Realtime publishers

The Definitive Guide™ To

Application Performance Management

sponsored by

Greg Shields

Copyright Statement

Chapter 8: Seeing APM in Action

You could easily think of this chapter as the "companion" to the previous chapter. In Chapter 7, you learned about the best practices in creating APM visualizations. By analyzing a sample of mocked-up dashboards, the previous chapter presented a number of ways in which APM visualizations enhance IT operations as well as business decision making.

However, it's worth saying again that APM *is all about its pictures*. The return provided by an APM solution comes from the data it presents to the multiple classes of users in your organization: administrators, developers, business executives, Service Desk employees, end users. As such, truly understanding the value in those visualizations needs a second look.

You've hopefully been enjoying the chapter story of TicketsRus.com, and how Dan and John and the entire cast of players have evolved along with their monitoring capabilities. You've seen how John's job has gotten easier as the notifications he and his team receive grow more useful. You've also seen how Dan gets the quality information he needs to make data-driven decisions as a business executive.

But what you really haven't experienced to this point is a complete walk through of the entire process. Such a walk through can tell the extended story of a potentially scary problem along with how the APM solution set assists. That storytelling happens in this chapter. Its goal is to add a dash of humanity into the 20,000-foot perspective that's been the forefront thus far in this guide.

The story you'll be reading is entirely fictional but is based off the types of problems and war stories that you probably experience every day. Every IT organization in every business has its share of technology issues; they're a fundamental part of doing business with technology. This chapter's story is written to show how the presence of a fully-implemented APM solution enables every actor to do a better job supporting the needs of the company as a whole.

You'll also quickly notice that you've already seen many of the images here in previous chapters. Many are slightly-adjusted views of those seen in Chapter 7, where this guide's extended discussion on pictures and data-filled visualizations was most prevalent. Where feasible, the images have been altered to fit the storyline and its characters. This reuse is done intentionally, to bring a sense of continuity between the topics that have already been discussed and the story that ensues.

APM Helps Avoid the "War Room"

Yet before we get into the story proper, let's step back a minute and think first about the traditional ways in which IT organizations identify, troubleshoot, and resolve situations as they occur. These traditional ways have been developed over time as IT professionals develop a sense for troubleshooting, "sniffing out" root causes to problems based on previous experience and gut feeling.

When triaging administrators become aware of problems through user interactions, and when troubleshooting administrators must track down solutions based on gut feelings rather than hard data, the resulting process is purposefully un-optimized. In immature environments that don't use a data-driven approach to problem resolution, eight steps are common: awareness, assessment, assignment, handoff, transaction-level triage, infrastructure-level triage, handoff, and solution.

Awareness

The first step in the unmonitored environment is often its most painful. Actually becoming aware that an IT problem exists is one of the hardest hurdles to overcome when IT organizations lack process maturity and monitoring integrations. Without formalized systems for watching and reporting on system behaviors, the first acknowledgement of a problem often comes after the users have been affected.

Assessment

"There's a problem with the database? We'll look right into that." This statement is a common response in the unmanaged environment. In immature environments, the assessment of the problem often starts with simply verifying that the user's called-in problem actually exists. With little or no instrumentation present, this process requires walking through the stated problem as presented to the Service Desk. For those with traditional, stove-piped monitoring solutions, this step can also involve verifying stated behaviors using each solution's individual console while attempting to manually aggregate their information. In short, without monitoring integrations, the assessment process here consumes time in actually determining *whether the problem even exists*.

Assignment

All but the most nascent IT organizations today use some form of work order tracking system to transfer ownership of problems from one team to another. This process, even in immature environments, can appear to be well solved, but in reality may be masking processes that are not optimized.

Consider the quality of the data that often makes its way into work orders as they are created. When a user calls in to announce a problem, the triaging Help desk operator must insert the correct amount of useful and pertinent data into the work order. If this doesn't occur, the assignment process fails and forces the next person in line to again assess the problem. The result is lost time and effort.

A central problem in this step relates to how problem information is documented as it transfers from one individual to another. Without a common visualization, like what is provided through APM, there is no common language between teams. The result is a bit like the "telephone game," with information quality growing ever worse as ownership of the problem changes hands.

Handoff

The handoff process itself also suffers in an immature environment, primarily due to the establishment of priority to the issue. In unmonitored environments, it is functionally impossible to determine the number of affected users except by sheer guess. Often, the decibel level of the caller determines the priority of the work order rather than its actual impact on business operations. The result is that problems get worked on in the wrong order, satisfying "louder" callers over others with greater needs.

> **Resource**
>
> This problem is, in fact, so endemic to immature organizations that it is a common theme in this guide's companion *The Executive Guide to Improving Your Business Through IT Portfolio Management*. There, the problem of "decibel management" as it relates to IT projects is discussed in greater detail.

Transaction-Level Triage

At the point of handoff, it is common for the ownership of problems to be forked towards either developer or infrastructure teams. Developer teams by nature look at problems in relation to their codebase, seeking out areas where individual transactions might be non-optimized or service components may be broken.

Infrastructure-Level Triage

Infrastructure administrators, in contrast, look at problems with a more big-picture approach. These individuals leverage their infrastructure experience and cross-system vision to analyze issues in relation to all the other components in the infrastructure.

Characterization

Both halves discussed in the two previous points approach problems using different paths. This double-pronged approach is particularly suited for the types of customized business services that are common in customer-facing applications. However, there exists a shortcoming when the problem cannot be characterized by either team. In this case, the problem is often bounced back and forth between each team or their sub-teams, such as "networking" versus "security," and so on. As will be discussed in a minute, complex problems that bridge problem domains require the support of multiple teams. Lacking a common vision, the only way to bring teams together is through a "war room" approach.

