APM WHITEPAPER

Approaches to APM

Welcome to the New Normal

Approaches to APM

Welcome to the New Normal

BY ROB VOLLUM

Before considering the various approaches (past and present) to APM, it makes sense to understand exactly what APM is and isn't.

What is APM?

Enterprise applications are the lifeblood of organizations both large and small and come in many different flavors. There are public e-commerce applications such as Amazon.com that are responsible for billions of dollars of revenue capture online. Online banking applications process hundreds of millions of transactions per day, moving money, paying bills, and serving as the face of the bank to its customers. Internal manufacturing applications manage inventory, work-in-progress, and finished products. Social networking applications, such as Facebook, have revolutionized personal networking and communication. The list goes on.

Application architecture has evolved over time from monolithic mainframe-based applications, to three-tier, to highly distributed SOA-based applications. During this evolution, there were painful periods where the tools used for performance monitoring failed to keep pace; they were "fighting the last war". This was particularly apparent as applications made the shift to the more highly-distributed model. The user complaints remained the same: "It's so slow I can't do my job!" However, given the increased number of tiers (failure points) and heterogeneous nature of the SOA approach, the information, access, and approach required to solve the problem changed dramatically.

The application performance equivalent of fighting the last war is using familiar, previous generation tools to attack current generation performance problems. As applications evolved into today's highly distributed architecture, previous generation tools remained focused on individual technology silos: web servers, java servers, databases, and so on. These silos were often owned and managed by different groups or departments. This resulted in a fragmented and dis-integrated view of application performance. With no clear picture of what was happening, the standard outcome was a large collection of technology (and often vendor) representatives in a so-called war room, locked in place until somebody figured something out. These "blamestorms" were mainly unproductive, finger-pointing sessions in which the common goal was to prove "It's not me!" And who was helping the user? The same answer: "It's not me!"

Blamestorms are caused by a downward focus. Technology support people work with profilers, query tools, log files, and so on, focusing on individual technology instances. These are the software equivalents of hardware scopes and sniffers. Tests are run to prove that the web server responds immediately, or that queries are sub-second, or that there is very low disk latency. There is no integrated view. That causes us to miss the fact that those sub-second queries are getting called millions of times by a poorly architected component in the Java tier. It's like trying to understand the world by looking through a mailing tube.

The answer is to look up and out, not downward. When the application has performance problems, we need to manage the application as a whole, not just the individual systems. What we need is *Application Performance Management*, or APM.

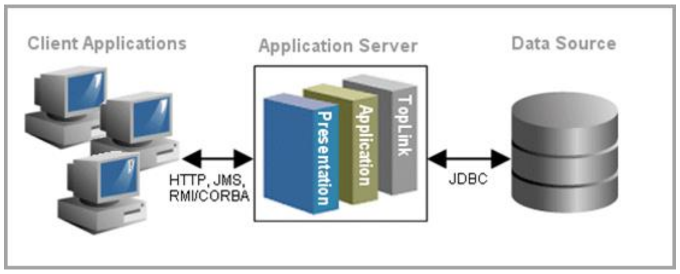
APM is:

- Actively monitoring processes…
- integrated, across all application tiers…
- ensuring the performance of business applications…
- and providing acceptable end-user experiences

Is APM new? No, not at all. In the past, it just wasn't so hard. Before computers, if you had problems with your accounting package, you just yelled at it!

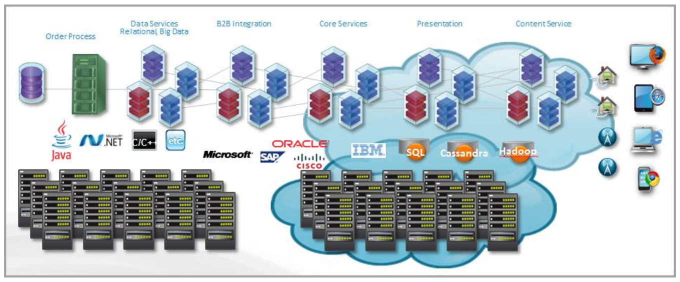
Pre-Computer APM

In the era of monolithic and three-tier applications, before the Internet changed the world, IT departments controlled everything, and there weren't so many moving parts. The negative aspects of a downward technology focus were not so acutely felt; they were manageable.



