Realtime publishers

Creating Unified IT Monitoring and Management in Your Environment

Don Jones

sponsored by

Copyright Statement

Chapter 4: Monitoring: Look Outside the Data Center

IT has moved beyond our own data centers, and nearly every organization has at least one or two outsourced IT services. Although we're probably always going to have on-premise assets to manage and monitor, we need to realize that in most cases, monitoring has to start *outside* the data center—both in the sense of accommodating off-premises services as well as focusing more closely on what end users are actually experiencing.

Monitoring Technical Counters vs. the End-User Experience

The traditional IT monitoring approach is what I call *inside out:* It starts within the data center and moves outward toward the end user. Figure 4.1 provides a visual for this idea, illustrating how typical monitoring focuses on the backend: database servers, application servers, Web servers, cloud services, and so forth. The general reason for this approach is that we have the best control and insight over what's inside the data center. If everything inside the data center is running smoothly, it stands to reason that the end users who consume the data center's services will be happy.

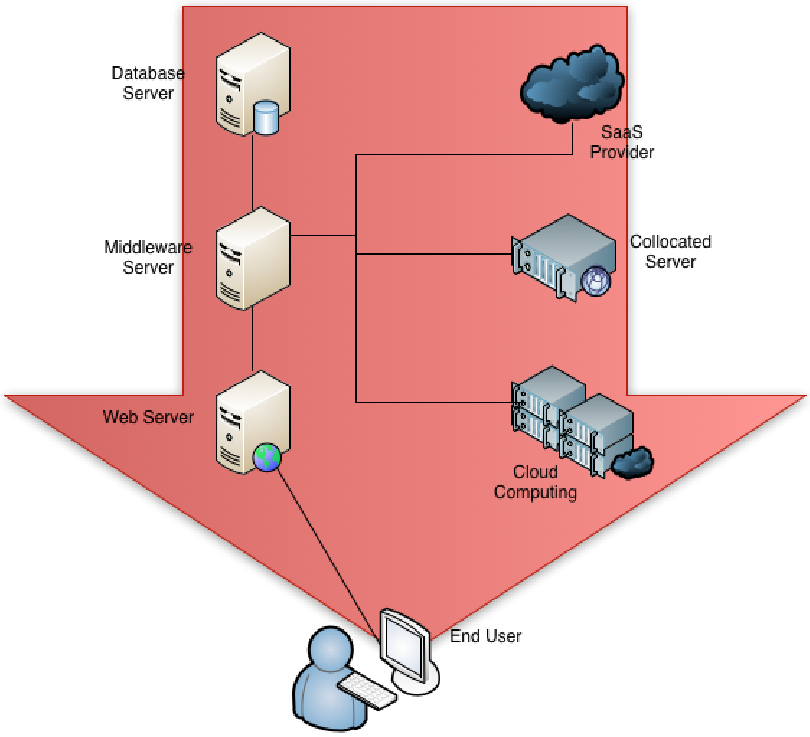

Figure 4.1: Monitoring from the data center outward.

Most Service Level Agreements (SLAs) derive from this approach: We promise a certain amount of uptime, and we set up monitoring thresholds around data center-centric measurements like CPU utilization, network utilization, disk utilization, and so forth. We also tend to look at low-level response times: query response time, disk response time, network latency, and so on.

There's something deeply and inherently inaccurate about the underlying assumption of this approach: Even if you start with a perfect pile of bricks, there's no guarantee that you're going to end up with a stable building in the end. In other words, what end users experience isn't merely the sum of the data center's various metrics. A smoothly-running data center *usually* leads to satisfied users, but that isn't always the case.

It's obviously important for us to continue monitoring these data center-centric measurements, but those can't be the *only* thing we monitor and measure. Current thinking in the industry is that we need to more directly measure what the end user experiences. In fact, "end user experience," or EUE, has become a common term in more forward-thinking management circles.

Here's another way to think of it: Suppose you go to a restaurant to eat. Your steak comes out cooked wrong, they brought the wrong side items, and the waitress is rude. The manager, standing back in the kitchen, thinks everything is fine: the steaks are hot, the veggies are hot, and the waitress smiles at him every time she goes back there. He's focused on the backend, with no knowledge of your expectations. Restaurants address this by having the manager periodically roam around and ask, "Is everything okay?" That's monitoring the EUE: Rather than looking at his back-of-house metrics, he's going out into the cube farm—er, restaurant floor—and testing the waters.

How the EUE Drives Better SLAs

You establish metrics for what the EUE *should be* for various operations: so many number of seconds to complete such-and-such a transaction, and so forth. When that metric isn't met, you start drilling down into the infrastructure to find out why. That's where more-traditional data center monitoring re-enters the picture. Rather than using query response time or whatever to *derive* the end user's experience, we're using it to *troubleshoot* things when the end user's experience is clearly not where we need it to be. Figure 4.2 shows how EUE monitoring sort of reverses the model.

