

Realtime
publishers

Creating Unified IT Monitoring and Management in Your Environment

Don Jones

sponsored by



Chapter 3: Connecting *Everyone* to the IT Management Loop 29

- Starting the Loop: Connecting Monitoring to the Service Desk 30
- Making Changes: How to Find a Change Management Window 35
- Communicating: How to Bring Users into the Loop 37
- SLAs: Setting and Meeting Realistic Expectations..... 39
- Tell Me What You *Really* Think..... 41
- When Everyone Doesn't *Need* to See Everything: A Multi-Tenant Approach 42
- Call It a Private Management Cloud: Allocating Costs 43
- Conclusion 44
- Coming Up Next..... 44

Copyright Statement

© 2011 Realtime Publishers. All rights reserved. This site contains materials that have been created, developed, or commissioned by, and published with the permission of, Realtime Publishers (the “Materials”) and this site and any such Materials are protected by international copyright and trademark laws.

THE MATERIALS ARE PROVIDED “AS IS” WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, TITLE AND NON-INFRINGEMENT. The Materials are subject to change without notice and do not represent a commitment on the part of Realtime Publishers its web site sponsors. In no event shall Realtime Publishers or its web site sponsors be held liable for technical or editorial errors or omissions contained in the Materials, including without limitation, for any direct, indirect, incidental, special, exemplary or consequential damages whatsoever resulting from the use of any information contained in the Materials.

The Materials (including but not limited to the text, images, audio, and/or video) may not be copied, reproduced, republished, uploaded, posted, transmitted, or distributed in any way, in whole or in part, except that one copy may be downloaded for your personal, non-commercial use on a single computer. In connection with such use, you may not modify or obscure any copyright or other proprietary notice.

The Materials may contain trademarks, services marks and logos that are the property of third parties. You are not permitted to use these trademarks, services marks or logos without prior written consent of such third parties.

Realtime Publishers and the Realtime Publishers logo are registered in the US Patent & Trademark Office. All other product or service names are the property of their respective owners.

If you have any questions about these terms, or if you would like information about licensing materials from Realtime Publishers, please contact us via e-mail at info@realtimepublishers.com.

[**Editor's Note:** This book was downloaded from Realtime Nexus—The Digital Library for IT Professionals. All leading technology books from Realtime Publishers can be found at <http://nexus.realtimepublishers.com>.]

Chapter 3: Connecting *Everyone* to the IT Management Loop

IT management has for too long involved discrete, disconnected processes that often leave key participants wondering what's going on. Bringing everyone—users, managers, IT professionals, and more—into the loop can create significant benefits as well as reduce the tendency to fall back into discipline-based silos. This is where the integration between monitoring and service desk truly happens, and these concepts deliver the most critical, central themes discussed throughout this book. It's all about communication—ways to better achieve communication as well as create opportunities for continuous improvement.

Users sometimes perceive their IT department as out-of-touch, ivory-tower geeks with poor people skills. Whether or not that's true depends on the actual IT team members, but the *perception*, fair or not, often exists. That's because IT can too often be the last ones to know about things that *users* perceive as problems. Sure, the server might be humming along within specs, but the order-entry application is incredibly slow. IT *says* that email is working fine, but I've been waiting on an incoming purchase order for an hour—the email system can't possibly be working correctly!

IT has its own unique problems to deal with, and they sometimes involve a disconnect with management. Finding windows in which to make approved changes, for example, can be incredibly tricky. Simply coordinating the changes that are proposed, approved, under development, ready for implementation, and so forth can be difficult. Many organizations have adopted change management frameworks, such as those proposed by ITIL, that outline specific processes for reviewing and approving changes. Physically *coordinating* that process, however, can seem like herding cats. It's even worse when IT has been divided into silos: The database team might have a change scheduled for tonight, but that change is going to conflict with the power supply changes being implemented by the data center team. We need to get everyone on the same page.

Starting the Loop: Connecting Monitoring to the Service Desk

Most organizations today have a ticket-based system for coordinating IT activities. These organizations also usually have monitoring systems in place to watch their IT systems and alert them to any problems. Too few organizations, however, have *connected* these two systems. Ideally, that's what you want: A single, integrated IT management system that can detect problems and then automatically open tickets for the appropriate individuals. If the email server is down, the appropriate administrator should get a ticket. Those tickets, of course, should include notifications via text message, email, or whatever other medium is appropriate so that alerted individuals *know* they have an alert.

That auto-assignment—you might even choose to call it *auto-routing*—of tickets needs to be pretty intelligent. Different systems, in different locations, at different times, all might change how the ticket is created, thus changing who is assigned to work the problem. Tickets should be as complete as possible, meaning as many fields as possible should be automatically populated—you shouldn't have to rely on a Help desk, or someone else, to fill in the details. Those details might include the affected server's information. Figure 3.1 shows what this kind of auto-generated ticket might look like, with several key bits of information pre-populated by the system.