Handoff

Handing off the problem once again from troubleshooting to issue management for its ultimate resolution is yet another step in the process. This step often requires coordination between change management and configuration management teams as well as communication between each component's stakeholders.

Solution

The final step is in actually implementing the identified solution, a long way from its initial awareness seven steps ago. You can see how this multiple-step process ensures that no issue goes untracked, but at the same time, it creates a burden of process overhead for the solution. Particularly problematic are those times when issues are much larger than a single team can handle alone—such as when a code issue impacts the infrastructure as a whole or when a change to the infrastructure breaks some piece of code.

In situations where multiple teams are needed to solve the problem, the "war room" approach becomes the tactic of choice in many organizations. By bringing representatives from every team into a single room, each becomes responsible for tracking down artifacts of the problem within their zone of control.

The problem with this war room approach is in its costliness. War rooms are necessary in the unmonitored environment because data is not consolidated into a single location for common consumption. Individual team members can't simply look to a Web page to see how others are doing with the problem. In order to actually characterize a problem that exists across multiple domains, the only way in an unmonitored environment to share metrics is through personal contact.

Bringing together such a large group of individuals is disruptive to normal flow of operations and creates an environment of distrust between teams. In most war room situations, the onus of responsibility lies on each team to prove why their domain *isn't at fault*. Finger-pointing commonly ensues, with problem resolution drawing out over extended and unnecessary periods of time.

A fully-realized APM implementation is very much like the digital representation of that war room, but without its actors. As has been stated many times before, an APM solution gathers its metrics from every part of the IT environment, consolidating them into single-glimpse views for consumption by all teams. The result is that environment behaviors across every domain can be seen by everyone without the need for "circling the wagons" and resorting to qualitative "gut feelings" for potential solutions.

APM Streamlines the Solutions Process

Environments that benefit from APM's data-driven approach consolidate the problem resolution process into six very streamlined steps. This new process consolidates many steps from the traditional approach, while at the same time adding a few new ones that improve the overall communication between teams and to the rest of the business. Consider the following six steps as best practices for an APM-enabled environment.

Visibility

Behaviors that occur outside expected thresholds are alerted via high-level visualizations. Through drill-down support, the perspective and data found in that high-level visualization can be narrowed to one or more systems or subsystems that triggered the failure. Using tools such as service quality metrics and hierarchical service health diagrams, triaging administrators can be quickly advised as to initial steps in problem resolution.

Prioritization

Counts of affected users are predefined within an APM solution's interface, enabling triaging teams to identify the actual priority of one incident in relation to others that are outstanding. As a result, those with higher numbers of affected users or greater impacts on the business bottom line can be prioritized higher than those with lesser affect.

Problem & Fault Domain Isolation

Triaging teams then work with troubleshooting teams, often through a work-order tracking system, to track the root cause of the problem. The same visualizations used before in the visibility step are useful here. Different from the unmanaged environment is that all eyes share the same vision into environment behaviors through their APM visualizations. As such, details about the problem can be very quantitatively translated to the right teams to assist in their further troubleshooting.

Troubleshooting, Root Cause Identification, & Resolution

Using health metrics, the problem is then traced to the specific element that caused the initial alarm. That alarm describes how the selected element is not behaving to expected parameters. Here, troubleshooting administrators can work with other teams (networking, security, developers, and so on) to translate the inappropriate behavior into a root cause and ultimately a workable resolution.

Communication with the Business

During this entire process, business leaders and end users are kept appraised of the problem through their own set of APM visualizations that have been tailored for their use. Business leaders see in real time who and how many people are affected by the problem as well as how much budget impact occurs. End users are notified through notification systems that give them real-time status on the problem and its fix.

Improvement

Throughout the entire process, the APM solution continues to gather data about the system. This occurs both during nominal as well as non-nominal operations. The resulting data can then be later used by improvement teams to identify whether additional hardware, code updates, or other assistance is needed to prevent the problem from reoccurring. By monitoring the environment through the entire process, after-action review teams can identify whether the resolution is truly a permanent fix or if further work is needed.

It should be obvious to see how this six-step process is much more data driven than the earlier traditional approach. Here, every team remains notified about the status of the problem and can provide input when necessary through the sharing of monitoring data. When problems occur that cross traditional domain boundaries, those teams can work together towards a common goal without the need for war rooms and their subsequent finger-pointing.

With this in mind, let's see how this new approach works for the ongoing story of TicketsRus.com. In this continuance of the story, John finds himself starting what appears to be a regular day at the office. He has started his day by completing the tuning of a behavior threshold within his APM solution.

The problem that is about to ensue can and will be handled much differently than in previous situations. This time, John and his teams have the data they need at their fingertips to turn what could be a disaster into a relatively minor impact on their daily operations. Rather than repeating the public horror of their previous "Labor Day Incident," John and his teams keep the problem from growing out of control and affecting their end customers.

