CHAPTER

# APPLICATION PERFORMANCE MANAGEMENT AND USER EXPERIENCE MANAGEMENT

# 12

*When you see something that is technically sweet, you go ahead and do it and you argue about what to do about it only after you have had your technical success. That is the way it was with the atom bomb.*
**J. Robert Oppenheimer**

## INTRODUCTION

Modern distributed **applications** are the most complex example of business applications on the planet. They can be supported by hundreds or thousands of hardware and software subcomponents and are heavily network-dependent for performance and availability. In managing these applications, it is easy to lose sight of the forest and focus on the trees. In this case, the forest consists of business- and user-facing applications, while the trees are the underlying hardware and software infrastructure silos. Visibility to both, plus **analytics** capable of correlating the two in context to the end-to-end transaction, are the most important elements of application performance management (APM).

However, all too often, complex applications are deployed first and application management is an afterthought.

Traditionally, applications were supported with a combination of silo monitoring and tribal knowledge—knowledge built from the experience of the hands-on specialists supporting a company's enterprise applications. Now, however, in virtually every company and industry the number of enterprise applications[1] is growing. Only 30% of IT organizations are supporting 10 or fewer such applications, 25% are supporting between 11 and 50, and 15% support more than 150. The sheer volume and complexity of applications deployed in the average company have outstripped IT's ability to rely on human skills and knowledge alone to support them.

The industry solves the problem of complexity, in part, by breaking the IT ecosystem (and IT support) into consumable chunks or silos, with each technology team covering a small subset of an increasingly broad technology panorama. The net result is that most IT practitioners—and most **management** tools—tend to focus on a discrete subset of an ever-expanding application fabric.

The problem is that applications (not simply infrastructure) are the core supporting digital **services** generating billions of dollars annually in revenue. From this perspective, all the infrastructure and the billions of dollars expended annually on enterprise hardware and software boil down to one endgame: IT exists to meet a business need, not simply to create code, manage databases, provision servers, or

[1]An enterprise application is defined by EMA for APM purposes as "a complex, distributed application used company-wide" (versus a desktop or mobile application). A single enterprise application may well span multiple platforms, including mainframes and/or cloud services. APM investments capable of covering complex, business critical use cases such as these—encompassing tiered, hybrid, and componentized application architectures—are very likely to support less complex application types as well.

**Application Performance Management (APM) in the Digital Enterprise.** http://dx.doi.org/10.1016/B978-0-12-804018-8.00012-7

configure networks. Meeting that business need requires **application management**, not simply infrastructure management. Traditional enterprise management products, many of which have domain-focused roots, are still adapting to this reality, with a strong focus on silo management. And though management products that address databases, networks, and servers are still a part of the big picture, the picture itself is also far bigger than it was in the past.

This chapter focuses on the big picture—the enterprise applications executing on top of the silos. It details techniques, capabilities, and products required to evolve from silo-based to application-focused monitoring and management. APM and user experience management (UEM) are the two primary product types in this class. If applications are the new *lingua franca* enabling businesses to digitally communicate with customers, partners, and providers, these types of solutions provide IT organizations with the tools they need to effectively fulfill the service provider role.

## APPLICATION PERFORMANCE MANAGEMENT: MULTIDIMENSIONAL VISIBILITY TO APPLICATION EXECUTION

APM requires visibility not only to underlying silos, but also to the executing transactions, applications, and/or services running atop the silos.

Managing applications versus simply monitoring applications requires automation supporting two primary functions:

- Monitoring and optimizing performance of the end-to-end application or transaction
- Troubleshooting and root cause analysis of performance or availability issues

While the majority of UEM solutions have functionality supporting the first, true **APM** solutions can do both. From this perspective, one key differentiator between the two product types is breadth of visibility. While UEM solutions can watch end-to-end execution, APM solutions can also see the interrelationships between the executing transaction/application and the supporting hardware, software, and other infrastructure elements. For the APM and UEM buyer, the line separating the two types of products is often blurred. For example, some UEM solutions do have the ability to correlate performance to underlying root cause to a degree, and many have features that APM products lack, such as specific visibility to end user actions. Likewise, leading APM solutions also encompass a level of UEM functionality, most notably the ability to trace transactions across the execution stream.

However, for the purposes of this book, which essentially focuses on capabilities versus specific products or vendors, keeping the two goals of performance management and problem resolution in mind provides a starting point for understanding the breadth of the APM challenge. Creating a management fabric sufficient for supporting automated troubleshooting and root cause analysis across complex application systems requires multidimensional visibility to the entire application ecosystem, and, often, the contributions of multiple product families.

Capabilities such as those characteristic of **configuration** and **metadata** repositories, **application discovery/dependency mapping** solutions, middleware monitoring, and code analytics offerings automate the process of detecting and modeling relationships among applications and infrastructure elements across the IT ecosystem—in effect, modeling an application execution fabric. Without such a comprehensive, multidimensional approach, activities like **change management** and root-cause

analysis become increasingly problematic. In lieu of automation, they rely on human expertise, domain specialists, and tribal knowledge, all of which tend to be siloed, fragmentary, and inconclusive.

Increasingly, however, a new class of APM products targeting both application experts and, often, **line of business** application owners as well is emerging. The overarching role of these tools is to develop a comprehensive understanding of the forest in context with the trees—in other words, to build visibility to the interrelationships and dependencies supporting an application across both vertical infrastructure and horizontally flowing execution paths. This integrated view—or **topology model**—is an essential basis for root-cause analysis. Otherwise, application troubleshooting is essentially a matter of trial and error.

## ANALYTICS

Any discussion of APM must start with a discussion of the role of analytics in supporting the APM function. Analytics are the core differentiators for vendors of APM and UEM solutions. Although the remainder of this chapter details the product types and instrumentation points supporting APM and UEM disciplines, the true "secret sauce" of any such solution lies in the heuristics, algorithms, and analyses conceived by the computer scientists creating the product.