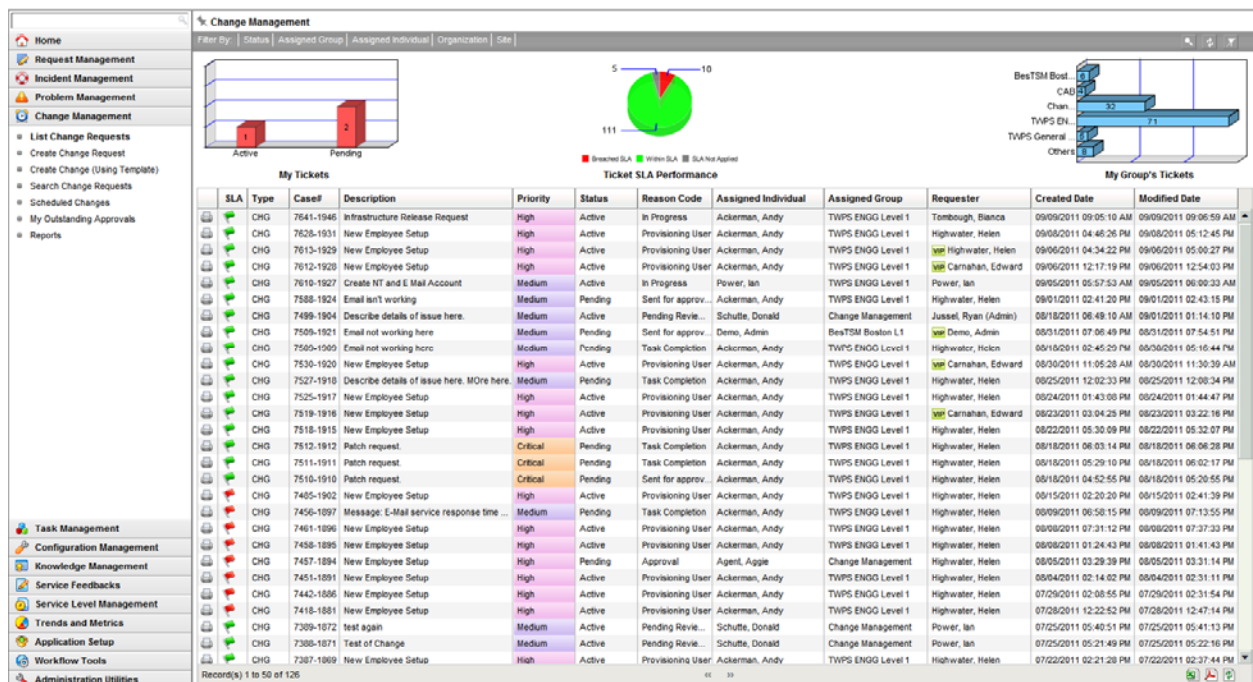


Figure 3.1: Automatically-generated tickets in response to alarms.

The idea is to have a service desk solution—that’s the software that helps coordinate and manage IT activities, often through tickets—working with the monitoring solution, thus creating a truly integrated response to IT problems.

This is all intended to provide specific benefits. First and foremost is faster problem resolution. By not waiting for users to inform you of a problem, you’re getting started on solving the problem faster. By having pre-populated tickets, the IT team is able to work more quickly because they’re starting with more information.

There’s a bit more depth that can be added, if you have the right service desk software in place. Frameworks like ITIL encourage *root cause analysis*, meaning your team should focus not only on solving today’s specific problem but also on making the overall environment more stable and problem-resistant. To that end, a service desk solution can define two types of problems: global issues and specific incidents.

Specific incidents might be day-to-day problems like, “Email moving slowly throughout the organization,” “Order entry application operating slowly,” and so forth. Those might all be tied to a global issue of “Unexplained network slowdowns,” which could be examined and solved—perhaps locating a router that was overheating and dropping more packets than usual.

Sometimes, specific incidents might not be entirely solved until the overarching global issue is solved. By tracking those individual incidents along with the global issue, you can help keep your users and managers more informed. For example, once that overheating router is discovered and replaced, everyone affected by an associated specific issue could be notified: “Hey, we think we’ve found the root cause for all the slowdowns, so things should be better from here on out.” Figure 3.2 shows how a single global problem can be attached to multiple incidents.

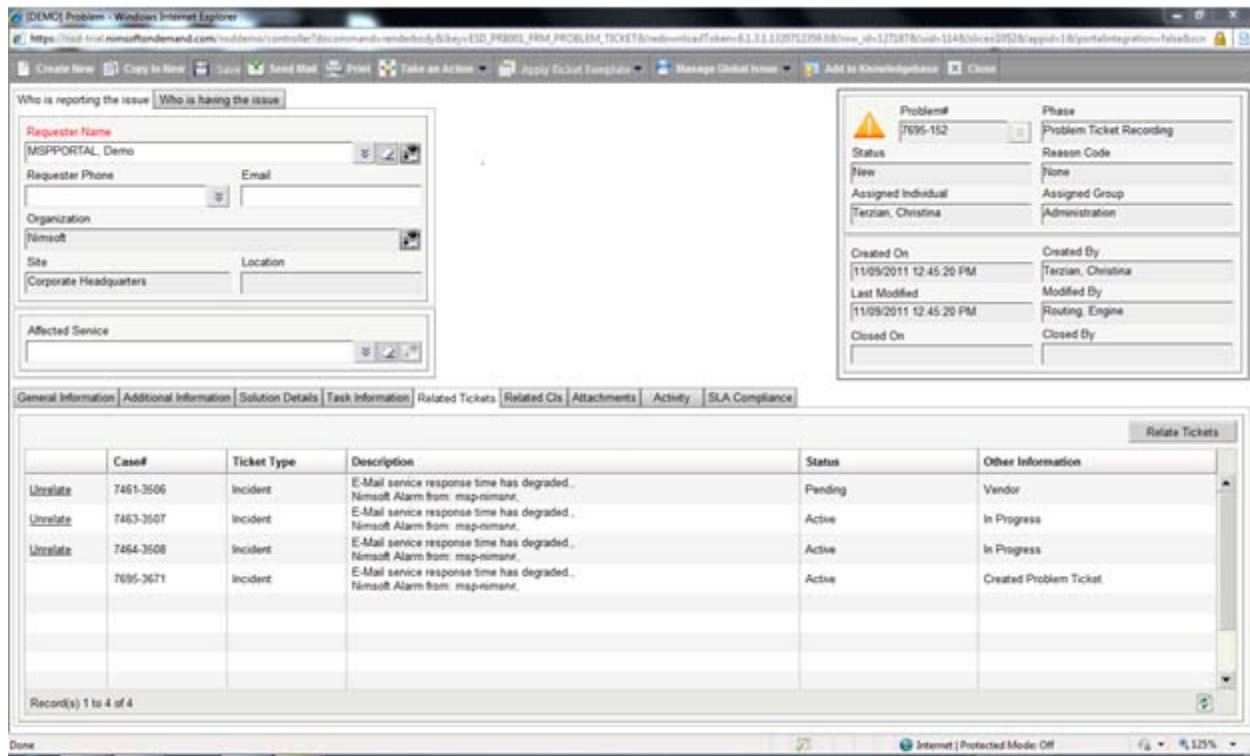


Figure 3.2: Relating multiple incidents to a single problem.