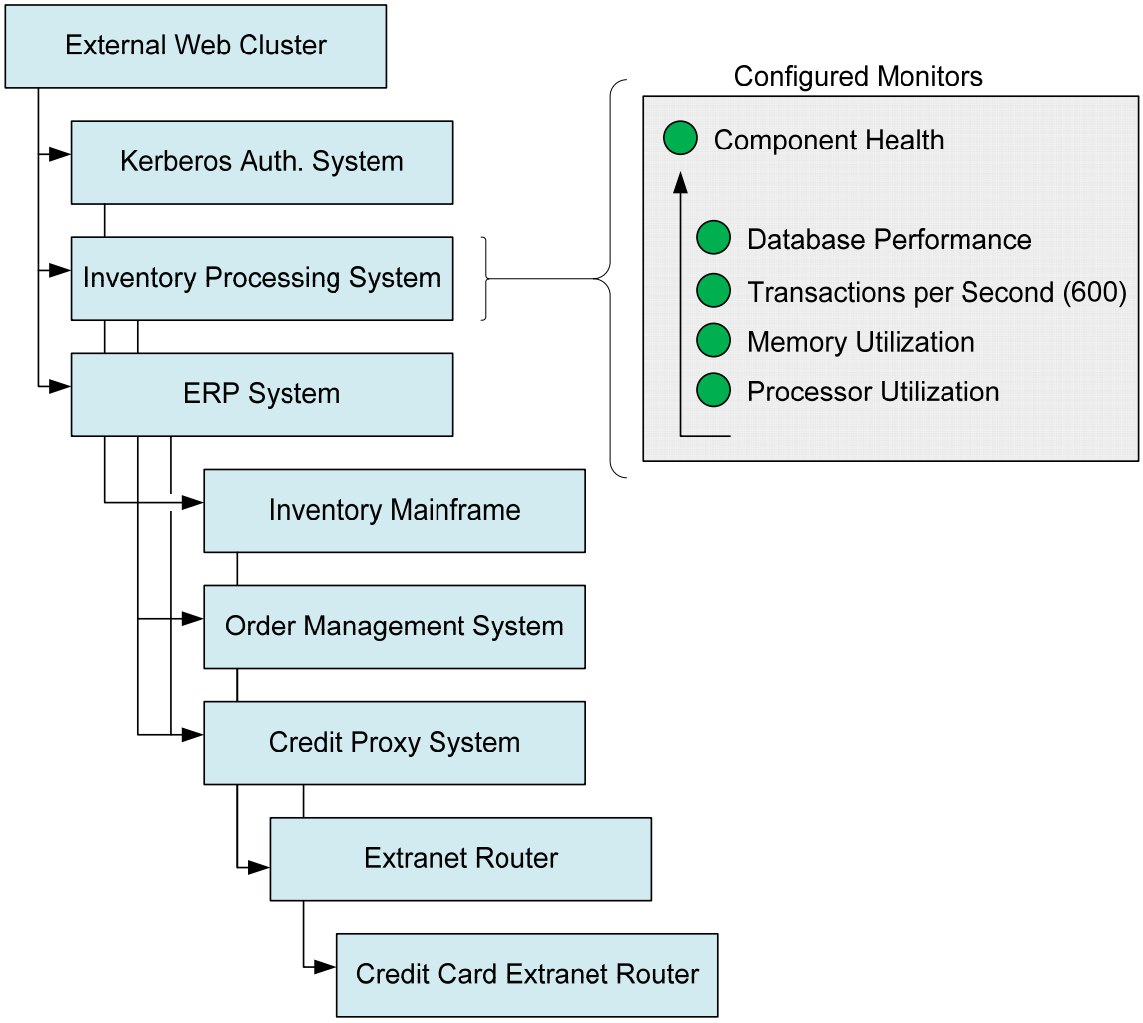
TicketsRus.com—A Day in the Life

Its 8:18a and TicketsRus.com IT Director John Brown finds himself putting some finishing touches on one of the elements in his APM service model. Today, John finds himself tuning a database performance threshold for a monitor that is attached to the Inventory Processing System.

As he begins this relatively trivial task, he thinks back to an afternoon last week. That day, a false positive in this particular monitor caused a minor stir at the Service Desk. During that afternoon, the company had just released a new set of tickets for a fairly popular concert to happen in a large local venue. The band for that concert had just returned from an extended break, and their fans were eager to see them perform once again. The resulting rush created an abnormally heavy load on the Web site that the company hadn't experienced in a while, and at least one incident that was related to this very counter.

After the incident, a few code changes were also made to the system. The same heavy load was expected this week as the same band's second series of concerts were to be made available to fans.

Clicking through his APM solution's designer tool, John brings open the Service Model for TicketsRus.com's external Ticket Sales application. There, he finds the hierarchical diagram that he and his teams had spent so much time tuning over the past 6 months. Including representations for every component in the system, from the External Web Cluster all the way through the Order Management System and others, this designer tool provided the workspace where they built the logical representation of the overall system.



John double-clicks on the Inventory Processing System element and brings up its properties. There, he is shown the list of health monitors that are attached to this particular system. Each monitor is configured with one or more settings that define what TicketsRus.com considers healthy versus unhealthy behaviors. Browsing down through its fairly comprehensive list, he finds the culprit in last week's alert brouhaha.

"A-ha!" says John to nobody in particular, "It looks like that last batch of tuning updates dialed down the Red threshold on Transactions per Second to a number that's far too low. Let me just set this back to 600, where it belongs."

He clicks to view the properties of the Transactions per Second element and makes the appropriate change. He thinks to himself, "APM is a great solution, but you absolutely have to keep on top of your tuning. Making sure that every alarm is tuned properly seems to be a never-ending activity in this tool."

Thankfully, his chosen APM solution arrived with a set of preconfigured profiles that got the monitoring team started. Those profiles gave the team a set of starting points for each class of hardware, application, and service that were based on known best practices—for example, 80% for Web server processor utilization, 600 for database transactions per second, and so on.

He recalls the first few weeks after its initial installation where a few of those "suggested" threshold levels weren't really all that tuned for his environment. During that period, he had shown up for work with an entire screen of health stats flashing Red for warnings that really weren't all that relevant to his environment. "There aren't that many Web-based ticket sales applications out there that service an entire continent, so we had our hands full with customizing for a while," he reminisces to himself.

It took a period of time to get those metrics tuned just so, but there were always areas for improvement in the system. That's why he spent a period of each day keeping tabs on how well the solution was doing its job. Today's adjustment for the Inventory Processing System's metrics is no different.