Three Tier-Based Application

Around 2005, with the commonplace adoption of the Service Oriented Architecture (SOA), applications exploded beyond the capabilities of siloed monitoring tools. At this point, humans lost the ability to compensate for the shortcomings of these tools, no longer able to manually or visually stitch together their non-standard results. There was too much information in too many different formats. Imagine responding to the "It's slow!" complaint with standalone, siloed tools in the following situation:



SOA-Based Application: "It's Slow"

Clearly applications are more complex than ever. They may rely on third-party services (think about car rental offers on an airline's website). There are things we don't control, such as the Internet. There are devices we don't control, such as smartphones and tablets. Having and maintaining an integrated APM solution is no longer optional or just a nice-to-have. Failure to keep both eyes directly on the big picture will result in performance disasters.

What APM *Isn't*

We have a working definition for what APM *is*. When trying to understand a concept, it often helps to also hear what doesn't match. Here are some examples of what APM *isn't*:

- Availability monitoring is not APM. It's nice to know that your site is up, and availability monitoring may be part of an overall solution, but availability monitoring alone is not APM
- While perhaps part of an overall APM solution, simply probing key transactions with a synthetic tool is not APM. This may help alert the team to slowdowns in key transactions, but it's not APM
- Tools or solutions that are not always on are not APM. Part of the definition is "actively monitoring". If you have to anticipate problems or react to them by turning on your tools, you are not doing APM
- Tools that just monitor server statistics, or "container" statistics of JVMs or CLRs, are not APM. There is no "Application" there
- A collection of siloed, non-integrated tools is not APM. Gathered data must be correlated in the same technical language to facilitate understanding across tiers
- Tools that are not able to factor in end-user response time are not APM

Crossing the Chasm: From the Old to the New Normal

Now that we understand what APM is and isn't, how do we apply that knowledge? The next step is to understand the context in which we are evaluating APM solutions. This is true for organizations that have tools already in place and are perhaps thinking of upgrading, as well as for those who are evaluating APM for the first time or perhaps for a new application.

The biggest change agent over the recent short past has been the rapid evolution of application architecture. Gaining traction in the mid-2000s and rapidly gaining general acceptance to the present time is the push to SOA-based systems. These highly distributed systems are fundamentally different than the three-tier (or low-tier) systems they are replacing. In a compressed timeframe, we have clear delineation between an Old Normal and a New Normal. To be successful, APM solution architecture must be tied philosophically to the application architecture it intends to manage. What this means for organizations dependent on their applications for their success, is that APM solutions that may have been workable in the past, are likely not workable now. Vendors who may have been the gold standard then may be hardly relevant now.

At this point, let's take a look at the Old versus the New Normal, identifying the characteristics of each. We'll look at characteristics of Old Normal applications and the tools that evolved to meet those needs and then discuss the characteristics of APM solutions that are required to handle the New Normal.

The Old Normal

There are no exact starting and ending points, but we can consider the Old Normal period to have run for about ten years, from about 1995 until 2005. During this time, applications evolved from monolithic entities into the early stages of the distributed model.

Early in this time range, applications fit the two-tier client-server model. There was not yet a defined need for APM, since applications lived on a single tier and were not overly complex. If there was a problem, it was clear where the problem was. Primary innovative efforts at that time were concentrated on network connectivity and the power of desktop client PCs as the industry moved away from green screen terminals in the first steps toward distributed computing. The first set of products geared towards application performance emerged, focusing on system and network management. Products, such as Tivoli and Patrol, were acquired by companies like IBM and BMC, respectively, in their first steps towards monitoring distributed environments.

Technology moved into the "APM age" with the continued evolution of applications from client-server to three-tier, as the client-server model gave way to online or Internet technology applications. Even with this shift towards application distribution, the middle tier – the "application tier" – remained coarse-grained and large in scope. Unlike the prior client-server architecture where the server tier was monolithic (often a mainframe computer), the middle tier of a three-tier architecture was not constrained to just a single instance of technology. While proprietary technology existed for the application tier (such as Tuxedo, a popular middleware transaction processing system), Java became the primary platform of choice for the application tier, followed by the emergence of Microsoft .NET. Techniques such as clustering servers or instances (JVMs and CLRs) existed for both high availability and horizontal scalability, but it was common and by design that an entire business unit of work, or business transaction, would remain completely within a single instance in the middle tier.

Driven by the added complexity of the three-tier model was the need for a new approach for managing the performance of these increasingly sophisticated applications. Addressing that need, a first generation of APM companies came to market. Leaders included Precise Software, Wily Technologies, and Mercury. It was here that we first saw APM solutions evolving in response to the new shape of applications.