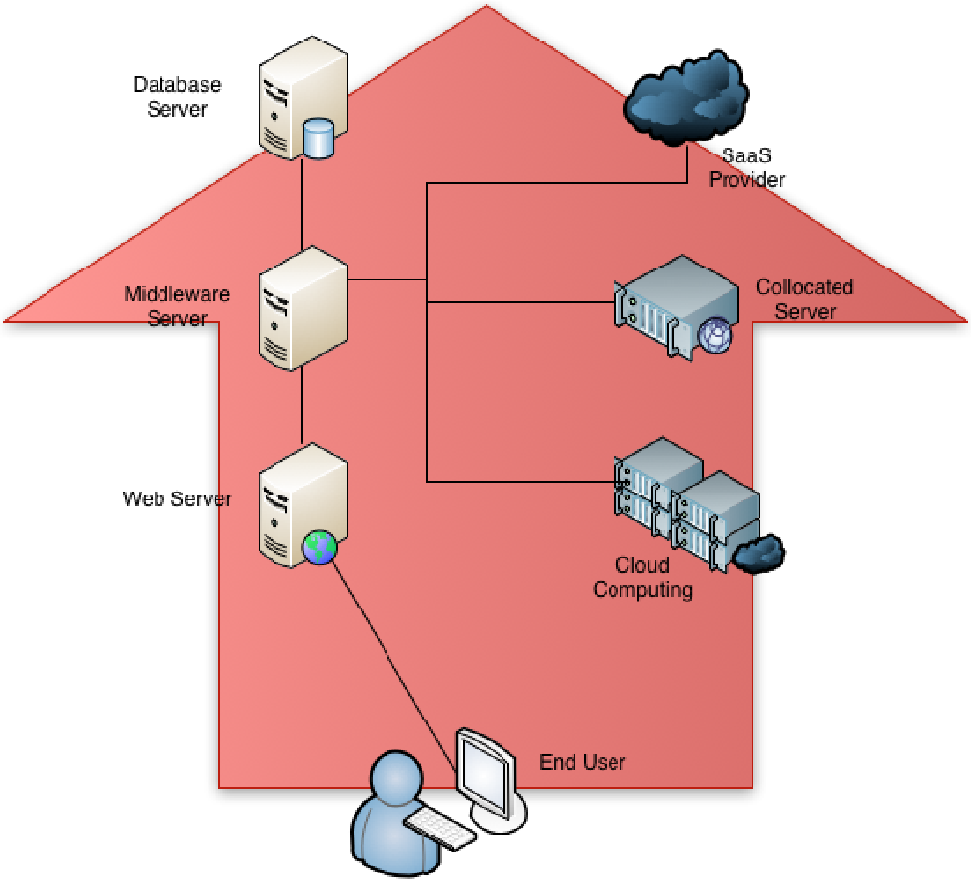

Figure 4.2: Monitoring the EUE.

You'll still have thresholds and other considerations, but they're set at levels that have historically been able to deliver an acceptable EUE. As Figure 4.3 shows, a failed EUE is your cue to start looking at deeper, more technical-level measurements so that you can see what's contributing to the end user problem.

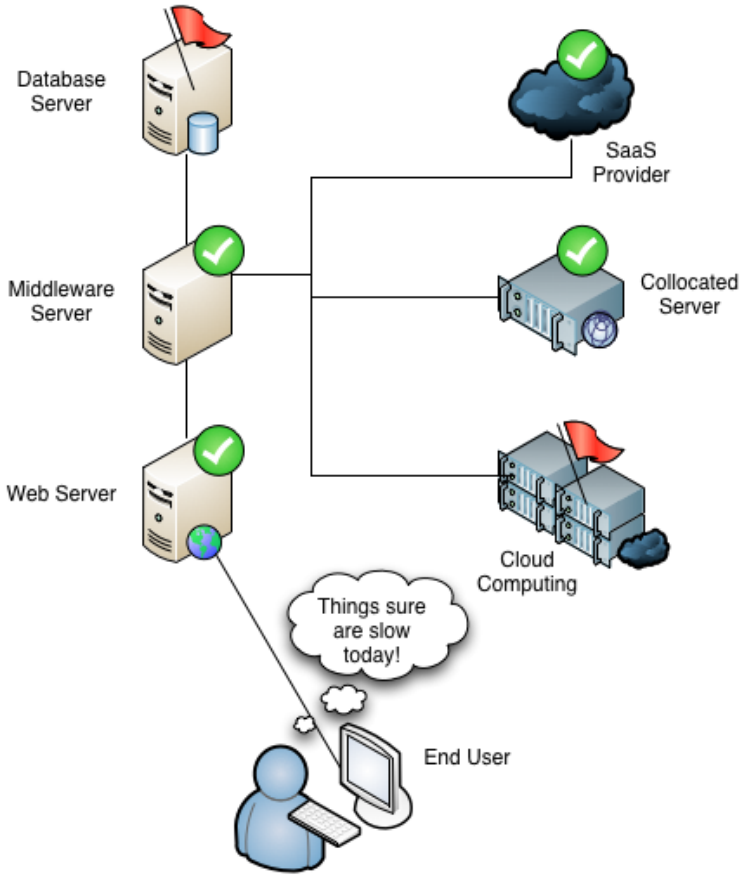

Figure 4.3: Tracking the cause of a poor EUE.

In reality, it doesn't always take a major change in the backend to cascade into a real problem for the EUE. A database server's response times slow by a millisecond or two, resulting in an application server taking an extra half-second to process a transaction, resulting in a front-end server taking an extra second to present the next screen of information, resulting in the user's client application taking a couple of extra seconds. Add up those couple extra seconds over the course of a day, and you've lost an hour or so, and told a lot of customers, "Sorry this is taking so long, the computer is slow today."

In Figure 4.3, both an internal database server and a cloud computing service are responding slowly (indicated by the red flags). Neither one might be alarming in and of itself, but together they're combining to form a noticeable problem for the end user.