Browsing around through some of the other screens, he notices something out of place. "That's funny," he exclaims, "Now why is the country map showing a yellow condition for one of my sites. The…Rochester site...it seems. We're not doing any work there today."

John calls up the Service Desk to see what the matter is. With Dan halfway across the world at some ticket seller's conference—"Yet another junket," thinks John—John is technically in charge of operations. And a major issue is all he needs.

"Service Desk, this is Eloise."

"Hey, Eloise, this is John. Tell me about the yellow condition I'm seeing in Rochester."

"Hi John. Well, it was yellow. Look again. APM now says we're having a problem with the entire Ticket Sales application, all across the map. Whatever it was, it started in Rochester and spread to the entire country map. We're looking into it now."

John's heart skips a beat as he recalls the Finals Week incident from not too long ago, a low point in his otherwise illustrious career. If this problem is for real, it'll be the moment of truth for his relatively new APM solution. Another repeat of that Finals Week situation, and he could be looking for a new job.

"I'm coming up to the NOC to manage the incident. I'll be there in a minute."

John drops the phone into its cradle. Before he leaves, he calls up his highest-level Service Quality visualization for all his monitored systems. The board shows Green everywhere, with the exception of a fairly-disturbing strip of Red across the line marked Ticket Sales. Displaying 2674 in the Overall column, John recognizes that some nearly-3000 people aren't able to buy tickets right now. He dashes off to the Service Desk to coordinate his teams.

	Overall	Canada - East	NA - Central	N
Broker Services				
Ticket Sales	2674	268	1520	
Internal Operations				
Internet Access				
External Processing				

The NOC is a flurry of activity. Though, John thinks, this incident's "flurry" doesn't seem as "frantic" as in previous major incidents. Rather than the usual rush of administrators scurrying in and out, seeking updates and where they can help, this time the hubbub is more like a murmur.

He heads to Eloise's desk for an update. "Whatever it is, it appeared to start in Rochester and spread through other parts of the system. Right now, we're showing Red conditions in the NA-East and Canada-East zones. The other zones are showing Yellow, which I'm not sure what to think," she states.

Name	Service Quality				
	Current	Duration	During Period	Available	Monthly Trend
NA - East	Down	1h 35m			
Canada - East	Down	45m			
NA - Central	Warning	1h 20m			
NA - West	Warning	1h 15m			
Canada - West	Warning	1h 25m			

"Was anyone doing any work on the system when it went down?" John asks. Although doing work on the system outside its designated maintenance windows wasn't supposed to happen, sometimes his eager administrators tried to sneak in work when they shouldn't.

"As far as I know," says Eloise, "nobody's even been in the data center this morning. The network administrators have been in a meeting, and most of the systems administrators have been at their desks prepping for the next Change Management meeting."

"OK," John asks of the room, "Will someone call down to the systems admins and see if anyone's been up to anything? And will you," he points towards one of the Service Desk employees at random, "run down to whatever conference room the network administrators are in and see what they know?"

By this point, pretty much everyone in the IT organization had probably received some kind of notification about the problem. Being the money-making system for the company, almost everyone was on notice when a problem of this scope occurred. Why they hadn't called into the Service Desk yet either means that they're looking at their own visualizations—proving, he thinks, that the processes surrounding his APM solution are working—or they're all huddled around the water cooler. He hopes for the former.

"Eloise, let's drill down a bit into these alerts and see what's going on," he faces the large screen in the center of the NOC's alert wall. Recently installed, it now operates as a kind of giant heads-up monitor for the entire Service Desk to see at once. That monitor has access to all of his APM solution's visualizations, from developer to administrator to even Dan's business executive views. It has already come in handy on a couple of occasions.

John continues, "Bring up the high-level network view. Are we still seeing those odd behaviors from last week?"

Eloise takes control of the projected screen's computer, clicking through visualizations to find the one John wants. She brings up the high-level network screen he's interested in seeing. It discouragingly shows more than a few elements that aren't in Green.

Region	Unique local hosts	Total bytes	Local RTT	Upstream loss rate	Downstream loss rate	Network performance	Affected hosts
Internet	3.6 k	28.2 GB	55.3 ms	0.7 %	0.7 %	89.9 %	380
Default	1.48 k	5.78 GB	31.9 ms	1 %	0.6 %	77 %	660
EMEA	164	1.72 GB	133 ms	< 0.1 %	0.7 %	88.6 %	42
APAC	104	851 MB	229 ms	0.8 %	1.1 %	96.7 %	4

"Somebody help me here," John asks no one in particular, "We were working on these network metrics just last week, and they're still in Yellow and even a few Red. Are these for real, or are these problems that are still residual from our tuning activities of last week?"

Right then John's network engineer pops into the room, "Ignore those colors, John. That's the meeting that me and my team where in just a few minutes ago when this whole thing started. We're still tuning our part of the system. You aren't seeing these error conditions roll up right now because we're still trying to determine what thresholds make sense for us."

"Have these changed from where they were last week?" John asks, "The colors look to be in the same place, but is today's 'real' situation impacting your numbers in any way?"

The engineer squints at the rows of numbers on the screen. They show his network performance metrics alongside upstream and downstream loss rates, round-trip time, and total bytes, "Everything looks OK. The numbers are on the same magnitude of what we were seeing before any of this happened. It looks like the problem isn't us this time."

"Now, I thought we decided we were ending that friendly rivalry?"

The engineer smiles, "Old habits die hard, man. Let me get back with my team and keep digging around over there. We'll see what we can find."

"Hey, John," Eloise catches his attention. While John's been chatting with his network engineer, she's been browsing through a series of other screens, "It looks like today's problem might be with the servers. Check this out."