Characteristics/approaches of these APM solutions included:

- Expectation that transactions would generally execute wholly within one instance at each tier
- First application of bytecode injection and other techniques for code-level statistics, providing runtime visibility into monitored applications
- Siloed monitoring with no correlation of transaction data between the tiers (web-app-data-infrastructure)
- No accurate transaction view (transaction purity) within application instances. Sampling provided some code-level insight with no context or actionable insight
- No business context or business impact data. With no transaction purity there can be no business context
- Substantial focus on infrastructure and app server container metrics as key performance indicators
- Network views focused on capacity, not content
- High overhead and cost of collecting data lead to post-event analysis or requirement to recreate performance problems since the solution wasn't running at the time of the problem

These tools provided more insight into applications than had ever been seen previously. To accomplish this, as pioneers they had to pick their technical battles. These tools reflected the architecture of the applications they were monitoring, and they made architectural trade-offs to bring their products to market. For example, most had heavy installation and burdensome manual configuration requirements, in terms of time and effort. Since applications did not automatically bring additional instances online and offline based on load, these products focused on single static instances. There was no always-on transaction tracking nor was there any end-user monitoring, as these techniques and technologies were beyond the state of the art. These tools were best-of-breed, however, and they set the benchmark for the time.

It's been mentioned previously that the evolution of APM tools and technology necessarily follows the evolution of application architecture. Through the end of the Old Normal era, application evolution had been steady and manageable. As the Internet changed what was possible, the rate of change in application architecture increased dramatically. We moved from three-tier applications quickly towards generalized N-tier applications, creating more granular, distributed applications. In about 2005, the shift to SOA-based applications created such dramatic changes as to form a clear line in the technological sand. This is the dividing line between what we're calling the Old Normal and the New Normal.

The New Normal

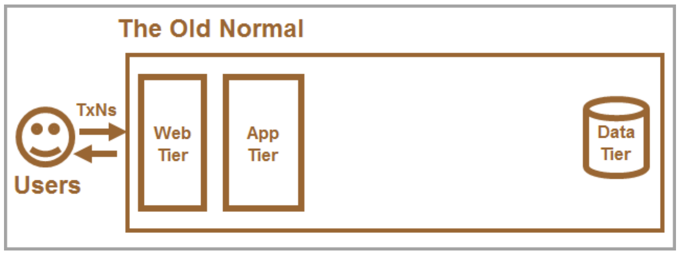
Applications in the Old Normal had simple architectures and were decidedly focused on corporate workers. They were primarily internal systems, used by internal people, for internal things. At the intersection between the Old and New Normal, two game-changing things happened. First, as discussed, application complexity increased dramatically as we moved from a static three-tier model to generalized N-tier complexity. When we add in virtualization and the fact that these N- tiers may find themselves running in different datacenters around the world, it becomes clear that we have entered a New Normal in application complexity. But the second game-changing thing is equally important. No longer were applications primarily inward-focused. In the New Normal, users of a company's applications are worldwide, and these applications represent the face of the organization.

New Normal Applications

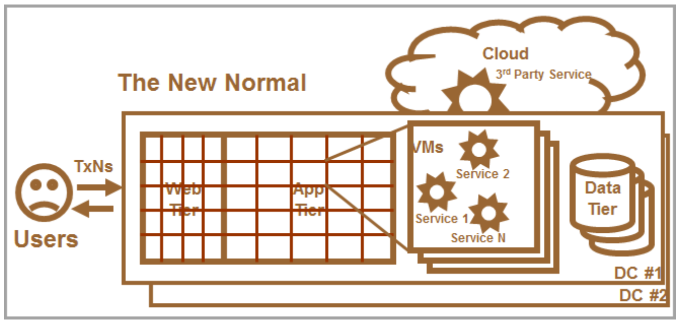
To understand where we're going with our APM approach, we have to understand where we are with our applications; the evolution of APM follows the evolution of the application.