Hardware and software systems generate execution metrics that can be gathered and analyzed for insights into system health. The same generic metrics and execution data are available to virtually every management solution and vendor. The true differentiators of a given product lie in two areas:

- Its ability to generate unique information from the data available to it, whether this is via standard metrics or proprietary metric generators such as software agents installed on the system
- Its inherent ability to analyze this information in a unique way to draw conclusions relevant to the process of application support

Enterprise Management Associates (EMA) calls these capabilities **Advanced IT Analytics (AIA)** and defines AIA as having the following three characteristics:

- AIA should be cross-domain and not restricted to just silos such as network, systems, application, database, or even business outcomes.
- AIA should assimilate many different data sources whether from third-party monitoring tools, unstructured log files, protocol-rich data sources, or ideally, all of the above.
- Over and above this data, AIA requires the application of advanced heuristics, such as machine learning, advanced correlation, anomaly detection, or predictive trending.[2]

State-of-the-art APM capabilities supplied by top vendors include ongoing self-learning of environmental norms, proactive notification when key metrics indicate an impending issue, and real-time topology modeling supporting troubleshooting/root cause analysis.

All of these capabilities, but particularly the analytics supporting them, are implemented differently by every vendor and are almost always patented. The most sophisticated APM solutions can also link tiny changes and glitches in execution to failures occurring minutes or hours later. These so-called rolling failures are extremely difficult to detect via manual analysis since the root cause of an issue and its

[2]Enterprise Management Associates, *Advanced IT Analytics: A Look at Real Adoptions in the Real World*. Available for download at www.enterprisemanagement.com.

manifestation may occur far apart in time. Sophisticated analytics solutions, however, can detect the fact that a given event or series of events has, in the past, set the ball rolling by initiating a chain of events that eventually results in a performance problem or service interruption.

The data and metrics collected at instrumentation points across the application ecosystem are essential to performance monitoring and root cause analysis. However, analytics capable of transforming data and metrics into an application-focused report or dashboard[3] are what separates actual application monitoring from relatively simple silo monitoring. Analytics add the context necessary to understand the role of each moving part in the end-to-end execution environment, a viewpoint that is absolutely critical for rapid—and eventually automated—problem determination and resolution. Without this context, these solutions would simply be operational data stores versus true tools capable of insight into application ecosystems.

## APPLICATION PERFORMANCE MANAGEMENT AND USER EXPERIENCE MANAGEMENT, COMPARED AND CONTRASTED

There are a wide variety of tools supporting these processes. Some classes of application-facing tools monitor execution from the perspective of the servers running the software. Some quantify performance from the perspective of the end user (**user experience**) by monitoring web server interactions, monitoring the endpoint (desktop, **mobile device**, etc.), or running **synthetic transactions**. Some tools monitor performance from the perspective of the connective tissue supporting the application—message queues, **enterprise service buses (ESBs)**, or **application programming interface (API)** gateways. Some combine all of the above in large-scale suite or platform products that are designed to automate virtually every monitoring/management aspect of a large-scale data center. What these tools all have in common is the ability to analyze and report on at least one essential dimension or element of the end-to-end application.

While APM and UEM are certainly related, they are significantly different in terms of the types of information they provide. There are multiple areas of overlap:

- Both provide a quantification or approximation of **application performance**.
- Both provide insights on availability.
- Both focus on monitoring/measuring an end-to-end, cross-silo service versus a simple server, database, or network link.

In comparing the two product categories, APM solutions are more comprehensive in their technology coverage and richer in their analytical capabilities. UEM solutions, in contrast, typically focus on quantifying the user experience from a particular perspective—a single slice of the user experience, such as the network, the web server, or the transaction. They provide an automated approach to approximating application performance as experienced by end users.

APM solutions gather and analyze metrics from a wide range of viewpoints across the application ecosystem. They quantify performance from the perspective of the ecosystem versus the perspective of

[3]For the purposes of this chapter, the terms “applications” and “transactions” are used interchangeably since the key point is to describe products capable of creating an end-to-end (multidimensional) versus silo (vertical) perspective on the data center and ecosystem.

the user; for example, an organization's data center or a consumer cloud service that a given APM solution is monitoring. Because of their insights into execution, their unique value proposition lies in their ability to support troubleshooting—detecting and pinpointing the source of a performance or availability problem to automate the process of root cause analysis. In other words, while both APM and UEM tools can be used to detect performance or availability issues, APM solutions are purpose-built to support root-cause analysis as well.

UEM solutions, as the name implies, quantify performance as experienced by a user. While the user is typically considered to be a person sitting at a keyboard, in the case of machine-to-machine interactions, the user could just as easily be another device. Although UEM solutions focus on top-level, end-to-end performance versus troubleshooting or root-cause analysis, the types of information they do provide is extremely valuable. Often, UEM solutions are used as the proverbial "canaries in a coal mine," early warning systems for impending problems.

They also focus on viewpoints that APM solutions often lack—notably functions such as endpoint monitoring (in which agents are installed on a desktop or mobile device) and browser injection (in which a small monitoring script is injected into the user's browser during execution), both of which have visibility to user actions. The capabilities supporting user management are aimed at answering questions such as these:

- Who are the application's users?
- Which applications are most critical to the business?
- Who is impacted if an application fails?
- What are the application's users doing and how are their devices responding?

UEM solutions provide insights garnered from a wide variety of monitoring methods. **Active monitoring** solutions incorporate robot scripts that quantify end-to-end performance, regardless of whether or not users are on a system. **Passive monitoring** solutions focus on real human interactions versus purely technical component behaviors. Both approaches lend themselves not only to IT-relevant metrics and **key performance indicators (KPI)**, but also to metrics supporting **service-level agreement (SLA)**[4] monitoring and insights into business-relevant information, such as online sales conversions. More detailed examinations of both APM and UEM product categories and features are provided later in this chapter.

## ON-PREMISES AND SOFTWARE AS A SERVICE–BASED APPLICATION PERFORMANCE MANAGEMENT SOLUTION

In purchasing an APM solution, form factor is one of the first considerations that must be decided. The two most common form factors are shown in Figs. 12.1 and 12.2.