Eloise has drilled down past the top-level screens to view each individual technical system. There, she's looking at a new visualization that shows health statistics for the system's Servers as well as Database, Network, and Software Infrastructure. The Service Quality indicator for Servers shows Yellow.

Service Title	Service Quality		Availability		MTBF
	Current	Duration	Current	Overall	
Servers	Warning	27m 0s	✓	100.00%	1m 25s
Database Infrastructure	Normal	35m 32s	✓	0.00%	
Network Infrastructure	Normal	35m 56s	✓	0.00%	
Software Infrastructure	Normal	8m 22s	✓	100.00%	1m 25s

"Double-click Servers. Let's see if we can't figure out which one's having a problem today."

	Unique users	Application performance	Affected users	Network performance
Frontline	1.15 k	96.8 %	108	77 %
Salesforce.com HTTP	4	-	0	100 %
Site HTTP	232	69.3 %	31	50.3 %
Changepoint SQL	8	99.9 %	4	93.7 %
Changepoint	369	99.7 %	29	86.4 %

Eloise double-clicks the Servers link to bring forward a new screen. On this one, she's presented a list of all the servers that make up the Ticket Selling system. "We know we can ignore the network metrics right now. Frontline and Changepoint both seem OK, as well as most of the others. Salesforce is unavailable now, but we know about that one. It's the Site HTTP that doesn't look happy. It's showing less than 70% for its application performance."

"Now, we're getting somewhere," says John. He looks at his watch. The time shows 8:29a. Exactly 7 minutes have passed since the first APM light went from Green to Yellow, "Bring up the Service Tree for our Servers."

Eloise drills further into the visualization, bringing forward the Service Tree. For TicketsRus.com, the Service Tree for Ticket Selling servers is broken down into Infrastructure, Application, Database, and Mainframe servers. The indicator for Infrastructure Servers glows Red.

She clicks the plus sign next to Infrastructure, and sees that the Web Servers seem to be triggering the Red alert. Drilling further, she exposes that the problem is with the Web server at 10.4.224.42, one of the servers in the Ticket Selling application's External Web Cluster. Clicking one more time, the error appears specifically to fault its CPU utilization.

"So, here's a source of the alert," John tells Eloise, "We now know the predominant symptom of the problem. Let's get this info to the systems administrators. I'll call them. You draw up a work order with this info."

Half a world away, Dan's been sitting at the bar with his old friend and now competitor Lee Mitchell. With his PDA viewing the same Web screens as everyone in his NOC, he's aware of the situation as well. Right now he's showing Lee the country map view on his PDA as one icon flashes from Green to Yellow.

"So, here you can see that we've got an issue in Rochester," Dan continues in his story to Lee. "Some Internet device is probably having a problem, which means that fewer people are connecting through that point of presence."

Dan clicks past the country map to bring forward his availability dials, the two indicators he looks to most to identify how well his systems are working. While talking with his friend, he notices that the dial for User Availability has dropped to 87.4% and continues to go down. With a couple of drinks in him, he's in too good a mood to worry too much. He knows that John's got him covered.

Back at TicketsRus.com's corporate headquarters, John hopes that Dan isn't watching this situation as it unfolds. It's always easier to talk with him about these situations after they've happened rather than when they're going on.

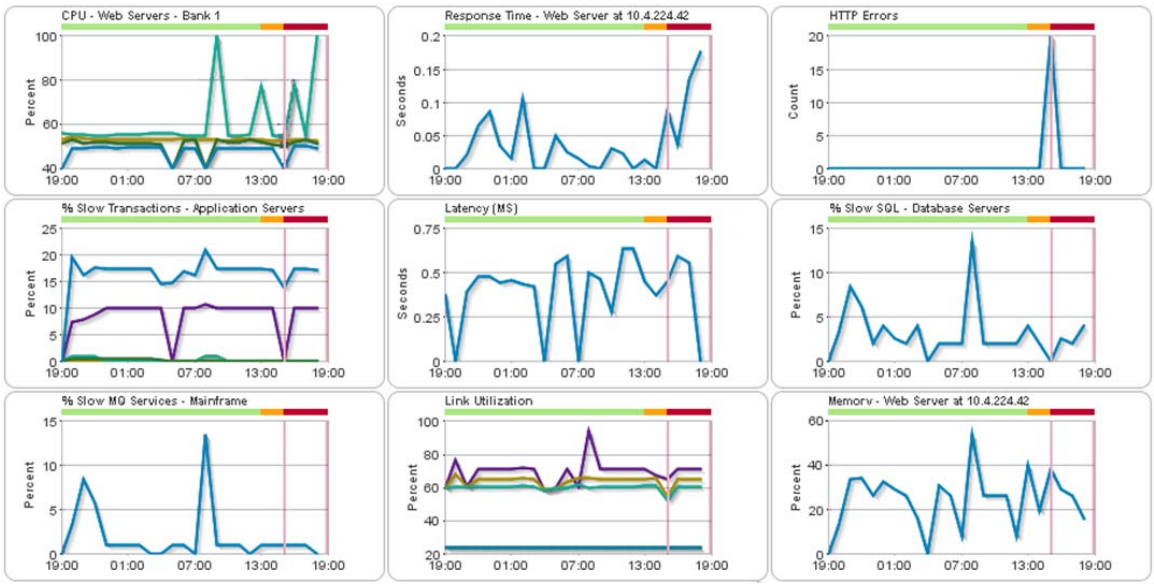
He calls down to the systems administrators, "You guys got the work order, yes?"

"Yep," reports Eric, the administrator who answers the phone, "We're actually way ahead of you. We were together working on a few things for today's Change Management meeting when the alerts started to come across. So, we've been digging into the metrics to see what we can find for the past few minutes."

John thanks Eric, "Let me know when you find something."