Forget back office order-entry applications where telephone reps in call centers take orders from customers and key in the data, or check processing applications where clearinghouses process boxes of cancelled checks. While these applications were important to the business, they were not real-time critical. After all, if the order entry system was down, the reps could fall back to taking orders on paper. And if boxes of cancelled checks stacked up for a day, who really knew or cared? In the New Normal, companies' customers are directly engaged, performing online transactions with immediate consequences and expectations. If they can't make an order, they'll go to a competitor. If they can't bank online, it's off to another bank. It's no longer the case that engineers in the back office can define a company's technology strategy. Today, a company's customers will do that for them. Customers are hyper-connected, demanding instant gratification from companies they do business with at any time day or night, from any location on the planet. They won't use company-designated browsers, kiosks, or terminals – instead they use their own hardware devices (smartphones, tablets, etc.), and they expect them to perform flawlessly. Customers are demanding that "You *will* be a cloud company." and "You *will* have a mobility strategy." The end result is, like it or not, every company is now a technology company. If not, quite simply they are out of business.

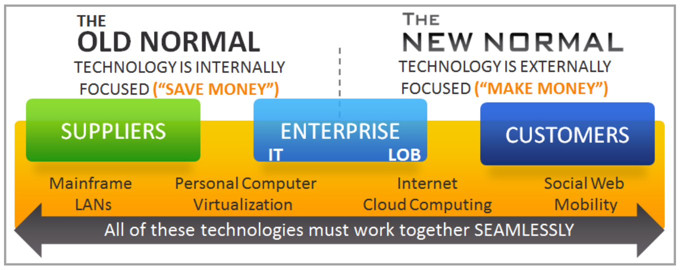
Framing the discussion in architectural terms, applications have evolved from this – static three-tier, inward-facing applications:



to this – dynamic N-tier, outward-facing, collaborative applications:



Framing the discussion in business terms, applications have shifted from those designed primarily to save money, to those designed primarily to make money by winning and keeping customers. In other words, applications are a profit center for today's businesses:


THE OLD NORMAL
TECHNOLOGY IS INTERNALLY FOCUSED ("SAVE MONEY")
THE NEW NORMAL
TECHNOLOGY IS EXTERNALLY FOCUSED ("MAKE MONEY")
SUPPLIERS
ENTERPRISE
IT
LOB
CUSTOMERS
Mainframe
LANs
Personal Computer
Virtualization
Internet
Cloud Computing
Social Web
Mobility
All of these technologies must work together SEAMLESSLY

This is not to discount the value of internal applications. Of course all of the internal workings of a company must still be met. However, the "internal" is just a small part of the path to the "external". And once we hit "external", the environment is out of the control of the internal application and IT teams.

Consider the following classes of today's applications:

- E-commerce and online shopping. In the New Normal, there simply is not a question about whether retailers have an online presence or not. Online retailing makes up a significant portion of retail sales in general. Jessica Kril, of Statistica, reported that in 2012 US e-commerce sales topped $289 billion. Amazon.com, with its online-only model, generated over $61 billion. Online shopping epitomizes the New Normal. Even brick-and-mortar giant Walmart realizes it must compete to survive and is rapidly expanding its web presence. Walmart global e-commerce sales are expected to hit $10 billion this fiscal year, CEO Mike Duke announced at the company's shareholder meeting
- Travel and travel rollup sites. Consumers have real-time access to the best travel deals and are able to book airline, car, and hotel reservations from a single site, which integrates up-to-the-minute availability data from dozens of partner sites. Stuck in an airport with a flight that's running late? Mobile travel apps can find and confirm alternate flights. Want a better seat assignment? Change it in real time with your airline's mobile app
- Banking and finance. Online bill pay is almost Old Normal at this point. Innovations to banking applications in this highly competitive field strive to keep customers happy and engaged. Mobile applications provide instant and up-to-date account information and provide innovative value-add features. For example, forget about going to the bank to deposit that check. Simply take a picture and upload it to your account, and it's done. Out to dinner and need to pay back your share to the person who picked up the check? No problem – use your smartphone to make a payment to his smartphone, and it's done. Small business? Use your smartphone to accept and process credit card purchases
- Insurance. Of course policy holders can log in to pay their premiums online or to ask for a rate quote. However, in the rush to serve customers better, the bar has been raised even in this conservative industry. For example, online insurer Esurance lets policy holders submit automobile claim information from the site of an accident, from their mobile app. Use the app to take a picture of the damage, upload it to your account, and the claim has begun. Get daily updates (with pictures) of the repair process directly to your computer or smartphone

The technical complexity and integration within these new applications, along with their business criticality, are staggering and drive the requirements for the evolution of the New Normal in APM.