Normally, a minor a fluctuation on the database server might not raise any alarms. It's the cascade of effects that result in a poor EUE. Once we definitively know—because we've been watching it—that the EUE is declining, we can start looking for causes. Because we're *looking for a problem,* rather than just routinely monitoring, that minor back-end performance decrease will be more noticeable.

The ability to measure the EUE lets you create much more realistic SLAs. Instead of telling users, "We'll guarantee a query response time of 2 seconds," you tell them, "Such-and-such a transaction will take no more than 3 seconds to process." That's something an end user can monitor for themselves: "Click enter, and count one-one thousand, two-one thousand, three-one… ah, it's done." That kind of SLA sets an expectation that users can *relate* to. They'll *know* when "the system is slow" because they're measuring the same thing you are. Ideally, *you'll* know of slowdowns before the user, or at close to the same time, because you'll have tools in place to monitor things from the users' perspective.

How It's Done: Synthetic Transactions, Transaction Tracking, and More

This kind of monitoring isn't always easy. It's possible to throw monitoring agents onto end user computers when they're all employees, but what about a Web application, where the end users are actually external customers? They probably wouldn't be excited about having you install monitoring agents on their computers just to track your application's performance.

Instead, modern monitoring tools rely on techniques like *transaction tracking.* With this technique, monitoring components on your end watch an individual transaction as it flows through your systems, literally measuring the time it takes the transaction to be processed. This can be done at a variety of levels of detail. For example, tools usually associated with software performance profiling can get *very* detailed, tracking transactions through individual software modules. At a higher systems management level, you might just track the transaction's start-to-finish time.

Often, this is also done by inserting *synthetic transactions* into the system. Essentially, a monitoring system pretends to be a client, then inserts transactions into the system that will be processed but then later ignored. These allow the monitoring system to more precisely figure out the actual time-to-complete for various transactions. This idea is illustrated in Figure 4.4.

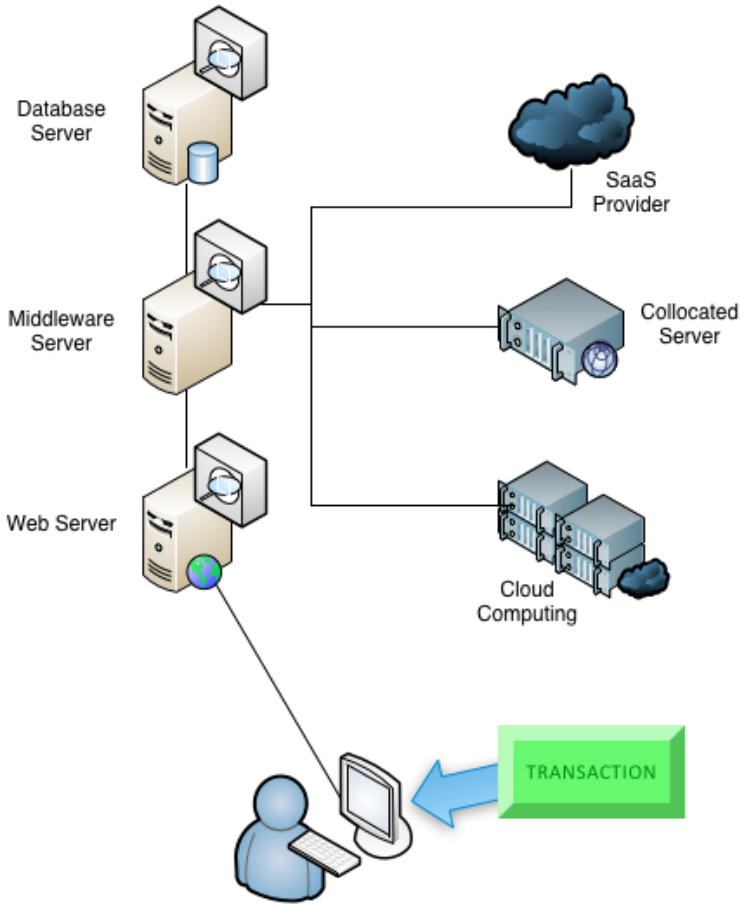

Figure 4.4: Using transaction tracking to monitor the EUE.

There are a lot of variations on these techniques, and a lot of specialized tools that you can acquire to actually implement them. In the end, though, it's important to remember that the entire activity is designed to measure just one thing: the EUE. You're not, at this point, trying to figure out what individual systems' performance is or trying to track down the root cause of the problem. You're simply trying to determine whether there *is* a problem.

Top-Down Monitoring: From the EUE to the Root Problem

The EUE is intended to be an extremely high-level diagnostic; it tells you that "something is wrong." It won't tell you what. For that, you're going to need to go back to the traditional monitoring you've always known and loved, only this time you won't just be watching in a vacuum: You'll be looking for a problem that you know exists.

This is *not* the time to pull out the domain-specific monitoring tools—we've discussed that in prior chapters. You still want to stick with a monitoring system that can monitor *everything* in a single "pane of glass." That doesn't necessarily mean a framework that's aggregating domain-specific tools, either—it means a monitoring system designed specifically to look at each of your systems. With the right understanding of what performance *should* look like at each component level, such a system can quickly tell you *where* the problem lies. *Then* you can dig out the domain-specific tools to troubleshoot the particular problem—again, with the knowledge that there *is* a problem, and that the component you're looking at *is* the one causing the problem.