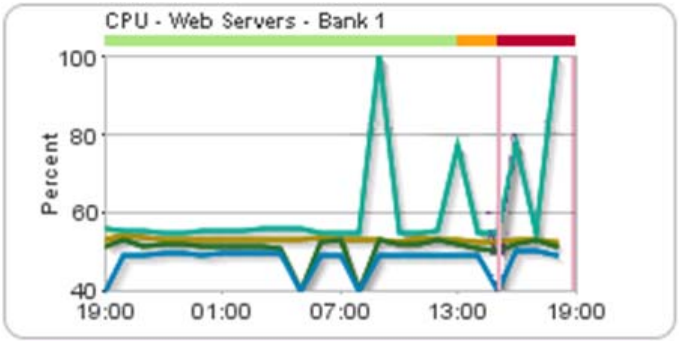
While the Service Desk has been focused on high-level quality alerts, the systems administrators have been focusing their attention to the actual performance metrics on their servers. This group believed pretty quickly that the problem might lie somewhere within their realm of control, considering the data they're seeing.

Eric brings forward what he jokingly likes to call his Monster Dashboard for the Web server that's having a problem. He likes this dashboard because it aligns all sorts of metrics by time. He knows when he looks at this dashboard that every metric shows the same period in time, giving him and his team a way to correlate different behaviors that may occur simultaneously or at least very close to each other.

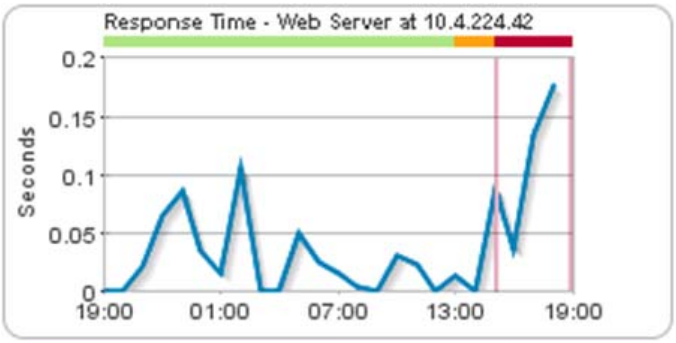


The Web server's Monster Dashboard is equipped with nine performance views all at once. With CPU utilization for the Web server in the upper left, response time for that Web server in the upper middle, and HTTP errors in the upper right. Right away, he sees some correlations between these three measurements that give him a bit more information about the problem.

He focuses on each of these three modules in turn. "Yep, we've definitely got a CPU problem here. It actually seemed to start a couple of hours ago, but is only now getting to the point where it's a problem. See this spike," he points out the trailing spike in CPU utilization to another administrator, "there's our problem."

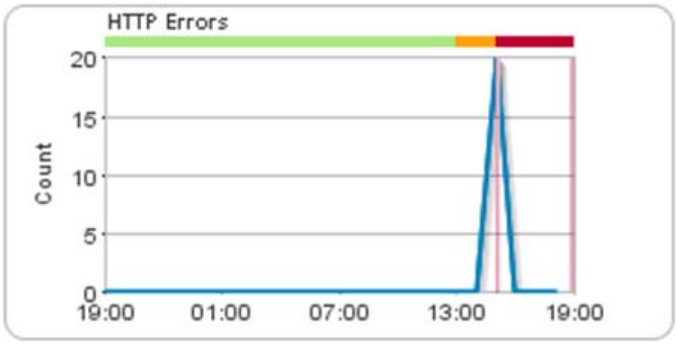


"Now, check this out. You see how the response time for the Web server got quite a bit worse as this problem escalated. Over the past few hours, we've been slowly creeping up to the point where this problem began to be noticeable by users. That's what tripped the alarm.



"Didn't we drop that second set of tickets today for that band's new tour, the one that's been on hiatus for, like, 4 years? I wonder how much of this has to do with an actual problem and how much has to do with their screaming fans trying to get their tickets?

"Well, now, doesn't this just get all. Check out the HTTP Error rate. Looks like we had a spike there, and then it just dropped to nothingness. That's not a pattern that we usually see when the load is super-high. HTTP Errors should be right around zero all the time, unless there's something going wrong. To me, that makes this whole situation feel like a developer problem. Something's wrong in that latest code drop perhaps…?"



Eric ponders this recommendation for a few seconds more and decides to make the call. In the old days, he thinks, this is when the resolution to the problem might get worse than the problem itself. Right around now is when John would have been frantically calling everyone in for a big meeting. We'd all need to bring our suggestions, draw them out, and see which ones made sense. Eric didn't like these meetings very much. They often went far into the night.

This has him thinking, "You know, I never really liked this new APM system when John was talking about it. I was always worried that it'd make my job harder. But now that I'm seeing it here today, I've got to give the guy a hand. It's making this process a lot…well…easier."

Eric picks up the phone to call the developers as he finishes logging his impressions into the work order. Also different than in the previous system, he can link his research notes directly to items on his dashboard. Since everyone in IT can read almost every dashboard—those with the financial info are locked away, but the troubleshooting dashboards are all common access—he needs only jot down the few notes he's made over the past minutes and paste links to the correct dashboard URL right into the work order.

On the other end of Eric's phone call is Rhonda, one of TicketsRus.com's lead developers, "Yep. We're looking at this too. We got the notifications just after you guys did. It is looking like a problem that's on our side. We ran that last batch of code updates through regression testing for days but we must have neglected to run some kind of test. C'est la vie."

Rhonda has also been looking at the problem through her own set of eyes. She first caught wind of the problem while grabbing another cup of coffee down the hall a few minutes ago. She figured she'd look into the problem a bit even before she got word that the ball was indeed in her court.