In either case, APM solutions are designed to gather, store, consolidate, and analyze a wide variety of operational data types. Data sources may include proprietary agents, logs, events, changes, and metrics [e.g., **Windows Management Instrumentation (WMI)**, simple network management protocol (SNMP), etc.]. They may also include messages from other systems that indicate changes in

[4]SLAs are discussed in Appendix A, which addresses Service-Level Management.

topologies or other relevant information. As an example, **release automation** solutions can provision and configure **virtual servers** as well as the applications and components that run on top of them. These solutions can then propagate topology and configuration changes to analytics systems for incorporation into automated topology maps.

Fig. 12.1 shows a simplified diagram of an on-premises APM solution. Operational data is gathered from agents supplied by the vendor, by third-party agents, and/or by operational protocols such as SNMP. All metrics are centralized, analyzed, and correlated on a central server that hosts the management software delivered by the vendor. All analytics functions, including topology modeling, analytics, and reporting, are done on the central server.

Operations staff are notified via emails, phone calls, pages, and such when the APM solution detects a problem. Operational users can also access real-time system status dashboards, supporting drill downs into underlying infrastructure and reporting software, typically via a web interface connection into the management system.

**Security** and fast processing are two of the key strengths of an on-premises APM solution. Management data sometimes includes sensitive data such as IP addresses, execution logs, and even network **passwords** ("community strings") that could compromise security if they fell into the wrong hands. Hosting the entire system on-premises eliminates this problem, at least to the extent that organizational borders are secured. Companies may also opt to host their own APM solution to minimize

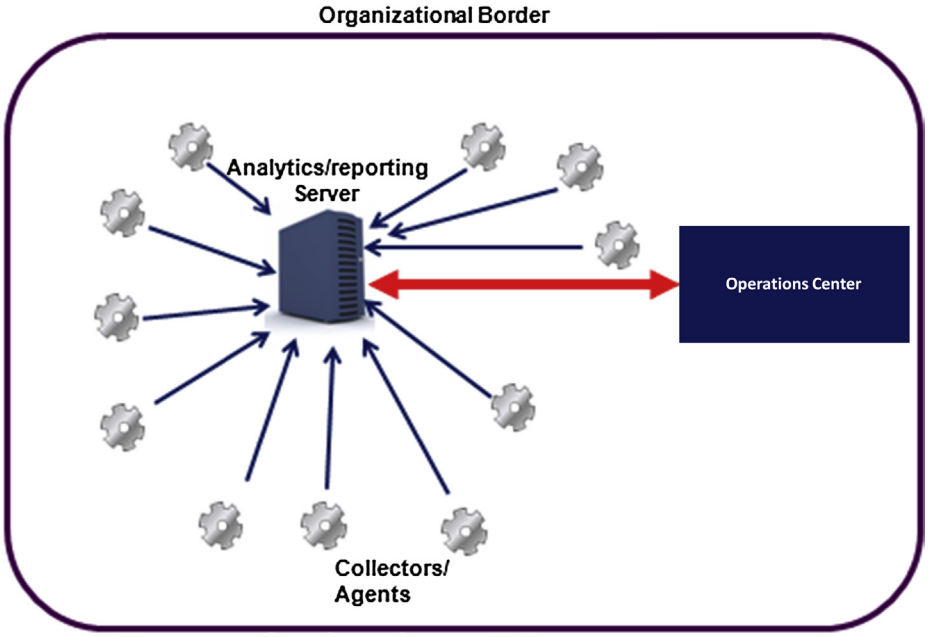


**FIGURE 12.1**

On-premises application performance management.

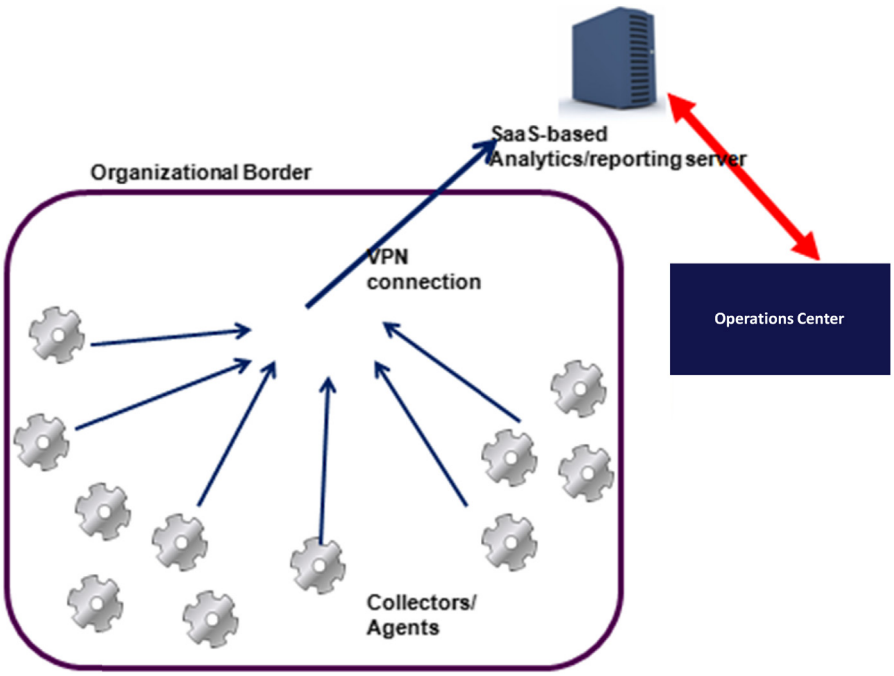


**FIGURE 12.2**

SaaS-based application performance management.

WAN utilization and latency. Due to the fact that the analytics server is within organizational borders, messaging to and from the server likely has minimal latency compared to the longer latency delays that may occur when the analytics are hosted in the cloud.