Deriving the EUE

So why can't you simply use better thresholds on your back-end monitoring to figure out when the EUE is declining? Because EUE focuses on the entire system. The database can be slower, provided that time doesn't cascade through the rest of the system. A slow router doesn't necessarily mean a slow EUE, although in combination with other factors, it might be the tipping point. That's why you need to look *directly* at what end users are experiencing, *then* go looking for the root cause.

Agent vs. Agentless Monitoring

There's a lot of argument in the monitoring industry about the best way to monitor. Do you install agents? Some folks believe so, and feel that agents provide the best and most-detailed information. Other folks don't like installing and maintaining agents throughout their environment, and correctly point out that not every component of a system can even *have* agents installed. Routers, off-premise services, and so forth typically can't support dedicated monitoring software, after all. So one approach is definitely to install agents, as illustrated in Figure 4.5.

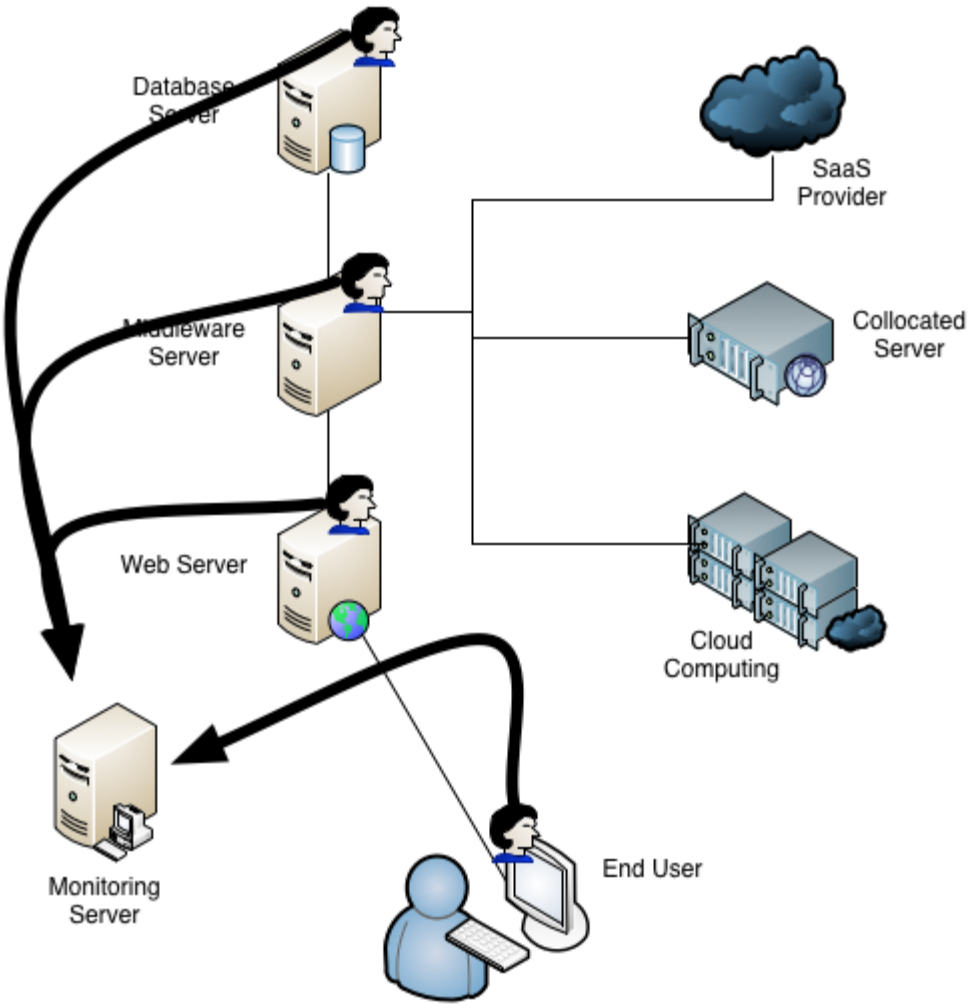

Figure 4.5: Monitoring via agents.

You'll typically have those agents reporting back to some centralized monitoring server or system. Depending on your approach, you might have agents installed on every system where they *can* be installed, potentially even on some end-user computers for spot-monitoring (although that's pretty unusual).

Some monitoring solutions will let you get away without installing agents on every system, and might not even need agents on *any* of your components. As illustrated in Figure 4.6, these solutions typically use external means of picking up on system performance.