Rhonda finds her answer as she pulls up her Apdex by Page URL report. In looking through it, she chuckles a bit about how the executives and the Service Desk keep thinking about this Ticket Selling system in terms of "quality."

Page URL	Apdex
https://www.ticketsrus.com/Wait.html	1
https://www.ticketsrus.com/_ScriptLibrary/rs.htm	1
https://www.ticketsrus.com/gateway.jsp?proc=logout	1
https://www.ticketsrus.com/PageDrivers/_ScriptLibrary/rs.htm	1
https://www.ticketsrus.com/PageDrivers/MainPS.jsp	0.99
https://www.ticketsrus.com/gateway.jsp?proc=processsearchresults	0.96
https://www.ticketsrus.com/gateway.jsp?proc=processsearchother	0.94
https://www.ticketsrus.com/gateway.jsp?proc=processsearch	0.94
https://www.ticketsrus.com/gateway.jsp?proc=processnewspage	0.91
https://www.ticketsrus.com/	0.88
https://www.ticketsrus.com/gateway.jsp?proc=processtransaction	0.88
https://www.ticketsrus.com/PageDrivers/PolicyViewPS.jsp	0.87
https://www.ticketsrus.com/gateway.jsp?proc=ratingsummary	0.79
https://www.ticketsrus.com/gateway.jsp?proc=processpolicyview	0.76
https://www.ticketsrus.com/gateway.jsp?proc=processnamedinsuredp	0.7
https://www.ticketsrus.com/SnareWeb/SnareWorks.bat	0.7

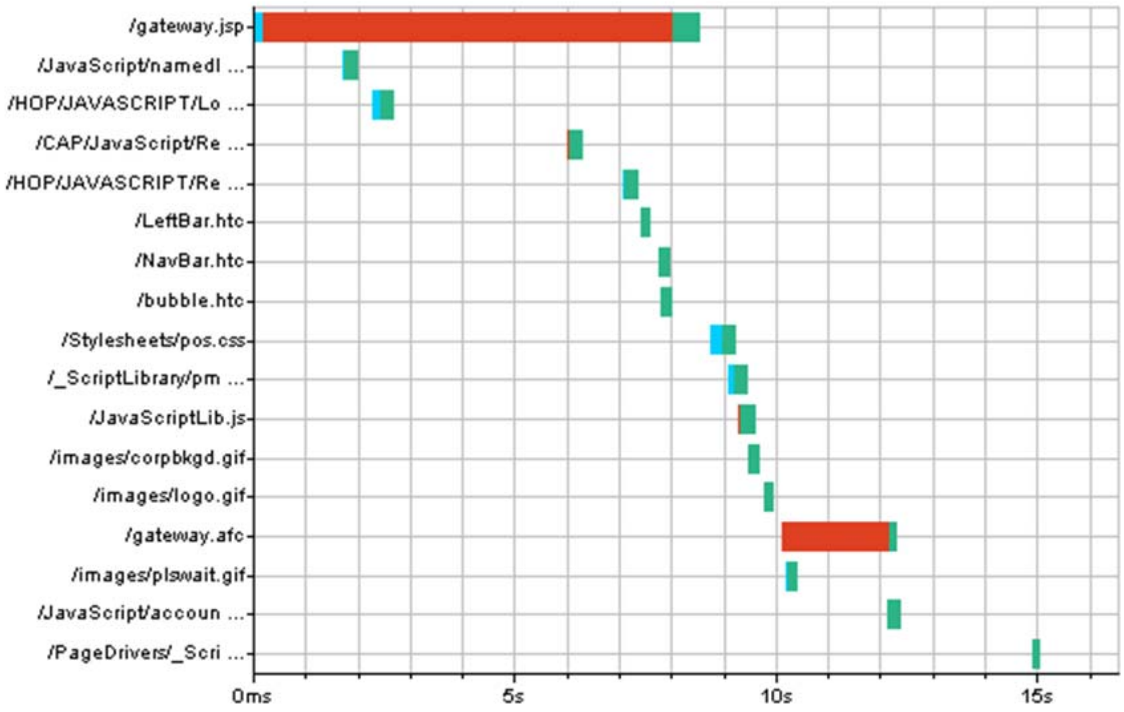
"What a subjective way to think about a deterministic system. Quality, heh," she mutters to herself, "What does quality really measure?"

As soon as the words come out, she stops to think about what she's just said, "Well, I guess, come to think of it, their numerical value for Quality is a lot like my numerical value here for Apdex. Apdex in this case measures how well a particular page is performing, with a value of one meaning 'perfectly' and those less than one describing their level of performance in comparison with each other as well as the established baseline of 'perfect.' OK. So, I'll admit each set of numbers has meaning to each set of people. I'll buy that."

Getting back to the issue at hand, Rhonda takes a look through her Apdex numbers and immediately sees that a few are absolutely not where they need to be. Four in particular give her cause for concern. She thinks to herself, "That batch file shouldn't be performing that poorly, but it's the gateway.jsp code that's really at fault here. Three procedures are absolutely unacceptable in terms of performance."

She continues to herself, "I wonder how much these three procedures are impacting the overall loading of the page."

Rhonda switches her view to call up a view of the average page load time by element. There, she can see each of the pieces of code that goes into a page load, and how much time each requires to complete executing its portion.