The primary drawbacks of an on-premise APM product are cost and time. Costs tend to be higher than software as a service (SaaS) options, and maintaining the analytics/reporting server—with updates, paging software, etc.—is a non-trivial affair. In actuality, most companies hosting management solutions on premises have tools teams in place to manage the management tools. In contrast, Fig. 12.2 shows a simplified diagram of a SaaS-based APM solution. The collection structure is virtually identical to that of an on-premises solution. However, instead of being routed to an on-premises management server, metrics are sent via a **virtual private network (VPN)** connection to a **cloud-hosted** system. Operations staff get the same notifications they would receive from an on-premises system and can access dashboards and reporting systems via a browser-based interface.

The choice of an on-premises or SaaS-based APM system is essentially a matter of customer preference. SaaS customers take advantage of world-class APM capabilities without needing to devote expensive personnel resources to system administration tasks and day-to-day maintenance, as well as ongoing updating of new software releases on the analytics server. However, some companies see operational risk in sending metrics to an external platform and prefer to use their own staff to maintain the APM software internally.

## THE APPLICATION PERFORMANCE MANAGEMENT ECOSYSTEM

APM tools can be a unifying factor providing a common language and viewpoint for cross-functional teams with diverse training and skills. Particularly when such tools are capable of gathering data from multiple sources (data center hardware and software, network-focused metrics, application execution data, etc.), they create a centralized application perspective that is virtually impossible to duplicate via manual support processes or by pooling tribal knowledge across teams. Such tools can be critical for delivering application-focused insight to **development, operations, and DevOps** teams across the lifecycle. Preproduction application testing ensures that developers and testers find and fix problems early in the lifecycle—when they are easier and cheaper to remediate. Postdeployment testing detects production problems early, providing an opportunity to fix them before they impact users. Used in such a way, APM solutions provide a unifying bridge, enabling development, operations, and DevOps to communicate across silos more efficiently.

At the same time tools selection can be confusing, largely because of the wide range of technologies and solutions being bundled under the APM umbrella. Different types of APM tools deliver different perspectives on the same application. The APM story is similar to that of the elephant and the blindfolded man—touching a leg, a trunk, and a tail yield very different experiences. At the same time, viewed holistically, it's still an elephant. Tools choices are also complicated by the fact that they aim at an ever-changing target. As new types of technologies are incorporated into the data center, and as software ecosystems evolve, APM solutions must also evolve in a way that addresses the unique management challenges of each.

Fig. 12.3 shows the evolutions of development practices, application architectures, application deployments, and supporting infrastructure over time. As an example of APM evolution, the rise of **containers** and **microservices** shown in the Application Architecture column has driven the growth of

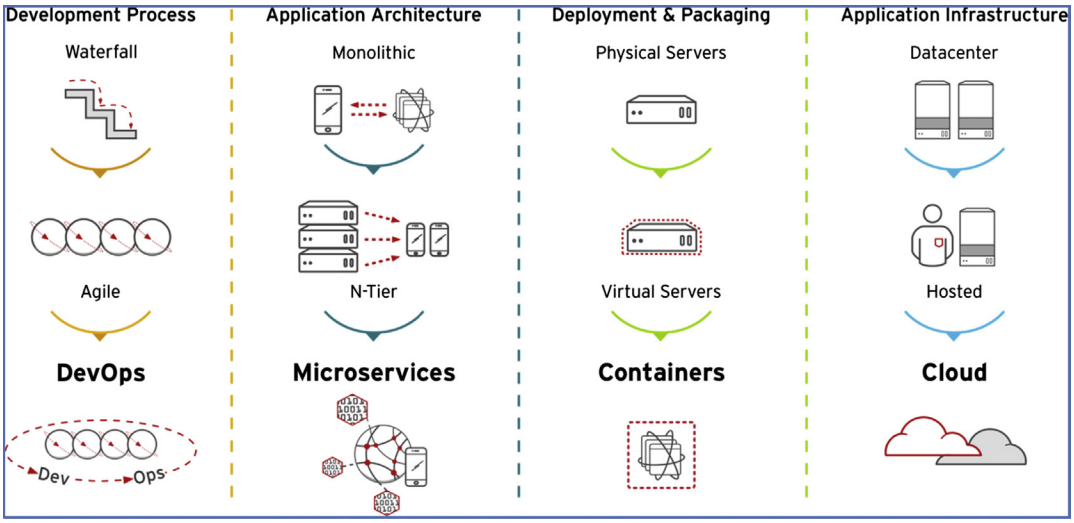


**FIGURE 12.3**

Evolution of development processes, application architectures, deployment platforms, and infrastructure.

*©2016 courtesy of Red Hat, Inc.*

API-connected applications as application components become increasingly granular. Microservices housed in containers are orchestrated into usable business services via APIs.

**Public cloud** services, owned and managed by external entities, are often part of the ecosystem as well. The reality for IT organizations consuming cloud services is that while they have a responsibility to their customers for performance and availability, they have little or no control over how—and at what levels—the service is delivered. To complicate matters further, the evolution of development practices (from waterfall to **agile**) created escalating rates of production change.

Each of these factors can adversely impact production. So as each evolution occurs, APM solutions must also evolve to support the massively connected, dynamically changing application environments that are becoming commonplace in modern clouds and data centers.

APM tools leverage analytics, encapsulated industry/technology expertise, and multiple monitoring protocols to develop visibility and control capabilities supporting applications and their underlying execution environments. They quantify application performance and automate the process of surfacing relationships among applications and infrastructure elements, making it easier to detect and diagnose issues and anomalies contributing to performance or availability problems. Without such a comprehensive, multidimensional approach, activities such as change management and root-cause analysis become increasingly expensive guessing games. It is also the case that many execution issues are never resolved, and therefore continue to recur.