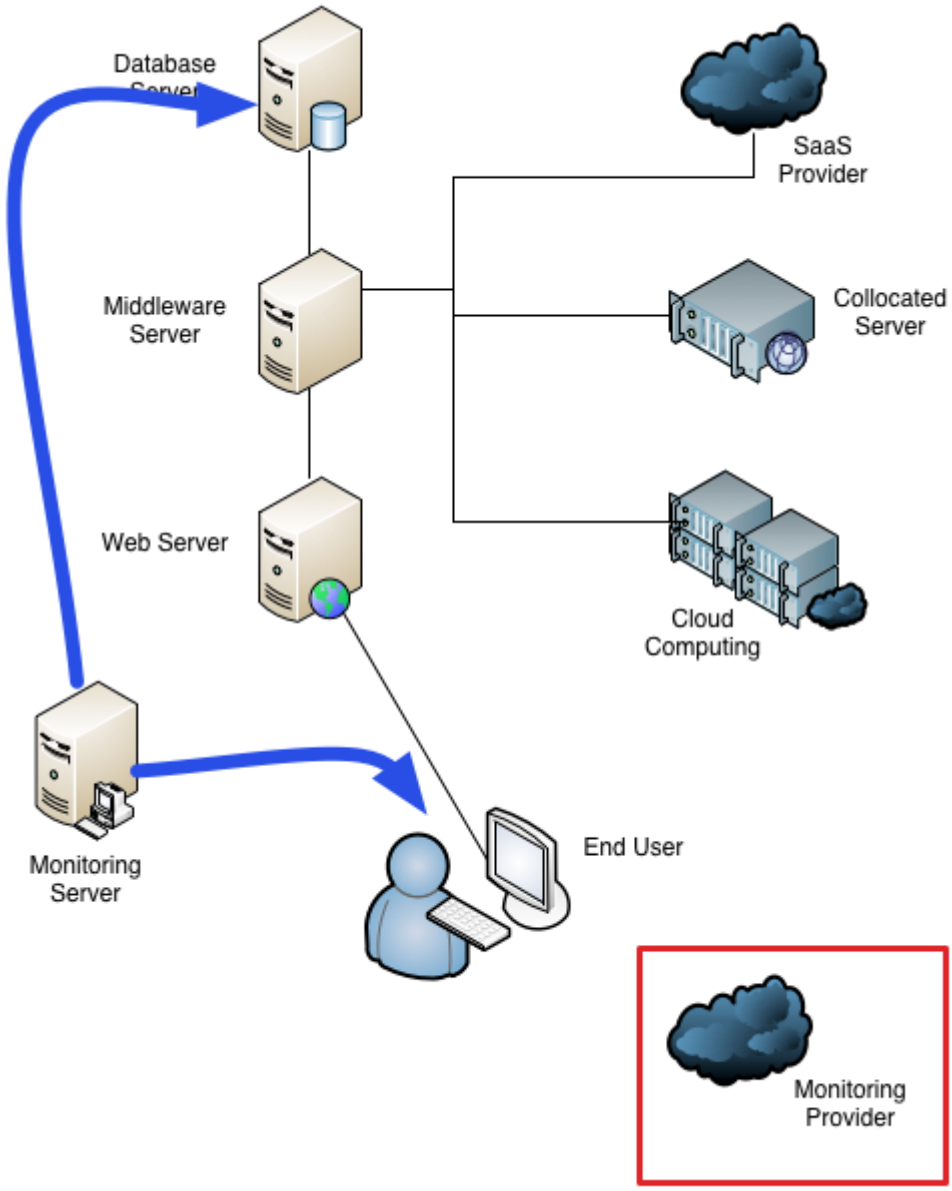

Figure 4.6: Agentless monitoring.

Whether agentless monitoring can pick up as much data, or pick up all the data you need, depends a lot on what kind of components are in your network, and what kind of monitoring techniques and technologies are in use. It's a major competitive point between different vendors, so it's something to pay close attention to.

That "monitoring provider" in Figure 4.6 is my lead-in to a key point about modern applications. You're almost *always* going to wind up with some kind of hybrid system that relies partly on agents and partly on agentless monitoring—because some of what you'll be monitoring won't be in your own data center.

Monitoring What Isn't Yours

The off-premise stuff is where our traditional monitoring falls part. It's unlikely that Amazon is going to give you detailed performance statistics into their Elastic Compute Cloud (EC2), and it's unlikely that Microsoft would do so for Windows Azure. SalesForce.com isn't going to send you database query response times or Web server CPU utilization. Even the hosting company where you've collocated your own servers isn't going to be sending you detailed data about their routers' dropped packet percentages, or any other infrastructure-level statistic.

Yet those numbers *matter* to you. If you have an application that relies on cloud computing components, collocated servers, Software as a Service (SaaS) solutions, or *any* other outsourced component, then the performance of those components *affects your application performance* your EUE. In short, when Amazon feels performance issues, so do your users.

That's where *hybrid monitoring* enters the picture. As Figure 4.7 shows, it usually takes the form of some external monitoring service, which collects key external performance information from major outsource providers—the "cloud components," if you will, with the data collection shown as red lines—and reports back to your central monitoring console, as indicated by the green line.

This is where a lot of what I've been outlining in the previous chapters really starts to come together:

- You need both your internal systems and any external components monitored in the same view. There's no way you can treat your systems as *systems* if you can't get all of the constituent components into a single monitoring space.
- The key competitive point for these hybrid monitoring services is the breadth of external components that they can monitor. Make sure you're choosing one that can monitor everything you've got—including all the outsourced dependencies.
- Monitoring the EUE becomes important because there may well be a lot of fluctuation with your external services and your use of them. For example, you won't care that Azure is experiencing slow response times during periods when your users aren't relying on that service. You only need to pay attention—and be alerted—when *your* users are experiencing a problem.