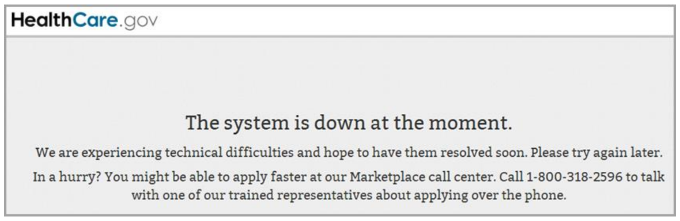
New Normal APM

For many reasons, product cycles often move slower than technological cycles, especially when taken by surprise by fundamental game changers such as the Internet and its related technologies. That was the case for APM, and the pioneering first generation APM solutions (founded or acquired pre-2005) could not keep up. SOA, the cloud, and virtualization, to name just three game-changing technologies, swarmed the business world, and support for these simply was not designed into the first generation of APM products and tools. Suddenly there was a disconnect in what applications required and what APM tools delivered.

That disconnect between what applications now demand and what Old Normal APM technologies can deliver leaves those who use them exposed. This exposure could be financial, could cause damage to the organization's reputation and credibility, or both. For financial impact, consider a 2013 survey, in which over 300 business executives were asked to assess the impact of using silo-based (Old Normal) APM technologies. This is what they had this to say:

- $10.8 million: the average cost per organization of a major technology performance issue
- 21 days: the average time for business operations to return to normal after such an issue
- 75%: the percentage of executives who said the number of technology performance issues are staying the same or increasing

For reputation and credibility impact, consider this very public failure from October 1, 2013: the rollout of the Affordable Care Act (ACA, or commonly called "Obamacare") healthcare enrollment site, www.healthcare.gov. After spending over $400 million on development of the site, it was unable to launch due to massive and overwhelming performance problems, with no good idea about what was causing them. Reports later revealed that pre-launch tests had confirmed the site could only handle 1,100 users at most, with no understanding of why. The negative publicity manifested itself as ongoing, front-page news across the U.S., as millions of citizens had a legal, hard deadline to purchase the coverage offered on the site. According to documents from the House Oversight and Government Reform Committee, on day one of the launch just six people managed to use the site to successfully purchase coverage. One month after the launch, messages such as the following were still common:


HealthCare.gov

The system is down at the moment.

We are experiencing technical difficulties and hope to have them resolved soon. Please try again later.

In a hurry? You might be able to apply faster at our Marketplace call center. Call 1-800-318-2596 to talk with one of our trained representatives about applying over the phone.

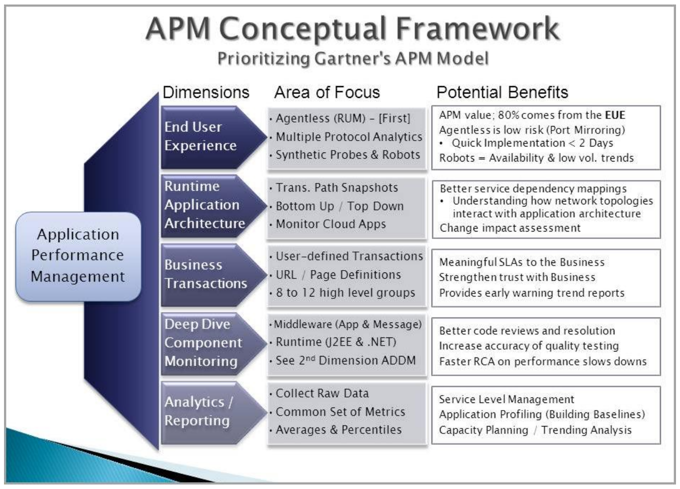
Crashed Due to Performance Issues – Oct 30, 2013

This is a clear example of what happens when using Old Normal tools and processes to manage critical New Normal applications.