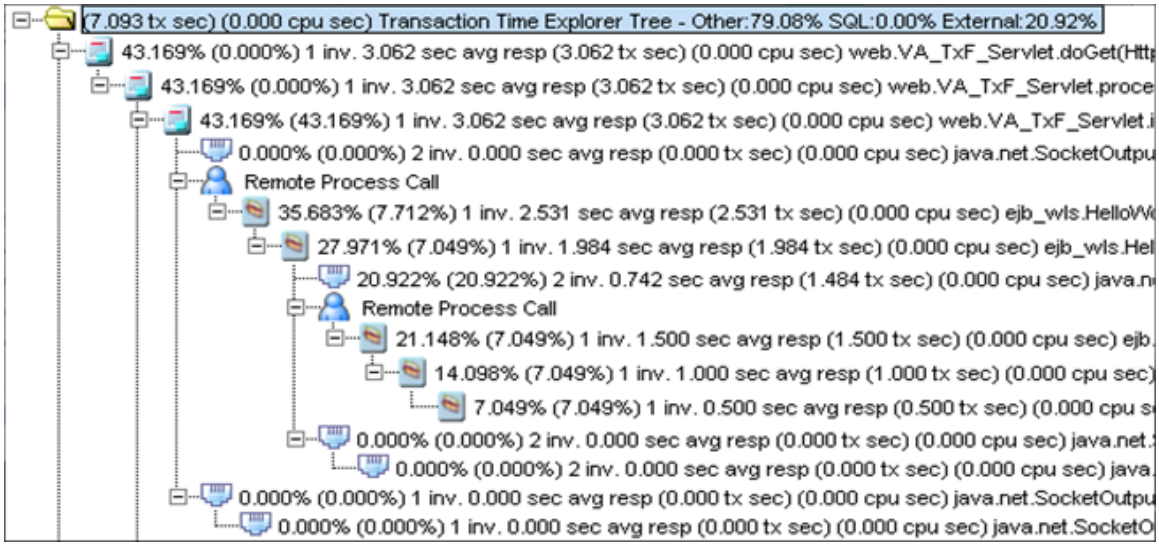


"Wow. Eight, Eight-and-a-half seconds to load that silly .JSP file?" Rhonda admits to herself, "That's waaaay out of the ballpark in terms of performance. Here's why we're having problems loading pages for a whole set of our users. This is one of the new code pieces we updated last week. We're under pretty heavy load right now. There's probably some non-optimized code in the file that didn't get caught by regression testing."

Rhonda feels she's getting closer to a root cause of today's problem. She thinks, "The big question now is, 'How does this load problem relate to the CPU problems that Eric was seeing? Usually a slow .JSP load doesn't necessarily equal a processor overuse scenario…"

To answer this question, she needs to see the communication between the Web server and one of its downstream databases. Rhonda remembers that John's APM solution now gives her some very nice transaction-level metrics between servers. In the old days, if she wanted to see bits and bytes as they cross the wire, she'd first need to call up the network engineers and have them set up a sniffer in the environment. With TicketsRus.com's new Change Control rules, that sniffer could take a while to get approved.

Now, with the APM solution, the sniffers are already in place. Rhonda needs only to find and pull up the right trace. She does just that, and discovers that the extended load time has to do with a double-looping construct in the .JSP code.



"A-ha! There's this slightly-more-than seven second lag in this subroutine. It looks like we're looping inside of looping. Amateur mistake. Sorry, everyone," she mutters to no one in particular.

Rhonda knows exactly the subroutine she needs to adjust to fix the problem. Rewriting a half-dozen lines of code, she calls up John to approve fast-tracking the fix.

John answers, "Rhonda. I thought this was a CPU problem."

"It is," she tells him sheepishly, "and I'm the one at fault. That last batch of code included a…ahem…bug that's causing a delay issue when we get under load. I've tracked down the problem and have already coded the fix. All I need from you is the approval to push it out to the External Web Cluster."

John harrumphs. He doesn't usually implement code fixes this quickly, preferring to run them through testing and Q&A first, but he needs to get this system back online quickly, "Alright. Patch the code."

Rhonda inserts the patched code into her automated staging and deployment system, thoughtfully runs a quick test on it in her testing environment first, then pushes it out. She waits.

John waits also. He looks at his watch again. Its 8:34a. Exactly 16 minutes have passed since the system alerted first on this problem. Sixteen minutes in the old days and they *might* have gotten everyone into the same room by now. Today, 16 minutes later and John finds himself again staring back at his country map. Without warning, his Red and Yellow icons begin flipping back to Green.

His APM solution, it seems, has come in quite handy.

On the other side of the world, Dan has finished his second drink at the hotel bar, as well as his story to Lee about his new APM solution.

"It gets better. Even my developers use it to trace down specific lines of code that aren't working correctly. Everyone from the techies to my aging brain gets the visualizations they need," Dan stops as the formerly-yellow light turns green, "Hey, looks like they've fixed the problem!"

Lee's eyes widen as he realizes the complete vision such a system brings, "Alright, you win. Drinks tonight are on me. Now, tell me more about this system."

Everyone Benefits by Seeing APM in Action

As mentioned before, this story is entirely fictional. But it's not far from the mark in terms of how a fully-realized APM solution brings benefit to IT operations. The Service Desk, administrators, network engineers, and even developers all share the same workspace and experience during times of trouble. Each group can independently work on solutions without the need for war rooms. Even executives gain the comfort that their systems are absolutely being managed correctly, and that their customers are being serviced.

Yet, there's one last part of this story that hasn't been told. This guide has repeatedly referred to the idea that performance metrics can be aggregated with business data to get a financial perspective on IT systems. This capability relates to the topic of Business Service Management (BSM), which is the topic for Chapter 9. By incorporating BSM's financial logic into the technical logic seen through an APM solution, stakeholders like Dan are greeted with real-world impacts to dollars and cents based on situations like the one told in this story. Chapter 9 is next, and in it finishes the story.

Download Additional eBooks 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 eBook to be informative, we encourage you to download more of our industry-leading technology eBooks and video guides at Realtime Nexus. Please visit http://nexus.realtimepublishers.com.