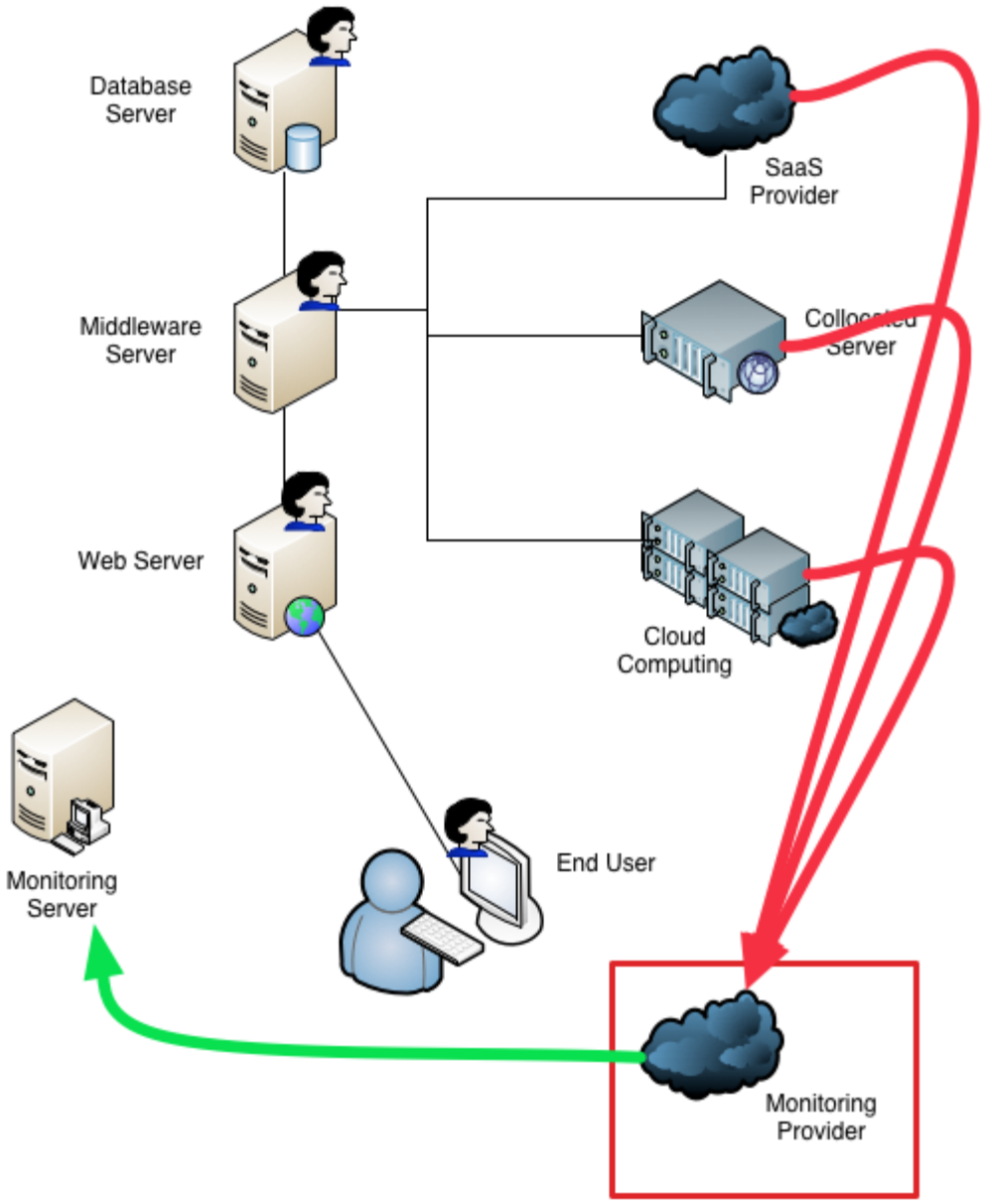

Figure 4.7: Hybrid monitoring.

In fact, this kind of monitoring of external, outsourced components is the key piece that makes many organizations feel they *can't* rely on cloud computing. "How will we manage it?" they ask. "How will we monitor it?" Along with data security, it's probably the biggest question asked when organizations start considering adding "the cloud" to their IT portfolio. *This* is how you'll monitor it: Using specialized monitoring services that add "the cloud" to your single pane of glass. These tools put outsourced components on the same level as your on-premise components, and let you monitor them the same way.

What's interesting is the way in which some vendors are architecting these solutions. Many of them sell on-premise monitoring solutions, which look a lot like what's in Figure 4.7. They actually monitor the cloud components on their end, but deliver that information to you; their solution then collects your on-premise data and presents everything in a consolidated view.

But it doesn't have to be that way. As Figure 4.8 shows, you could also go with a *hosted monitoring solution,* where your *internal* performance data is shipped to the cloud (shown by blue lines), combined with performance information on your outsourced components (red lines), and presented in a single view via a Web portal or some other tool.