Recognizing the high bar that is set by today's complex, highly distributed applications, let's consider the requirements for and characteristics of New Normal APM solutions:

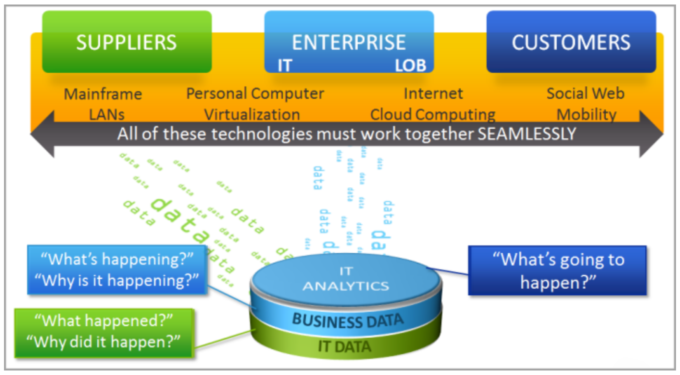
- Real end-user experience is king. In some sense, nothing else really matters. If individual tiers or components of the application are showing "all green", but the end-user experience is unacceptable, then we have failed. Tools and procedures must be in place to capture details about and manage the performance of applications out to the end user, period
- The system must be always on and transaction pure, meaning that we are capturing 100% of user transactions, end to end, with automatically correlated data across all application tiers. Anything less is just sampling. For true performance management (just like with true financial management) we need the raw data
- Application instrumentation – out to the edge, meaning browsers and mobile apps. Code-level visibility allows us to peer into the running application to find bottlenecks and spot architectural and performance anti-patterns
- Tools for application-aware network performance monitoring (aaNPM). The application is more than program code; it traverses complex network paths and devices, many of which directly influence application behavior, Network visibility must be correlated with application visibility
- Performance data must combine with application/business context data providing real-time business impact analysis, effectively mapping application performance to the business bottom line
- Ability to track and manage business transactions, which represent key areas of interest or focus within an application. For example, on a travel site, an interesting business transaction might be "search by travel destination". Capturing such information may help spot trends, for special deals or perhaps new travel packages
- Performance data must combine with application-aware infrastructure data for a complete correlated application performance picture
- Solution must be able to discover and display the runtime architecture or transaction flow of the application
- Low overhead and a low cost of gathering data. New Normal APM tools must be tightly integrated into the applications they monitor and must be designed to "do no harm". The long-term storage of detailed application performance data enables the coming need for IT analytics, or the process of predicting what's coming based on past trends
- Able to scale easily out to thousands of nodes in highly distributed application architectures. Must adapt seamlessly to multiple datacenters and cloud-based partners
- Low to zero configuration requirements with designed-in elasticity supporting modern applications that grow and shrink dynamically based on load
- On-premise and cloud-based hosting of performance infrastructure
- An open architecture to be able to extend the solution in ways not previously envisioned. These could be APIs for creating or customizing agents or service interfaces for connecting with other tools and existing customer investments (e.g. enterprise reporting or alerting tools)
- Designed with DevOps in mind, sharing common data in appropriate formats, across the lifecycle of development, test, and production/operations

A significant list, indeed. As validation, these characteristics of successful New Normal APM tools align with and extend the five dimensions of the APM Conceptual Framework as enumerated by Gartner in its 2012 *Magic Quadrant for APM* report:


APM Conceptual Framework
Prioritizing Gartner's APM Model
Dimensions
Area of Focus
Potential Benefits
Application Performance Management
End User Experience
· Agentless (RUM) – [First]
· Multiple Protocol Analytics
· Synthetic Probes & Robots
APM value; 80% comes from the EUE
Agentless is low risk (Port Mirroring)
• Quick Implementation < 2 Days
Robots = Availability & low vol. trends
Runtime Application Architecture
· Trans. Path Snapshots
· Bottom Up / Top Down
· Monitor Cloud Apps
Better service dependency mappings
• Understanding how network topologies interact with application architecture
Change impact assessment
Business Transactions
· User-defined Transactions
· URL / Page Definitions
· 8 to 12 high level groups
Meaningful SLAs to the Business
Strengthen trust with Business
Provides early warning trend reports
Deep Dive Component Monitoring
· Middleware (App & Message)
· Runtime (J2EE & .NET)
· See 2nd Dimension ADDM
Better code reviews and resolution
Increase accuracy of quality testing
Faster RCA on performance slows downs
Analytics / Reporting
· Collect Raw Data
· Common Set of Metrics
· Averages & Percentiles
Service Level Management
Application Profiling (Building Baselines)
Capacity Planning / Trending Analysis

Prioritizing Gartner's APM Model
Larry Dragich, APMDigest.com March 2012

Understanding current and upcoming application architecture means that we can understand where we need to go with our APM evolution. As always, learning from the past (generation one APM products) helps us to avoid making the same mistakes. In the near-term future, this tight coupling of applications and APM solutions will lead us to the next step in performance management, from reactive to proactive to predictive, with IT analytics:


SUPPLIERS
ENTERPRISE
IT
LOB
CUSTOMERS
Mainframe
LANs
Personal Computer
Virtualization
Internet
Cloud Computing
Social Web
Mobility
All of these technologies must work together SEAMLESSLY
data
"What's happening?" "Why is it happening?"
"What happened?" "Why did it happen?"
IT ANALYTICS
BUSINESS DATA
IT DATA
"What's going to happen?"