I've used a couple of keywords in the forgoing discussion and want to take a moment to specifically define them in the context of this book:

- An *incident* is something that happens in the environment, such as a failed server or a slow application.
- IT staff create *problem* records to help manage the incident. Problems may in fact be associated with multiple incidents, as in the case of that overheating router, which caused multiple disparate failures throughout the environment.

I'm going to start using those two terms more consistently from here on. Hopefully, some of the benefits of combining monitoring with problem solving will become clear. For example, more simplistic Help desk solutions allow multiple tickets to be opened against what is essentially the exact same issue. That can result in a lot of duplicated effort, as multiple IT team members attempt to work the issues on their own. It can also result in a lot of paperwork because solving the root cause then requires technicians to spend time laboriously closing each ticket. With a more sophisticated system in place, everything can be consolidated into a single, managed problem record. Doing so creates additional benefits, such as identifying solutions or workarounds, which I'll discuss in upcoming chapters.

Problems and incidents, however, aren't the only reason that users interact with IT. Hopefully, they're not even the *major* reason your users interact with IT! Aside from reporting incidents, users also need to request routine services: advice, new hardware requests, routine change requests, access requests, and so forth. These interactions should be managed through a more formal workflow in which users submit their request, have it assigned to the appropriate technician after being approved, and be able to track the status of their request.

For example:

1. A user might visit a Web site to browse a "catalog" of items they can request, such as access to systems, changes to hardware, and so forth.
2. A user selects an item from the catalog, and provides whatever details are necessary to complete the request.
3. A ticket is created in the service desk that represents the user's request. Depending upon the request, the ticket might first be routed to the user's manager for approval.
4. Once approved, the ticket would be automatically routed to the appropriate technician or IT team for completion.
5. The user would receive status updates, perhaps via email, throughout this process, keeping them informed of its progress. The status updates would include a "completed" update once the request was finished.

By using the same ticket-based system employed for problem-solving to address routine requests, IT technicians can rely on a single interface to manage their workload. Figure 3.3 shows what a routine request ticket might look like.

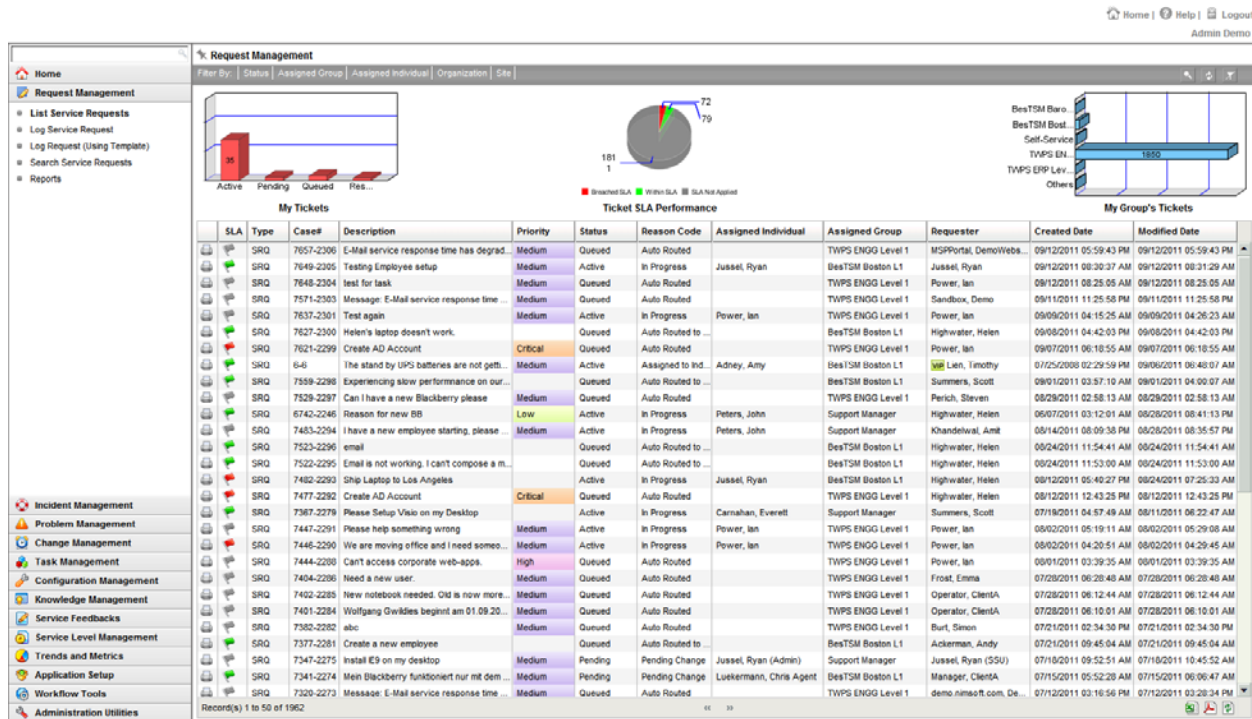


Figure 3.3: Routine requests can also be made into tickets.

Even better, IT management can rely on all IT work being documented and tracked in a single system, enabling management to stay informed through reports, dashboards, and other mechanisms. Figure 3.4 shows an example of what such a report might look like.