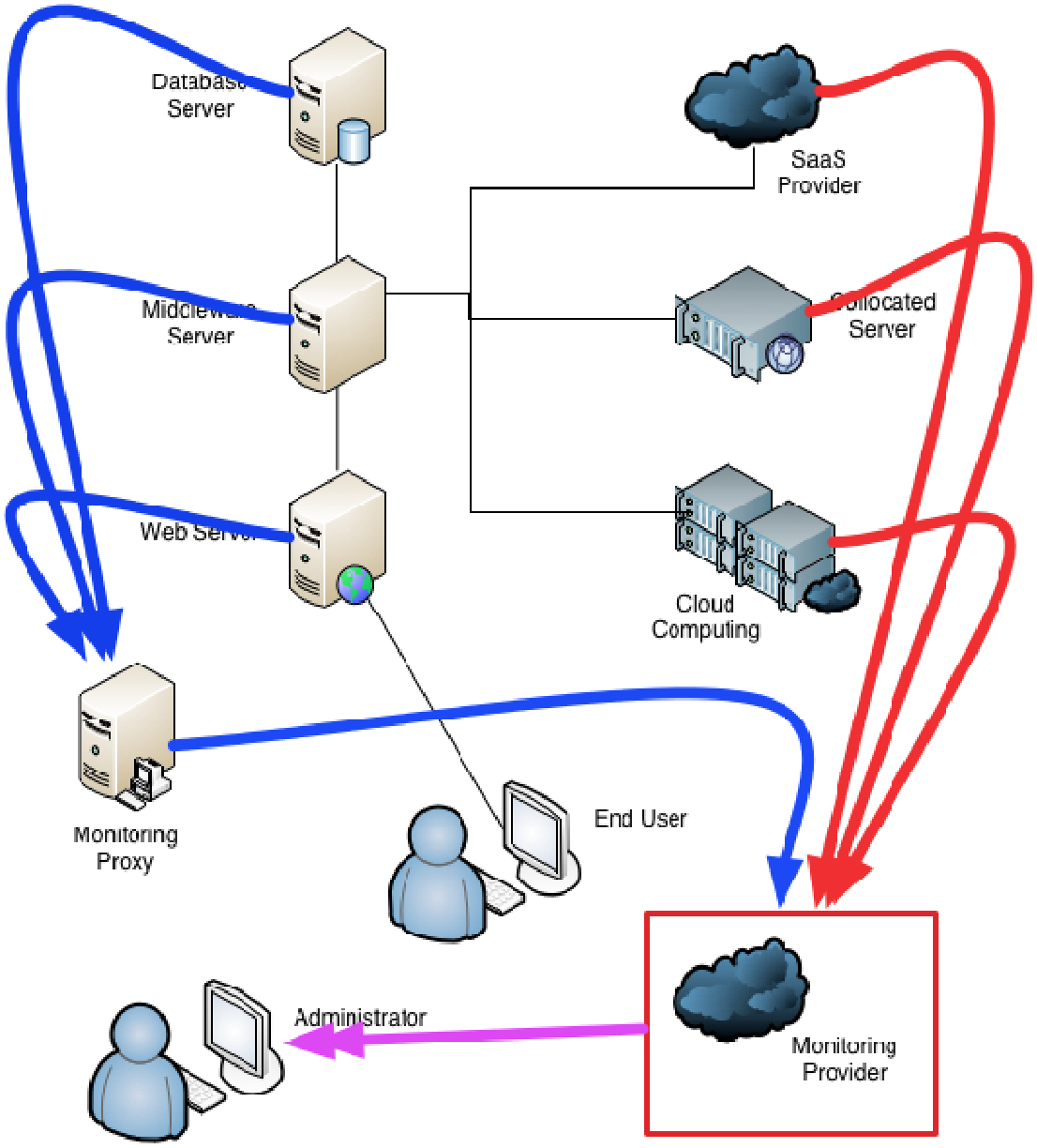

Figure 4.8: Outsourced, hybrid monitoring.

It's an interesting model because it takes much of the responsibility of monitoring out of your hands, and lets you focus on the services you're delivering to your users. It isn't the right model for every organization, of course—but it's an interesting option.

Critical Capability: You Need to Monitor *Everything*

The last really important piece of the puzzle is to make sure you're monitoring *everything.* Every-everything. Take a look at Figure 4.9, which is the example system I've been using all along. Is anything missing? If we monitored each of the components shown, in some fashion, would we be monitoring enough?

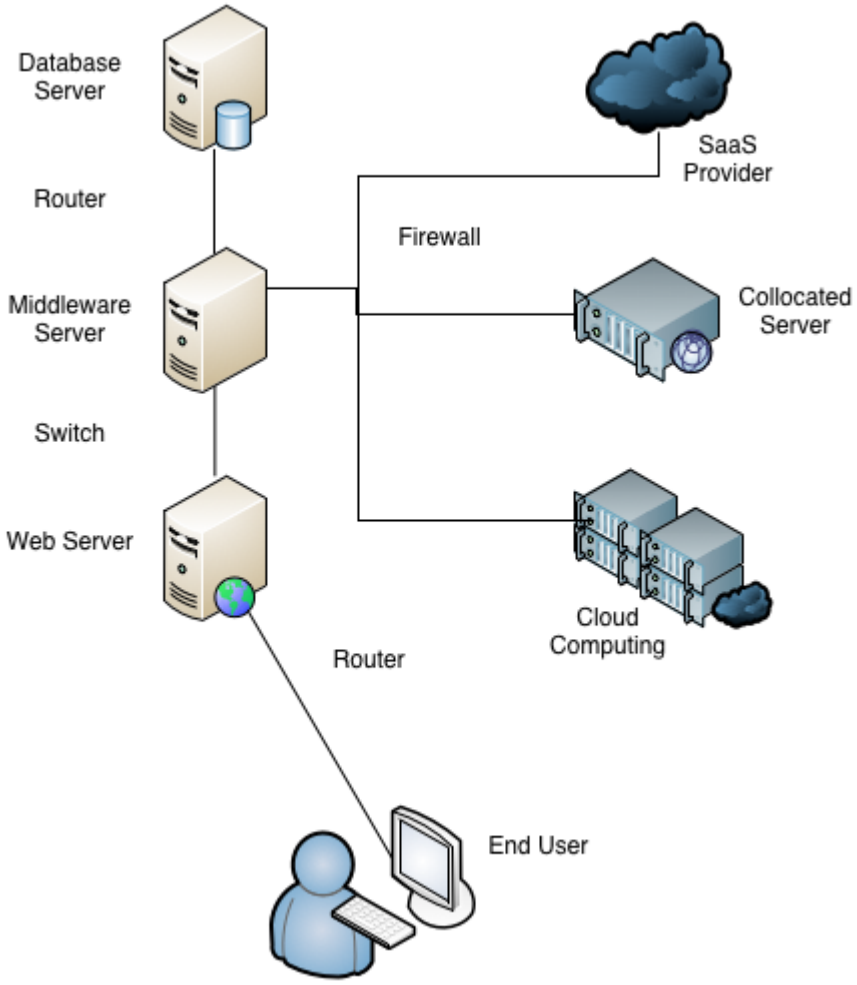

Figure 4.9: Is this everything you'd need to monitor?