Predictive APM with IT Analytics

With the complexity and importance of today's applications, nothing less than this New Normal APM will suffice.

A Glimpse at Data Collection Techniques

All of this APM data has to be gathered from the application somehow. Is there one way, a best way? As one would expect, with such a wide base of technology to support and cover, there is no one best way to gather performance data. Here are some options, at a high level. Any review of New Normal APM solutions will encounter these approaches or philosophies.

Agent-Based vs. Agentless

In the APM world, an agent is a software component that incorporates (injects) itself into the monitored application. In an agentless model, the application itself is unchanged, and a software component interacts with the running application from the outside. Each approach has its advantages and disadvantages. Note: there are different agentless approaches; some work to gain visibility into the running application by observing system (OS) behavior, similar to the agent-based solutions. Others take a probe-based approach, basing their visibility on network data. In this comparison, we look at the application visibility scenario (probe-based agentless solutions are discussed later):

- Benefits of an agent-based solution: tight coupling/integration of the agent into the app for expanded visibility; specialized data gathering capabilities such as business context in addition to application performance details; detailed data gathering; very low overhead; ability to be part of the executing process
- Cost of an agent-based solution: added overhead to the monitored application (it's extra code running in the application); changes are required to the application startup script to add the agent; added code always has the chance to cause execution problems; consideration needs to be made for agent upgrades and patches
- Benefits of the agentless solution: ease of setup – it's all within our control. There is no need to touch the app servers. Administration and maintenance can be simplified
- Cost of agentless: limited depth to the collected data; additional cost to collecting data (remote procedure calls or similar as opposed to integrated data collection); data gathering is arms-length (RPC or other network connection). Little or no possibility of specialized content at a reasonable overhead price

For adequate APM coverage in today's applications, an agent-based solution is required. The detail required for transaction purity and business context can't happen with an agentless solution; the integration simply isn't there. Agentless solutions add "color", for example, in providing monitoring data into devices such as load balancers and other application acceleration hardware. But for the deep dive into applications, agent-based solutions are required.

Application Centric vs. Network Centric

APM has had a broad definition over the past decade. One of the flash points has been over solutions that are application-aware versus those that are network-aware. The difference here is that application-aware solutions provide a deep dive into application code as it executes, with access to business context in the execution environment, while network-aware solutions provide a broader infrastructure view by applying intelligence to network packet analysis, to profile the application footprint across all heterogeneous hardware tiers. While comparisons are provided below, it needs to be clearly stated that an APM strategy is not complete without a tool that provides an application focus. That's what APM is all about. Details:

- These are complementary approaches, competitive only in the sense that there are limited budget dollars available for APM solutions. They provide different information. Ideally, both would be applied for complex applications
- Strengths of an application-focus: the main benefit of an application-focused approach is the deep dive into the application at the code level. An application focus, best supported by agent-based data collection, allows us to see into the application as it's running, gathering data about the execution environment (e.g. the containing JVM or CLR), execution traces of the transactions through the code itself (transaction purity), and related business context contained in the transaction. It also allows for automatic application mapping, or transaction flow, with the capabilities to gather and correlate infrastructure performance data with application performance data. Triage at the code level with root-cause analysis is directly supported with an application-based approach.

  A secondary benefit is that an application focus allows us to take a DevOps or lifecycle approach to product release, providing team members in each of the dev, test, and prod groups with relevant information, all from the same detailed base of performance data. The challenge faced by an application focus is that technology coverage is dependent on the vendor's engineering team. Since the application focus is primarily accomplished with an agent-based approach, coverage depends upon which agents exist. Java, .NET, and PHP are common today. Coverage of a different platform may require a new release from the vendor or a good agent development toolkit for self-service by the customer
- Strengths of a network-aware approach: this approach is accomplished through network taps and port mirroring, which can happen virtually anywhere along the application's infrastructure. Therefore, coverage of the entire application footprint is much broader than with an application focus. For example, with a network-aware approach, components such as load balancers, switches, and custom hardware components can be included in the overall footprint. By analyzing the packets traversing the network, network aware tools have visibility into what technology is running where. From a troubleshooting perspective, fault domain isolation capabilities are enhanced with this approach. The primary challenge faced by this approach is that there is no visibility into the application. So while we may be able to identify that there is a bottleneck on a particular JEE tier, it's not possible to dive in to determine the root cause