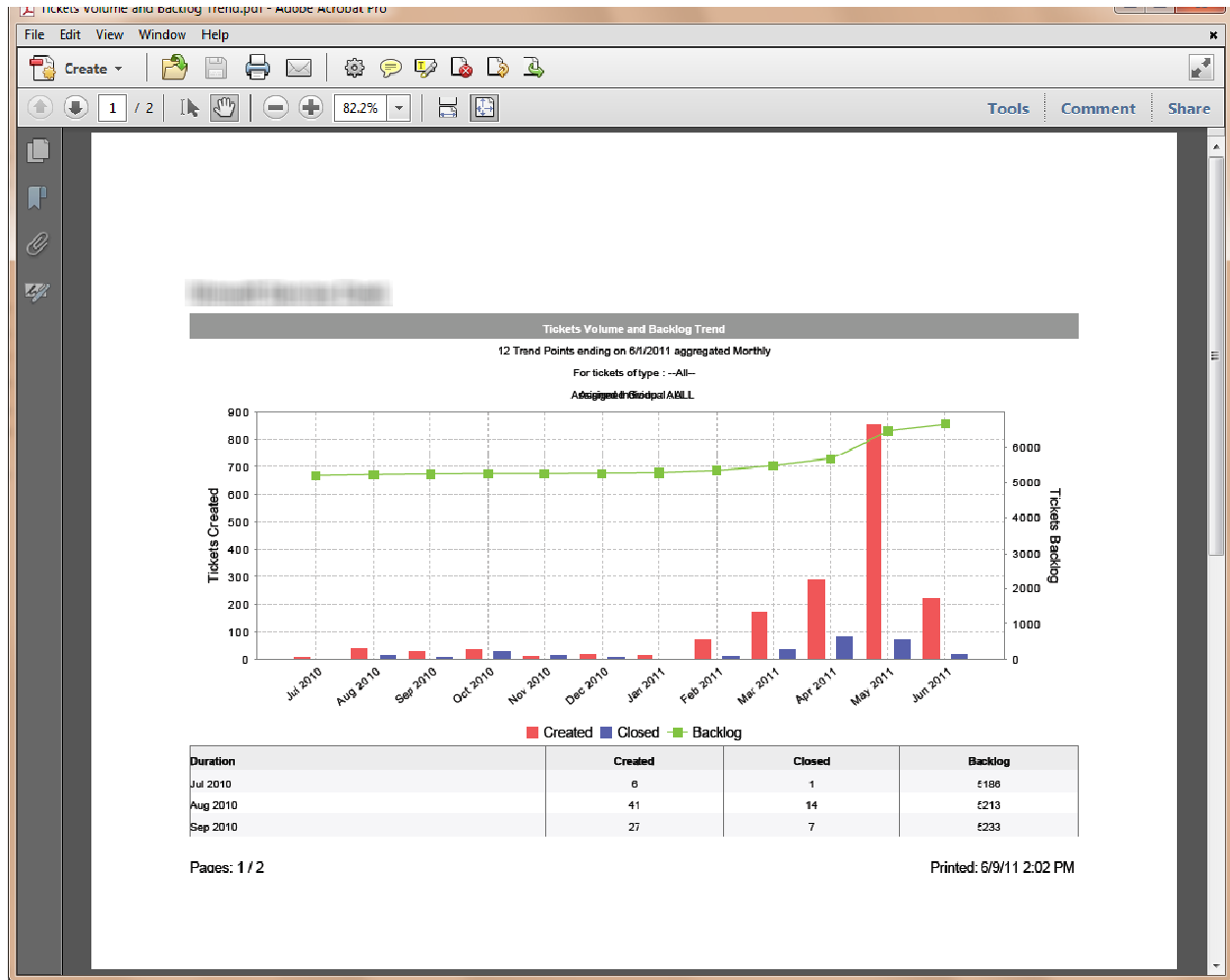


Figure 3.4: Management reports become more effective when they include all IT workload.

The idea is to keep *everyone* in the loop: users remain informed, IT remains informed, management remains informed. Much of the burden of keeping everyone informed is handled by the software, which can send email updates and other kinds of notifications so that everyone is aware of what’s happening at all times.

Making Changes: How to Find a Change Management Window

Large, multi-discipline IT departments have inherent problems. In the previous chapter, I discussed the problem of silo-based problem solving, where domain experts spend time passing a problem back and forth because everyone is looking at different tools and data to determine whether the problem is “theirs.” We’re certainly not going to get rid of domain experts, so the solution is to get tools that could put *everything* into a single console in order to unify everyone’s efforts.

Another problem created by those silos relates to change management. At the start of this chapter, I outlined one of those problems: The database team is ready to implement a change, but it's going to be in conflict with a change being implemented by another group. Managing change windows is becoming increasingly difficult. Not only are applications and services needed round-the-clock, creating tiny change windows in the first place, but the varying needs of different experts creates contention for those already-small windows. "Boss, we'd have that fix in place, but we can only implement it at night. It's going to take 4 hours, which just fits inside the window management allows us. But all this week, other teams have been using the window, and the changes they're making are blocking us from doing anything at the same time." It's not an unusual situation. It gets tough for management to even track what changes are pending and to slot them into the shrinking time that's available to make them.

The lack of visibility into these windows, and the contention for them, makes it impossible to even make a management decision. For example, if management could *see* the number of changes stacked up, and *see* the contention, they might decide to expand the window for a period of time in order to get the changes implemented. They might not decide to do that, but they'd be *consciously making a decision* rather than remaining ignorant of the actual problem.

The solution, of course, is software that facilitates the coordination of departments. Think about it: If you're using a service desk solution to track tickets, then tickets can be created for proposed changes. Those tickets would be assigned to a technician, routed for reviews and approvals, and so forth, all via some workflow you designed. That's an excellent way to support ITIL processes, by the way. The tickets themselves can then feed a unified calendar, built right into the service desk, which allows change planners to schedule activities. They can see agreed maintenance windows, manage contention between conflicting changes, and so forth. By getting this information into a familiar calendar form, they can also make decisions about whether to widen maintenance windows if doing so is necessary and beneficial to the organization. Figure 3.5 shows a change management calendar.