Definitely not. There's a lot missing from this diagram, and it's mostly things that can have a massive impact on performance. Take a look at Figure 4.10—it's a bit more complete.

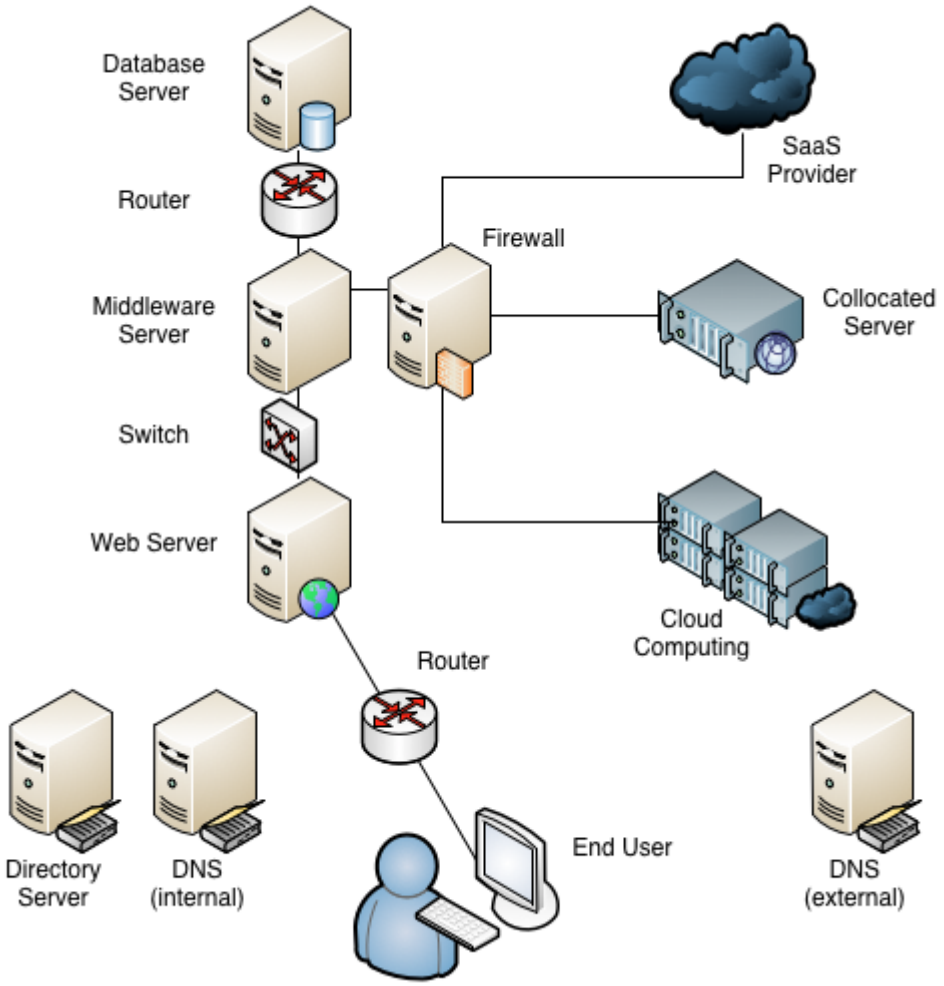

Figure 4.10: Make sure you're monitoring *everything.*

Routers. Switches. Firewalls. Proxy servers. Directory servers. DNS—both internal and external. And probably a lot more. If your EUE is declining, you need to be able to find the root cause—and you can only do that if your monitoring solution can *see* every possible root cause.

This is why a lot of monitoring solutions these days offer automated discovery, in addition to letting you add components on your own. Discovery can find the stuff you're likely to miss because you're not thinking of it as "part of the system." Infrastructure elements like routers and switches. Dependencies like directory services and DNS. Potential bottlenecks like firewalls and proxies. It *all* matters, so it's important that it *all* get onto that "single pane of glass" that you use to monitor overall system health.

Conclusion

Monitoring is the thing that lets us manage SLAs, lets us spot problems brewing, and lets us keep our systems running the way the business needs them to. But traditional monitoring isn't necessarily the only or best way to go about meeting the business' needs. More importantly, as businesses start to rely more and more on external components in their systems, traditional monitoring just can't get all of the necessary facts into a single view.

Hybrid monitoring can. By using a combination of traditional monitoring techniques, cloud-provided performance data, and other techniques, we can get entire systems onto a single view, into a single dashboard, and into a single focus.

Coming Up Next…

In the next chapter, we'll address a fundamental problem that all organizations seem to struggle with: *repeatability.* In other words, once you've solved a problem, how can you solve it more quickly if it happens again in the future? We'll look at turning problems into solutions and improving service delivery in the future.

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.