While businesses are proficient at developing sophisticated applications, the statistics suggest that they are not as successful at maintaining and managing them. This is borne out with multiple research findings:

- More than 35% of IT professionals surveyed in 2015 indicated that they lacked the tools needed to support their application environments.
- More than 40% of application outages are reported by users versus management tools.
- Support costs consume 60–80% of the average IT budget. Although this statistic is so commonplace as to have become a cliché, there is a good reason for these high costs. In lieu of adequate management products, people are required to close the management gap—and people are expensive.

At the same time, while the industry coined the term end-to-end to describe the process of managing the application fabric, there is a notable lack of definition regarding what end-to-end actually means. For the purposes of this chapter, the term is used to describe the execution of an application, service, or transaction from start to finish.

Ideally, the process of managing a software service in a business context requires the ability to quantify and track performance and availability, to understand normal performance patterns (including time-based fluctuations), to auto discover dependencies and supporting infrastructure, and to monitor and notify on departures from the norm in real time. APM solutions are designed to address critical questions related to application performance. Examples include:

- **Dependency mapping**:
  - Which infrastructure elements support which applications?
  - What dependencies exist between applications and/or software components?
- **Capacity management**:
  - How are applications impacting the infrastructure that supports them?
  - Is overutilization of underlying resources adversely impacting performance?

- **Change management**:
  - What changed?
  - How will a proposed hardware or software change impact performance of existing applications?
  - Once a change to production is made, how is that change impacting production performance?
- **Proactive analysis**:
  - How can the business impacts of application and infrastructure problems be minimized?
- **Root-cause analysis**:
  - When performance slows or the application becomes unavailable, where is the problem and how can it be fixed?
- **Service-level management**:
  - How does the application normally behave, performancewise?
  - Are there performance variations based on time of day, day of month, and such?
  - How do the levels of performance and availability actually being delivered compare to contracted SLAs?[5]

As **containerization**, virtualization of every type, infrastructure abstraction, network centricity, and massively distributed applications become increasingly common, the answers to questions such as these are no longer straightforward or even forthcoming. To complicate matters, there is a wide variety of APM-related product flavors, most of which focus on specific subsets of the overall APM-related big picture.

## INSTRUMENTATION AND DATA SOURCES SUPPORTING APPLICATION PERFORMANCE MANAGEMENT

### ENTERPRISE MANAGEMENT ASSOCIATES APPLICATION MANAGEMENT SEMANTIC MODEL

To assist IT organizations in tools planning and acquisitions, EMA has developed an Application Management Semantic Model (see Fig. 12.4). This is a comprehensive approach to documenting the instrumentation points supporting delivery of the metrics required for the management of complex applications. The model also documents a tools framework supporting instrumentation of each major dimension of application execution as a basis and foundation for AIA tailored to APM practices.

The model assumes that gathering and analyzing execution information compiled from multiple vantage points (represented by the layers in the model) is ultimately necessary to build a management fabric. This fabric, in turn, will act as a foundation for the multiple functions performed by APM solutions, including analyzing, alerting, reporting, and ultimately fixing application-focused issues. Collectively, the layers of the model quantify the dimensions of visibility that are necessary to develop a holistic, 360 degree view of the execution ecosystem.

### TOOLS, DATA, AND ANALYTICS, AND THE END-TO-END PERSPECTIVE

Table 12.1 is a functional breakdown that documents the function of each layer of the Application Management Semantic Model shown in Fig. 12.4. Each provides a unique execution perspective, gathered from direct instrumentation, protocols operating at that layer, or, in some cases (network, for example) by direct analysis of transactions or data flows occurring at that layer.

[5]See Appendix A.

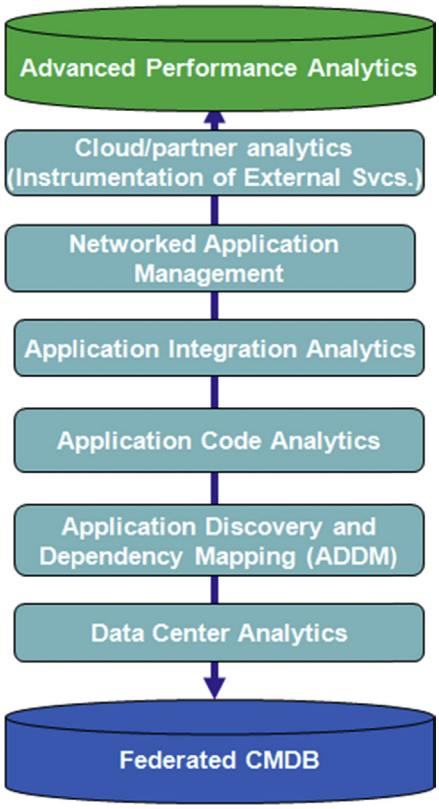


**FIGURE 12.4**

Enterprise Management Associates Application Management Semantic Model.

Needless to say, full automation requires that APM solutions include advanced analytics encompassing very high levels of built in industry expertise. It also requires near real-time monitoring and analysis based on insights into infrastructure performance, application topologies and dependencies, transaction execution, and system configurations. For this reason, in addition to instrumentation points, the model also includes layers supporting advanced performance analytics as well as a federated **configuration management database (CMDB),** also known as a **configuration management system (CMS)**. Respectively, these layers support intelligent data analysis and real time topology modeling.