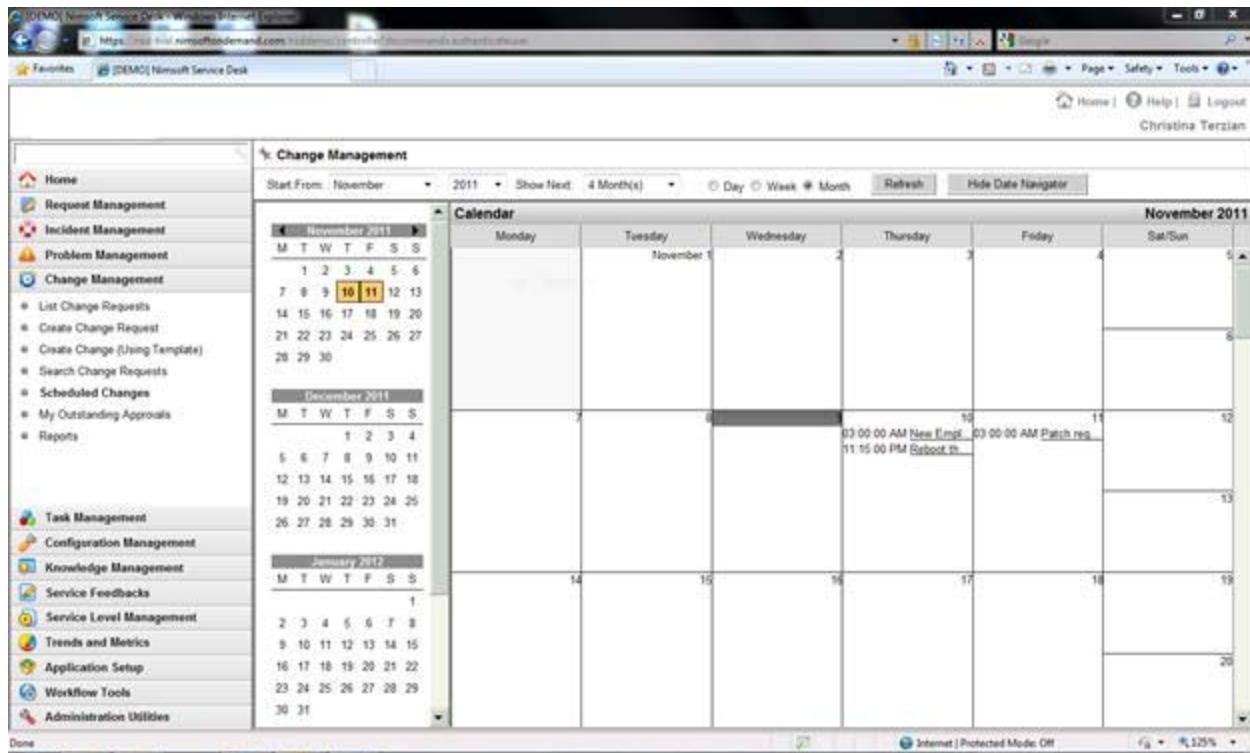


Figure 3.5: Managing change schedules in a calendar view.

This is just another way to help keep everyone in the loop. Management now has a clear visual depiction of change and schedule contention. Such a calendar could even be made available to users so that they could see what changes were scheduled and plan their own activities accordingly.

Communicating: How to Bring Users into the Loop

The idea of keeping users informed certainly isn't new, but many organizations that have attempted to better engage their users haven't met with unqualified success. Too often, "keep users in the loop" solutions take the form of self-service Web portals, where users can log in to check the status of their tickets or to check the status of a particular service. That's all well and good, but Web portals like that don't always fall within the natural workflow of a user. For example, most users, when confronted with some kind of problem, don't necessarily think to check a Web site and see if something's wrong—they call the Help desk.

Users do, however, spend a lot of time in their email inbox. Why not make that your channel for communication? Organizations don't use this method of communication in part because doing so could easily become a time burden for your IT team. "So on top of solving the problem, I have to send out hourly update emails with the status of the problem?" Sounds like a Dilbert cartoon!

In reality, a good service desk solution can do it for you. Sending an email update when a user's ticket is updated, for example, is an easy operation for a piece of software. Such emails can be informative, and help users feel comfortable that their request is being handled. Figure 3.6 shows what one might look like.

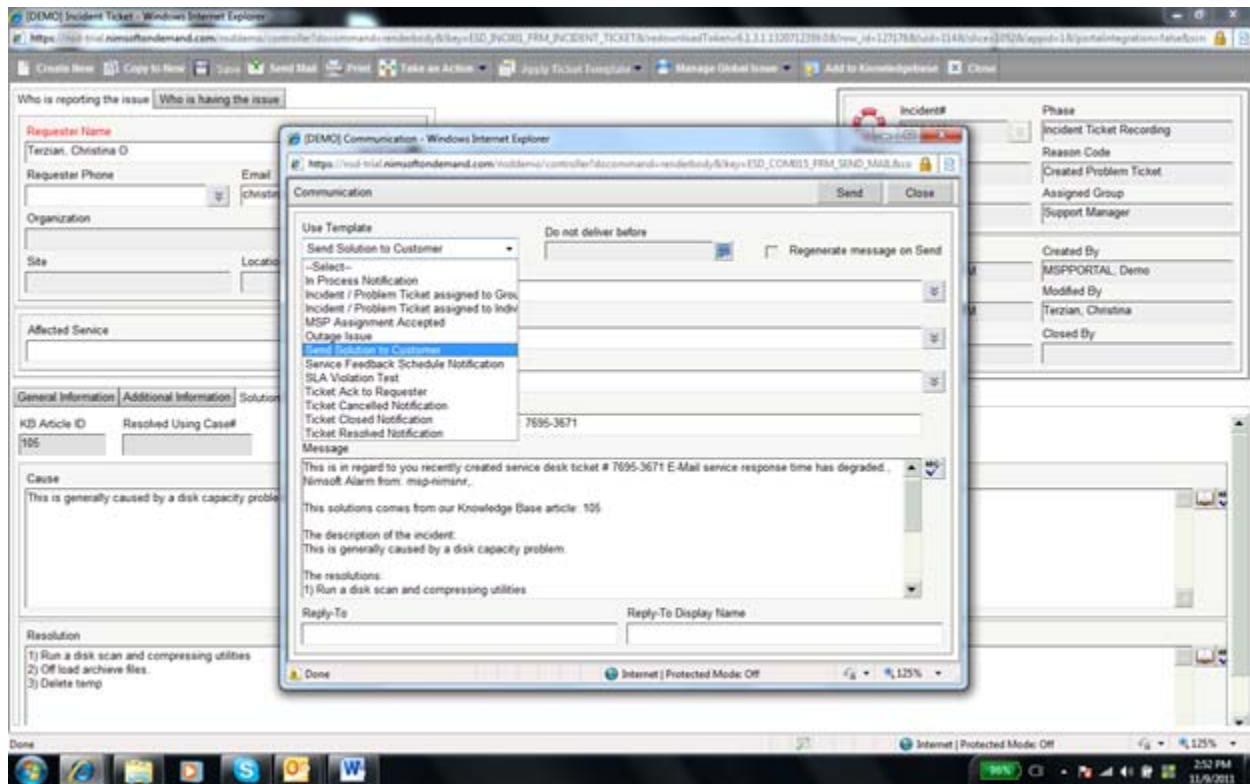


Figure 3.6: Keeping users informed with detailed emails.

