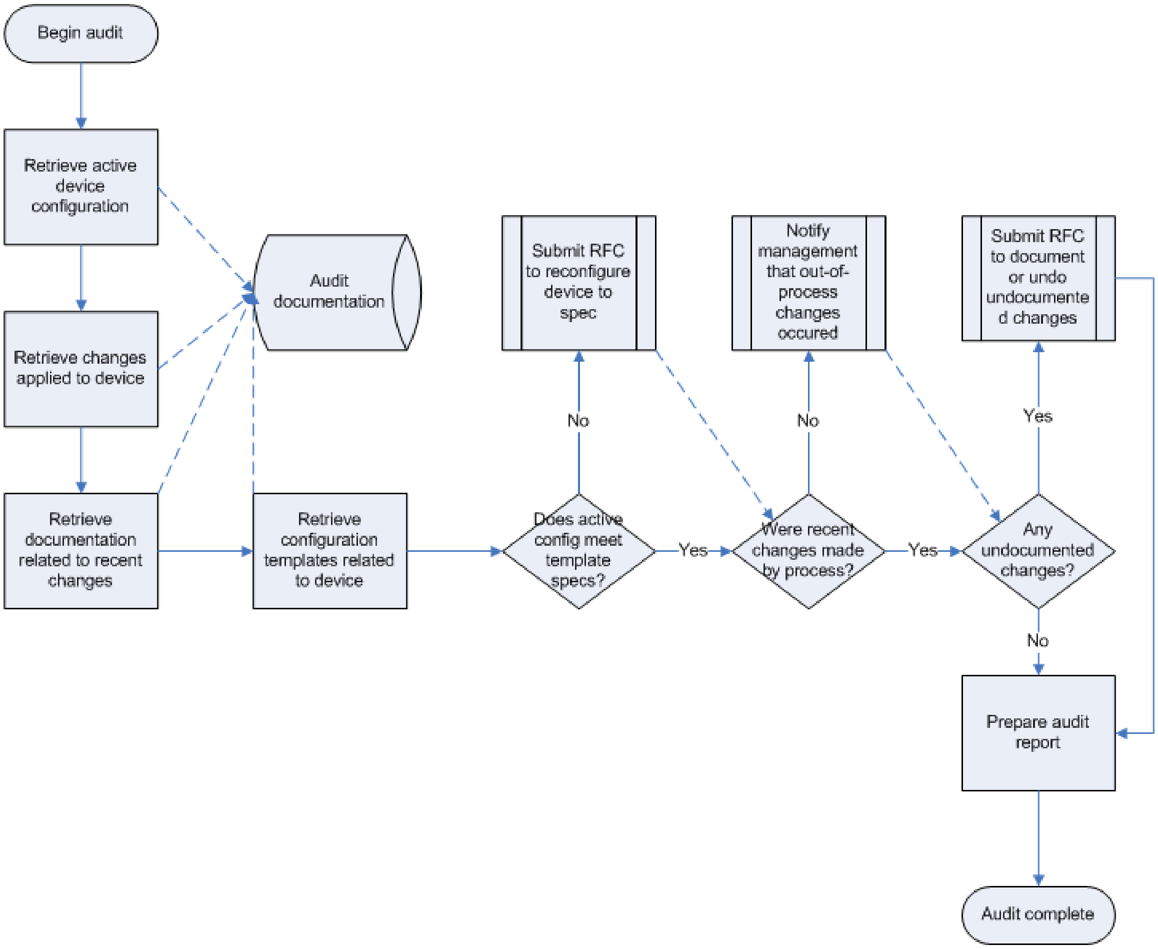
Begin audit
Retrieve active device configuration
Retrieve changes applied to device
Retrieve documentation related to recent changes
Audit documentation
Retrieve configuration templates related to device
Submit RFC to reconfigure device to spec
No
Does active config meet template specs?
Yes
Notify management that out-of-process changes occured
No
Were recent changes made by process?
Yes
Submit RFC to document or undo undocumented changes
Yes
Any undocumented changes?
No
Prepare audit report
Audit complete

Figure 8.12: Sample audit process.

Let's look at what is going on in Figure 8.12:

1. A device is selected for auditing.
2. The device's active configuration, recent changes, change documentation, and applicable templates are retrieved.
3. The active configuration is compared with the applicable templates. If the device does not comply, an RFC is created to bring the device into compliance. Some change management solutions can automatically alert you to out-of-compliance devices under certain circumstances; these alerts should trigger a full audit.
4. Recent changes are examined to ensure that they followed the formal change and configuration management process. Management is informed of any changes that were made outside the process.
5. Any undocumented changes are noted, and an RFC is created to either document the changes or remove them from the active configuration. Again, some change management solutions can alert you automatically to changes made outside the solution and the process it enforces; such alerts should trigger an immediate and full audit of the affected device because undocumented changes represent the biggest single security and operational threat to network devices.
6. Upon completion of the audit, a report is prepared detailing the audit results. All information used in the audit, such as the active device configurations, must be maintained along with the audit report. This information maintenance ensures that the audit can be repeated using this "point in time" information, and that the same audit results can be achieved.

Summary

This chapter presents several complete change management processes, and you should be able to find at least one that fits your organization's size and specific business needs. Throughout this guide, we've explored the concepts and practices behind configuration and change management, including their impact on network stability, overall security in your environment, and more. I've covered the scope of change management and shown you what parts of your environment you can realistically expect to manage through a single solution. I've covered the technologies that make configuration and change management possible, including foundational technologies such as SNMP, Syslog, RADIUS, and more.

In addition, I gave you practical advice for selecting a change management solution and building a toolset that will meet your organization's change management needs. Finally, I introduced you to the industry's best practices for change and configuration management, as outlined in the ITIL.

This guide is meant to give you a start on implementing better change and configuration management for your network. As you move forward with your management practices, I think you'll realize quick benefits in terms of reduced downtime, less stressful maintenance, and an ability to better meet the operational goals of your network.