Today, the layers of the model support:

- Management of the predominantly custom versus packaged applications running in the majority of companies
- The emergence of massively componentized applications such as those deployed utilizing **service-oriented architecture (SOA)** and **web services**
- Widespread use of APIs and other **integration** technologies
- Use of public cloud
- Extensive use of virtualized systems hosted on premise and/or deployed into **infrastructure as a service** clouds

**Table 12.1 Functional Breakdown of Instrumentation Points Supporting 360 degree View of Application Execution**

| Layer | Function |
|---|---|
| Advanced performance analytics | Assimilate/analyze ecosystem data and metrics to deliver operational and business-relevant performance insight |
| Cloud/partner analytics (external services) | Deliver visibility to performance of systems hosted and/or managed by outside entities (i.e., public cloud or partner-delivered services) |
| Network | Gather performance-related metrics via real/synthetic transactions, sniffing, packet analysis, or other flow-centric analysis |
| Application integrations | Track and monitor transactions across middleware and variable transaction paths |
| Application code | Analysis of software code to identify, map, and monitor custom application execution in real time |
| Application discovery and dependency mapping | Discover, identify, and map applications, their interdependencies, and supporting infrastructure (via specialized application discovery tools) |
| Data center analytics | Discover, identify, monitor, and manage infrastructure and foundational technology elements (data/metrics from silo tools) |
| Federated configuration management database | Metadata repository modeling application and technology elements, configurations, dependencies |

However, the intent of the model is to be flexible enough to support the data/metrics sources of the future that will evolve from the introduction of new technologies into the execution fabric. For example, one future addition will likely be a layer supporting the incorporation of big operational data into management systems. Incorporating data from Internet of Things (IoT) will become an important addition for many companies and we are already seeing IoT data integrated with back-end systems of record for a variety of business-focused analysis and reporting purposes—the addition of a data store and processing supporting big operational data will support these new use cases.

Each layer also addresses a specific viewpoint or perspective on the execution environments supporting today's distributed, virtualized, and tiered application architectures. In Fig. 12.5, instrumented data sources are shown in the left column, while supporting analytics and data/metadata repository functions are shown in the green and dark blue layers on the right-hand side of the diagram.

The ultimate goal of full instrumentation of each layer in the model is to provide the building blocks required to automate the process of creating and maintaining a dynamic model of the entire execution fabric that can be updated in real time. Doing so, of course, relies on APM solutions capable of analyzing the data rapidly enough to build and maintain such models. Finally, Fig. 12.6 documents the types of metrics generated and gathered at each instrumentation point. The vertical arrows show data flows to and from the analytics processing engine and, where relevant, the CMDB/CMS (to update topologies, dependencies, and configurations).

The analytics layer consumes data created and/or stored at the other layers to perform the analytics, correlation, and reporting functions. The CMDB/CMS provides an additional data source used by the analytics layer for information about dependencies, topologies, and configurations.

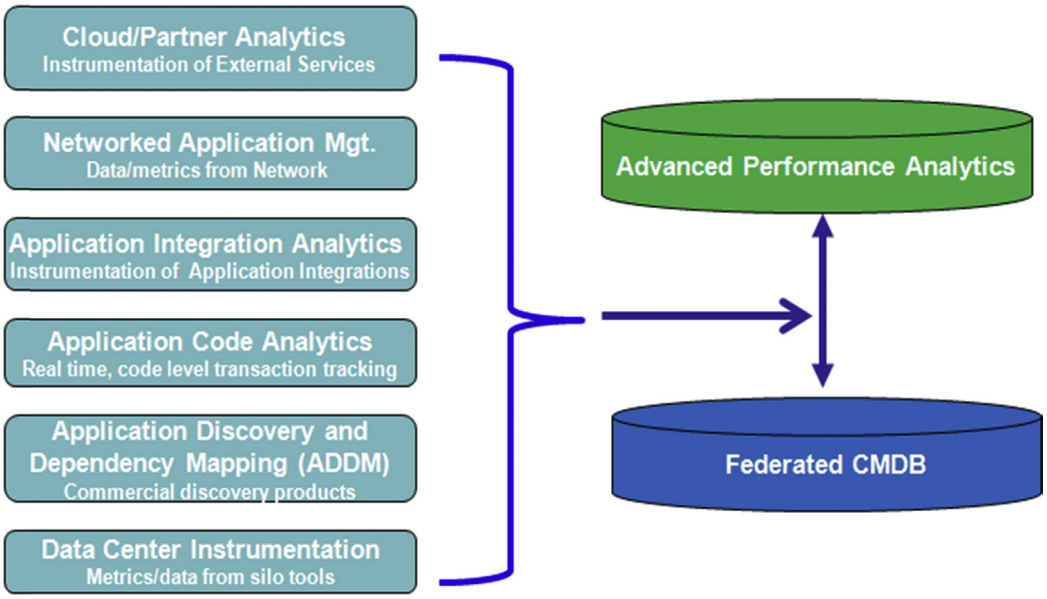


**FIGURE 12.5**

Instrumentation points providing visibility to application execution from multiple perspectives.

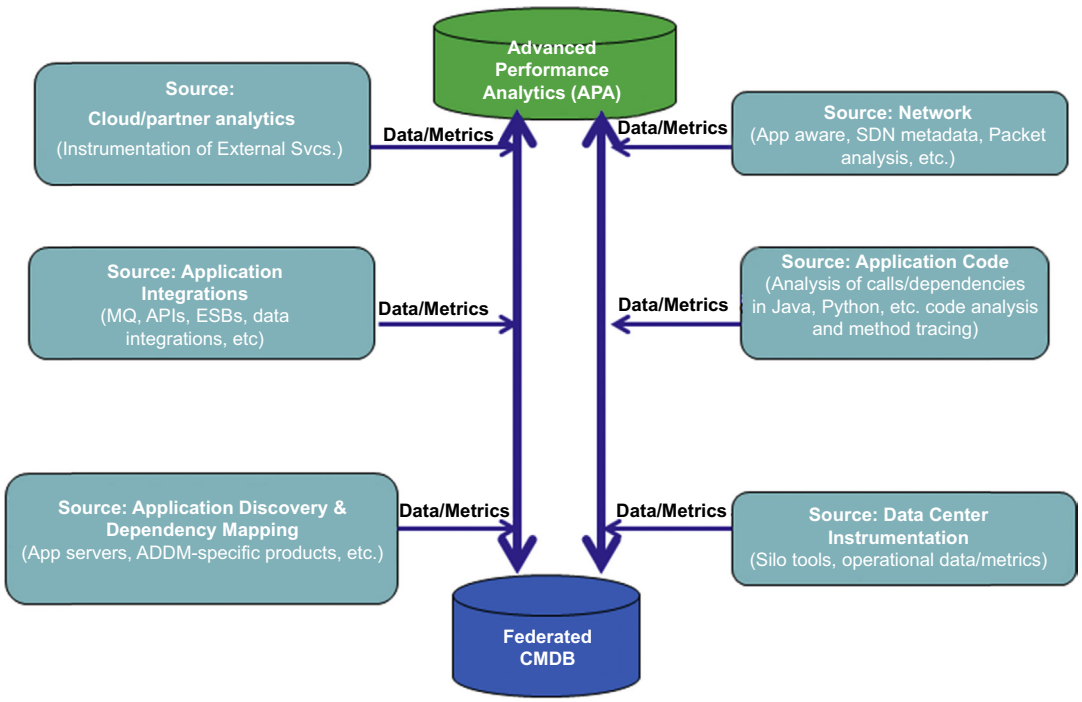


**FIGURE 12.6**

Runtime diagram showing data sources underlying each layer of the model.