What's more compelling is a service desk solution that can actually *accept requests via email* rather than expecting users to go to a self-service Web portal and open a ticket. Face it: Your users are more likely to pick up the phone than visit a Web site, unless you've placed significant artificial barriers in the way, like complex voice menus in the phone system. Users are *more* likely to send an email. If your service desk, rather than a human technician, can receive those emails and use them to create a ticket, you've truly created a system your users are likely to embrace. Such tickets could still be auto-assigned and – routed, helping the right technician to start working the problem more quickly.

Even for your users' routine, non-problem requests, email updates can be valuable. When their request is approved, rejected, underway, completed, and so forth, an email update helps keep users informed without additional human effort.

Note

I want to emphasize that self-service portals *are a good thing*. They can provide a rich user experience, help guide users to self-service solutions, and more. They just shouldn't be the *only* means of communicating with users.

SLAs: Setting and Meeting Realistic Expectations

Unless you've been living under a rock for the past decade or so, Service Level Agreements (SLAs) are probably pretty familiar to you. These are, in their simplest form, an agreement by the IT team to provide a specific level of performance or availability for a specific service or application. "The email service will be available 99.999% of the time on an annualized basis" is an example of a very simple SLA.

But SLAs can get complicated quickly. You can't just pull a number out of thin air; what level of service can you reasonably provide? What level of service have you historically provided, and is that meeting the business' needs? Once established, how do you track the SLA to make sure you're actually meeting it—and ideally get some kind of notification when you're in danger of breaking the agreement?

SLAs might not be the only type of agreement you need to define and track. Some organizations also use *underpinning contracts* (UCs) or *operational level agreements* (OLAs) for different in- and out-sourced services; these often support SLAs.

A well-built service desk and monitoring solution can help you handle these agreements more precisely. You'll start by defining top-level SLAs, then creating and managing UCs and OLAs as appropriate.

Once defined, the solution should be able to track ongoing performance and availability, perhaps offering a simple dashboard—like the one shown in Figure 3.7—that illustrates your compliance with your SLAs. You might also have more comprehensive and detailed reports on SLA metrics.

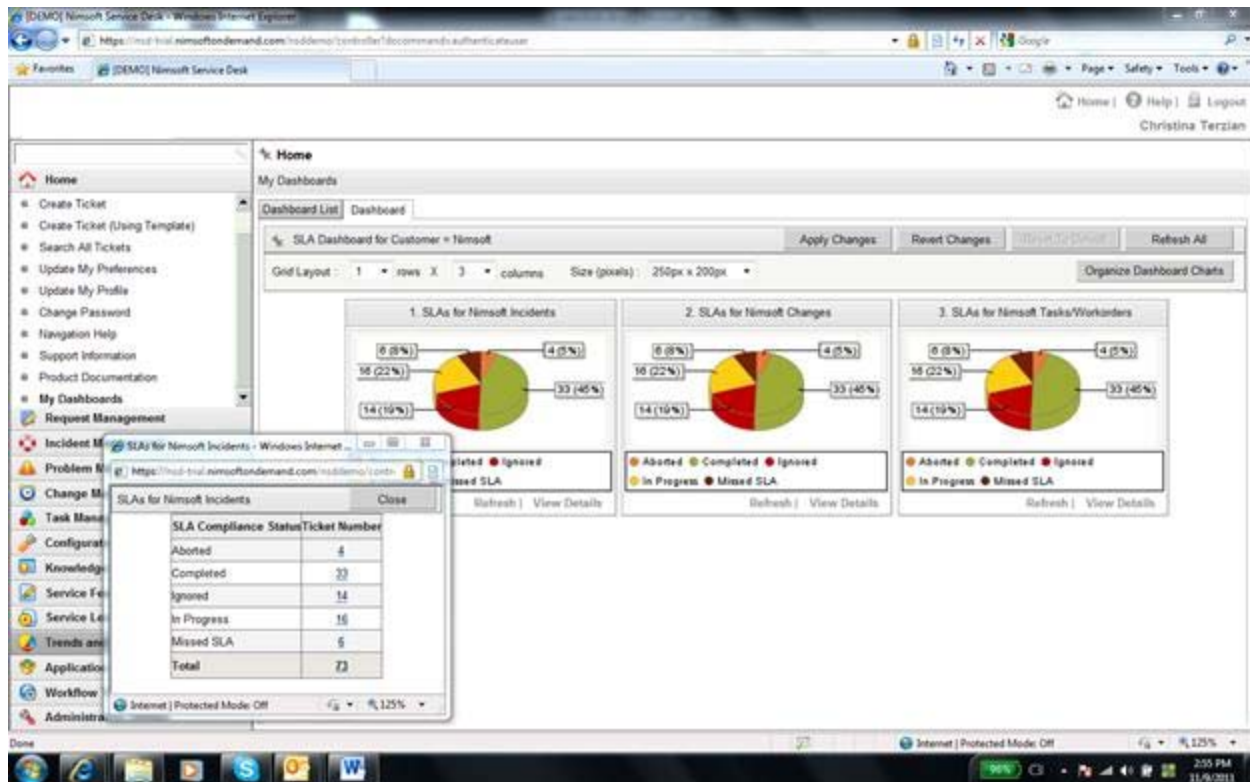


Figure 3.7: Managing SLAs with at-a-glance dashboards.