Transactional vs. Technology-focused

At the heart of this discussion is the value placed on transaction purity (tracking every transaction) versus just the discovery of hotspots via sampling technology. Hotspots, particularly in a production environment, are helpful for problem identification. We can easily see that "method X is taking 35% of CPU resources". However, hotspots alone do not lead easily to problem resolution or root-cause analysis. For that, we need transactional context. Which particular transactions, users, or access paths are bringing us to the situations where method X is misbehaving?

More considerations: keep in mind that it's the current state of application evolution that drives (sets requirements for) APM evolution. With application teams driven by agile methodologies and DevOps-based rapid product release cycles, acceptable APM solutions must address more than just one of the three families (dev, test, prod). This need from the application side drives the requirement of transaction tracking and purity for us. Today's APM solutions must track at the transaction level; otherwise they are ignoring the full lifecycle.

Ideally, an APM application-focused tool would not choose one approach versus. the other. This is a case where "and" is better. Managing transaction purity though judicious use of agent-based instrumentation, combined with inexpensive snapshotting technology to fill in the gaps, is a strong approach.

Real-User vs. SaaS-Based (Synthetics)

Absolutely and fundamentally clear in the New Normal of APM is that the end-user experience is king. Accordingly, any APM solution worth considering must have an integrated end-user component.

An end-user focus by itself (having no integration into datacenter-based APM tools) is not an APM strategy. It will be useful, but it won't be enough. A SaaS-based (or synthetic) approach is a good way to track availability of the application from the outside world, and it helps set benchmarks for SLAs on key transactions or from key locations in the world – all of which are useful. A real-user approach will let us see the live application footprint from around the world -- again, useful. However, what we have in this case is "necessary but not sufficient". Neither approach does anything to help diagnose problems beyond the browser.

When integrated into a full APM solution, end-user experience monitoring completes the picture. It provides visibility from the browser or mobile app, over the Internet or cloud, and into the one or more datacenters hosting the backend application. SaaS-based synthetic monitoring can be used, again focusing on availability and key SLAs, but real-user monitoring is required for true APM. Real-user monitoring is the only way we can assess the true effectiveness of the application, as it can tell us: how many people are abandoning the site, how many conversions we see, what the actual click path is followed by users, what errors are real users experiencing, and so on. Any APM solution without real-user monitoring is missing a key element required by today's New Normal.

Summary

Out of necessity, the evolution of APM follows the evolution of applications themselves. For a time, applications evolved slowly, and APM solutions followed along with ample time to grow and adjust. Around 2005, however, due to the broad acceptance of Internet technologies, combined with the emergence of applications designed in a new SOA-based way, a fundamental shift occurred. The change was so dramatic that we've made a distinction in this paper between the Old Normal and New Normal. In this shift, we've seen that technology has become the primary connection with customers. This means that, like it or not, every company must become a technology company. Furthermore, we see that an overall application performance strategy is required, not just a cloud strategy or a mobility strategy, but an overall performance strategy. End-to-end performance is a new business imperative. Performance is not just about IT anymore.

From an APM perspective, this shift brought difficult times to Old Normal APM companies and their approaches to APM. For organizations to be successful today, APM is no longer optional for key corporate applications. When evaluating APM vendors and solutions, organizations must ensure that their vendor of choice has a comprehensive New Normal solution and a plan to keep it current as application needs continue to evolve.

Compuware Corporation, the technology performance company, makes a difference by providing software, experts and best practices to ensure technology works well and delivers value. Compuware solutions make the world's most important technologies perform at their best for leading organizations worldwide, including 46 of the top 50 Fortune 500 companies and 12 of the top 20 most visited U.S. web sites. Learn more at: **compuware.com**.

Compuware Corporation World Headquarters – One Campus Martius – Detroit, MI 48226-5099

©2013 Compuware Corporation

Compuware products and services listed within are trademarks or registered trademarks of Compuware Corporation. Oracle and Java are registered trademarks of Oracle and/or its affiliates. Other names may be trademarks of their respective owners.