## TAXONOMY OF PRODUCT CATEGORIES SUPPORTING APPLICATION PERFORMANCE MANAGEMENT

To some extent, the model also defines a taxonomy of product categories, each of which contributes its own unique perspective to the management fabric. For example, element managers are still important for monitoring and gathering platform-specific health metrics at the data center layer. However, insight into these vertical components is of limited value to the overall APM task without context of the applications they support.

This context is provided by the five remaining vantage points: ADDM, application code analytics, application integration analytics, networked application management, and cloud/partner (external) analytics—and by the advanced performance analytics capabilities at the topmost layer. The following section details the functions and tools supporting each layer in additional detail.

### THE CONFIGURATION MANAGEMENT DATABASE SYSTEM

Although the CMDB (or CMS) system is not explicitly responsible for gathering information about application execution, it does provide the context and data repository necessary to manage it. The federated CMDB/CMS system consists of data stores containing data and metadata about the end-to-end application fabric. In conjunction with modeling and management products, the CMDB/CMS ideally provides the key that describes how all the moving parts comprising the application interact with one another. EMA sees the CMDB/CMS as a prerequisite for end-to-end application management and a foundation for an efficient support process. EMA has written extensively about the CMDB/CMS in recent years and consulted with numerous companies worldwide on CMDB/CMS readiness and adoption.

EMA also sees metadata repositories, such as SOA registry/repositories and other repositories of execution data, becoming increasingly integrated with CMDB/CMS systems. Metadata provides critical information about application execution that helps build a big-picture view of loosely coupled application systems. Overall, EMA analysts believe the CMDB/CMS (composed of distributed repositories containing increasingly auto-discovered and self-maintained topologies and configurations) could become the Rosetta Stone, which, combined with analytics, would be positioned to translate isolated infrastructure elements into a holistic, application-focused context. Without such insight, managing applications is essentially a process of trial and error, heavily dependent on manual processes and organizational expertise.

### DATA CENTER INSTRUMENTATION

The bottom layer of the model shown in Fig. 12.4, the data center layer, is one level of instrumentation that most organizations already have in place. Since the term data center has different meanings across the industry, for the purposes of the semantic model, the data center includes all the in-house hosted infrastructure elements necessary to deliver an application. From the tools perspective, tools at this level are the point products that manage servers, the network, and other foundational elements. Virtually all of these silo-focused tools include correlation capabilities designed to narrow down the potential sources of a given performance/availability issue for troubleshooting purposes.

There is nothing mysterious about these products as many have been in the marketplace for years. However, they are still foundational to managing the data center, and they also provide valuable data that is a fundamental element of application support.

This foundational layer provides insight into the health of the application's execution platform. It enables application issues to be correlated with infrastructure issues, when they exist, so they can be managed in context with one another. This is a prerequisite for efficient troubleshooting and root-cause analysis activities, among other functions.

Network management products are one example of a product that functions at this layer. The need to manage network devices drove the development of SNMP and its introduction to the industry in the 1980s. Not only is SNMP still widely used today, but multiple additional protocols and standards have been introduced over time as well. This market is very mature, with a host of quality products purpose-built to gather operational metrics from routers, servers, databases, storage, and almost every other platform in the data center. Even **uninterruptible power supplies (UPS)** have **SNMP MIBs**, allowing them to be monitored by SNMP-focused management stations.

## APPLICATION DISCOVERY AND DEPENDENCY MAPPING

Application discovery and dependency mapping (ADDM) products are commercial solutions purpose-built to discover applications running in a production ecosystem and document them in context with supporting infrastructure and dependencies. Typically, such products utilize application signatures, or fingerprints, to discover applications, their downstream dependencies, and their supporting infrastructure, then assemble the resulting models into topology maps.

Commercial discovery and mapping products are shipped with prebuilt discovery signatures for an array of packaged applications. Since they lack signatures for custom applications, virtually all include tools designed to simplify the task of manually modeling custom applications so the automation understands how they are built.

ADDM vendors and products encompass varying levels of application awareness and functionality, along with varying levels of support for signatures and custom (usually homegrown) application support tools. However, most require a significant amount of manual scripting and grouping even after automated discoveries are complete, and these tasks are often performed as part of initial setup by vendor services organizations.

This product family is foundational to managing distributed applications for several reasons. These products document the links between applications and supporting infrastructure in an automated fashion. Further, they keep these links current, even within today's constantly changing IT systems. This information can then be used for multiple support tasks, including troubleshooting, root-cause analysis, and change management. In terms of change management, for example, they provide insight into upstream and downstream dependencies that may potentially be impacted by a given change.

However while ADDM technology continues to improve, it is also true that leading-edge APM solutions are starting to support a far greater level of topology modeling than they have in the past. Leading-edge APM solutions can automatically map component-based applications by watching intra-component interactions (see the auto-generated topology shown in Fig. 12.7, generated by IBM APM software). In many cases, these solutions mitigate or eliminate the need for ADDM solutions, since this function becomes part of the day-to-day, real-time topology modeling function.