Most importantly, however, the solution needs to provide you with the ability to define rules for your SLAs so that tickets can be created—and auto-assigned to the appropriate technicians—when SLAs are in danger of being broken. Further, the solution should support escalation rules so that if an SLA that is in danger of being broken is not corrected within a certain amount of time, the solution can automatically call for backup, summoning additional technicians, notifying management, and so forth.

There's also a strong need to recognize that no SLA is perfect. Sometimes, for whatever reason, the business will decide to take a service offline. Perhaps it's for a software upgrade or for some kind of infrastructure maintenance. In those cases, you're not *breaking* the SLA; you're *agreeing*—along with whatever part of the business will be affected—to *temporarily suspend* the SLA to get the work done. A service desk solution should support these types of exceptions, including SLAs that are only valid during certain hours, holiday exceptions, agreed-upon reduced service windows, maintenance windows, and so forth.

The idea is to automate SLA definition and management—and to automate the notifications that go with SLAs. If an SLA is broken, you might agree that the affected business users will receive an automatic notification. That lets *them* know that *IT* knows about the problem and is working on it—without forcing users to visit a self-service portal and open a ticket. That kind of proactive response can go a long way toward improving IT-user relationships, and in helping IT be viewed as responsive to, and supportive of, business requirements.

Tell Me What You *Really* Think

IT managers like IT to think of users as “customers.” In some cases, your users might actually *be* customers, in the sense of “sending you a check for specific services you provide” customers. In other cases, your users might be internal users—but still “customers” in the sense that they consume services you, the IT department, provides, and that you get paid for your efforts.

A big problem that IT has always struggled with is its perception by its customers. Do customers think you’re doing a good job? What *is* a good job?

For this reason, monitoring End-User Experience (EUE) metrics, which I discussed in the first chapter, has become a hot trend in the IT industry. *You* might see that your servers’ performance is within norms, but by the time you throw in old client computers, routers, network cabling, and everything else involved in delivering a service to users, *they* have a completely different perception of the server’s performance. Measuring the EUE is a way to get some insight into that aggregate perception that your users—your *customers*, if you prefer—deal with.

Businesses have traditionally used another important technique to discover their customers’ perceptions: surveys. Phone your credit card company, and the robot who answers the phone might inform you that you’ve been selected to participate in a short satisfaction survey, which will begin when you finish speaking with the agent who is about to come on the line. Walk into a theme park, and a smiling employee with a tablet computer asks you a few questions. Look at the register receipt from your last purchase, and you might find that you’re eligible to win a gift card or other prize if you complete an online survey about your shopping experience.

Surveys are an effective way of finding out what users really think, and a good service desk application should provide you with the ability to survey your customers. Perhaps you want to ask them their opinion after each request that’s completed. Maybe you want to be a bit less intrusive, and only survey them after every 3 or 4 requests. Whatever you decide, a service desk solution should be able to automate the process. You might even want to engage customers in ad-hoc surveys to further your understanding of their perceptions about day-to-day performance, availability, service levels, and so forth.

Of course, surveys are useless without the ability to aggregate the data and see how you’re doing. The back end of a survey system must include reporting capabilities, perhaps with charts and graphs, that help you visualize your customers’ perception of your service. Compare this report to your SLA compliance report—do you see any differences? If your SLA shows that you’re doing a great job, and your customer surveys aren’t so glowing, then maybe your SLAs aren’t set at the right levels—or maybe your SLAs aren’t the only metric you should be looking at.

I've worked with a number of customers who have found themselves in exactly that situation: "Our SLAs are all being met, every day, but our users still don't think we do a good job. What's the problem?" We discovered the answer with a few ad-hoc surveys that touched on "soft" issues, such as the IT team's "attitude" when helping users. Turns out that the team came across as brusque and sometimes rude. We spent some time with the team, and discovered that they felt an incredible amount of pressure because of the number of tickets assigned to them. In the end, the company developed internal metrics to track each IT member's workload and efficiency, and worked to bring each person's workload to a more manageable level—while continuing to survey those "soft" issues such as attitude. The moral of the story is that SLAs aren't the only metric you need to concern yourself with, and integrated surveying can help reveal critical information to help pinpoint overall service problems.

When Everyone Doesn't Need to See Everything: A Multi-Tenant Approach

Multi-tenant is a growing trend amongst IT solution vendors, and for good reason. Obviously, service providers operate the very definition of multi-tenant systems. If you're a service provider, or perhaps more specifically a Managed Service Provider (MSP), then you know the importance of having tools that can be customized and partitioned for each of your customers. Customer A wants *these* dashboards, while Customer B wants *those*. Customer B certainly doesn't want to see Customer A's tickets (and Customer A doesn't want Customer B to see them!). In the past, it's been pretty common for such multi-tenant features to *only* be present in solutions that were designed for MSPs.

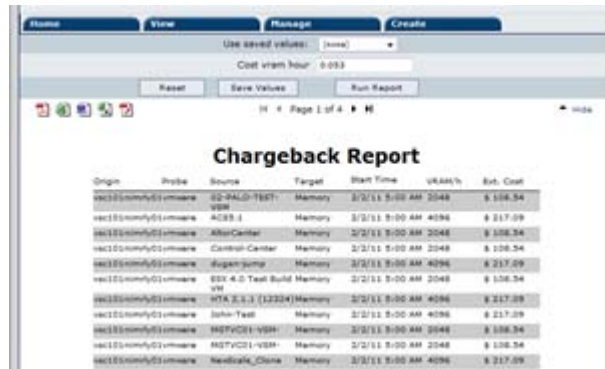
Today, however, that's changing. Large, multi-divisional companies want to deploy solutions that can serve all of their divisions' needs without necessarily deploying a unique solution for each division. That's where multi-tenancy can help, enabling a single solution to be customized, partitioned, and presented to each division *as if* they were the only ones using the solution, when in fact the solution is consistently serving everyone. Different divisions can get a different view of just *their* portion of the environment. For example, Division A might see a dashboard, while Division B saw something completely different.

Again, multi-tenancy isn't something that every single company or organization is going to need. However, it's a nice feature to have in your back pocket if the time comes when you *do* need it, so be sure to consider this functionality as you're evaluating various solutions—even if multi-tenancy isn't an *immediate* need. Of course, if you're an MSP, multi-tenancy is definitely a must-have feature.

We're continuing to support this chapter's theme of keeping everyone in the loop: The ability to provide specific, customized, partitioned environments to your varying customers—whether internal or external—helps keep them more informed and more *accurately* informed.

Call It a Private Management Cloud: Allocating Costs

There's one more thing we should look at to keep everyone in the loop, and that's with regard to their costs: The ability to provide customers with detailed reports on their usage of the infrastructure, and to potentially bill them for their usage based upon those reports. Figure 3.8 shows what such a report might look like.



The screenshot shows a web-based interface for a 'Chargeback Report'. At the top, there are navigation tabs for 'Home', 'View', 'Manage', and 'Create'. Below these, there are input fields for 'Use fixed values: [none]' and 'Cost per hour: 0.033'. There are buttons for 'Reset', 'Save Values', and 'Run Report'. The main content area displays a table with the following data:

Origin	Probe	Source	Target	Start Time	UKAs/s	Est. Cost
10.1.1.1	10.1.1.1	10.1.1.1	10.1.1.1	3/2/11 9:00 AM	2048	\$ 158.54
10.1.1.1	10.1.1.1	10.1.1.1	10.1.1.1	3/2/11 9:00 AM	4096	\$ 317.09
10.1.1.1	10.1.1.1	10.1.1.1	10.1.1.1	3/2/11 9:00 AM	2048	\$ 158.54
10.1.1.1	10.1.1.1	10.1.1.1	10.1.1.1	3/2/11 9:00 AM	2048	\$ 158.54
10.1.1.1	10.1.1.1	10.1.1.1	10.1.1.1	3/2/11 9:00 AM	4096	\$ 317.09
10.1.1.1	10.1.1.1	10.1.1.1	10.1.1.1	3/2/11 9:00 AM	2048	\$ 158.54
10.1.1.1	10.1.1.1	10.1.1.1	10.1.1.1	3/2/11 9:00 AM	4096	\$ 317.09
10.1.1.1	10.1.1.1	10.1.1.1	10.1.1.1	3/2/11 9:00 AM	4096	\$ 317.09
10.1.1.1	10.1.1.1	10.1.1.1	10.1.1.1	3/2/11 9:00 AM	4096	\$ 317.09
10.1.1.1	10.1.1.1	10.1.1.1	10.1.1.1	3/2/11 9:00 AM	2048	\$ 158.54
10.1.1.1	10.1.1.1	10.1.1.1	10.1.1.1	3/2/11 9:00 AM	2048	\$ 158.54
10.1.1.1	10.1.1.1	10.1.1.1	10.1.1.1	3/2/11 9:00 AM	4096	\$ 317.09

Figure 3.8: Reporting on usage-based metering and billing.

Again, this kind of reporting is an obvious, must-have feature for MSPs—but it has increasing applicability to organizations who deal only with internal customers.

One of the key elements of cloud computing is the concept of billing you based on your actual usage. The cloud provider builds and manages the infrastructure, which is shared amongst their customers. Each customer then pays for the bits they use. That's an obvious and well-understood model for the public cloud—but it's becoming a model for the *private* cloud as well. Rather than accepting IT as a giant bucket of overhead, companies are looking more and more at allocating IT's costs across the consumers of those IT services. "Marketing wants to spin up a dozen virtual Web servers for a new Web site? Okay—do they have the budget to pay for it?"

Chargebacks, as they're called, are certainly nothing new. But monitoring and service desk solutions are increasingly able to provide the level of detail that you need to actually make chargebacks *work*. The technological advancements that have made public clouds possible can be readily integrated into private data centers for the same purposes: billing (or allocating costs) for actual usage.

Tying IT costs directly to the consumers of those IT services is a great approach for helping IT make better business decisions. Rather than putting IT in the role of "gatekeeper" for who can and cannot have specific services, the organization's *management* gets to decide what money will be spent, by whom, on what services. That's how it should be. In one sense, IT has always been an outsourced activity: Although the IT team might be paid by the organization, they don't materially participate in the organization's actual profit-making activities. They're a separate division. Essentially, the business has "outsourced" IT (albeit to an internal team)—why not have IT deliver usage-based billing statements just like any other vendor would be expected to do?

It's just another way of keeping everyone in the loop. Even if you don't use your usage-based billing reports for actual billing or chargeback, they're a useful way of helping upper management understand the cost and value of their IT investment. "Yes, you spent a zillion dollars on IT last quarter but here's why, and here's how that investment was consumed by the organization. If you want to cut back, start by looking at the consumers, and finding ways to make them consume less."

Conclusion

This chapter has been all about keeping people in the loop when it comes to IT management. From keeping users more updated and engaged in the IT process, to keeping technicians more connected to ongoing events, to keeping management more informed so that they can make better decisions—it's been about *communications*. There's very little I've discussed in this chapter that any organization couldn't start doing today, if they were willing to expend enough effort. The key, however, is in accomplishing these goals with little or no effort, by using a system of integrated software tools that understand how to do these things for you.

Coming Up Next...

In the next chapter, we're going to look at a challenge that's become more and more common in IT: key services and IT elements exist outside the data center. Yes, you can call it "the cloud" or you could simply call it "outsourced services." Whatever you call them, they're still critical to the business, and you need to treat them the same way you treat all of the in-house services. You *can't* treat them as a separate silo, because then you'll be forcing yourself to manage them differently. Of course, monitoring outsourced services is a whole different ball game than managing in-house services, so we'll need to find some clever solutions.

Download Additional Books from Realtime Nexus!

Realtime Nexus—The Digital Library provides world-class expert resources that IT professionals depend on to learn about the newest technologies. If you found this book to be informative, we encourage you to download more of our industry-leading technology books and video guides at Realtime Nexus. Please visit <http://nexus.realtimepublishers.com>.