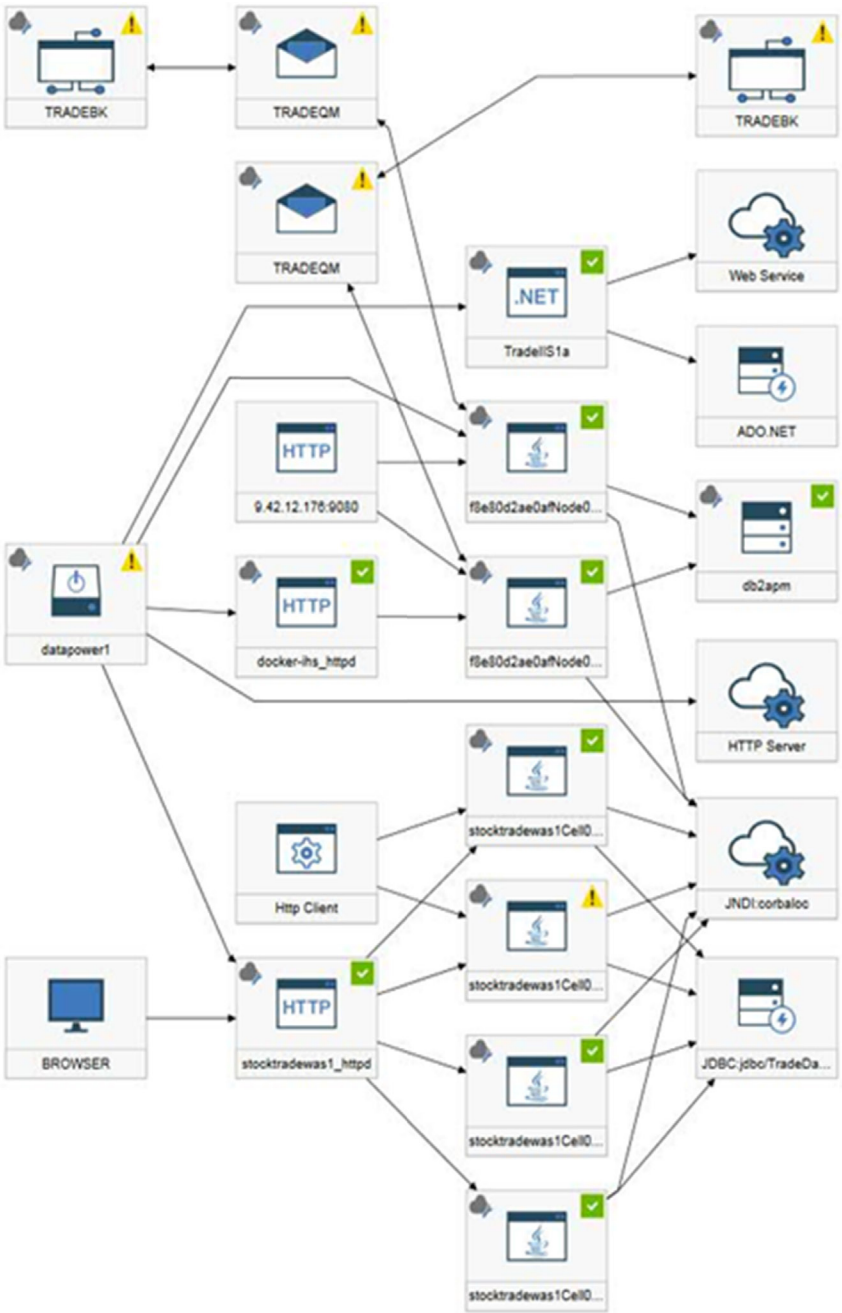


**FIGURE 12.7**

Automated topology map generated by IBM application performance management software.

*©2016 courtesy of IBM.*

## APPLICATION CODE ANALYTICS

Code analytics products deliver deep insight into execution systems from the perspective of instrumented application servers. The value proposition of these solutions is that they are able to pinpoint execution-related problems down to a given piece of application code.

In some leading-edge APM solutions, code analysis also provides a basis for ADDM functionality by developing signatures or fingerprints for automatically discovering and mapping custom applications. Discovering custom applications is a problem for traditional application discovery products because no fingerprint exists to detect and identify the application. These products fill that gap by identifying executable code and, in doing so, generating a fingerprint by which custom applications can be tracked and identified.

There are a wide variety of products in this space, some of which have only recently come to market. The newest analyze application code predeployment and then use that analysis to discover and manage applications in production. Other products in this space monitor Java, node.js, and/or .NET executables during execution to identify availability and/or performance problems. They can break down execution into steps and quantify the time spent at each step. All of these capabilities facilitate root cause analysis of code-related issues for both custom and packaged applications and, in doing so, also helps identify opportunities for code redesign.

## APPLICATION INTEGRATION ANALYTICS

This family of products tracks applications and transactions through the layers of integration that proliferate in **client/server**, web, and SOA-based applications. Many traditional application management products lack visibility to integration layers and technology touch points. Without this visibility, such products are unable to track a transaction across variable execution paths, multiple technologies, and complex middleware stacks. Integration technologies become, in essence, black boxes to the monitoring/management systems. This can be a source of significant risk, particularly if they support critical business applications.

Instrumentation supporting this layer covers the touch points between infrastructure elements, transactions, applications, components, and services. In addition to handoffs between infrastructure elements such as application server to database server, it can include interconnections among J2EE platforms, ESBs, registries and repositories, and related platforms that together route, execute, and manage underlying connections.

The most advanced products in this category can also track the variable routes that a transaction can take. Tracing these routes becomes particularly challenging when elements such as **XML** offloading, load balancing, WAN optimization, and similar technologies become part of the application execution chain. Products in this category are also essential for mapping real applications to real transactions and transaction paths, and for providing the insight into dependencies that is necessary for root-cause analysis. For example, **message queuing (MQ)** has been a standard method supporting data interchange for years, and many companies are still running hundreds of instances of MQ, allowing a wide variety of application types to interact. Still, many APM solutions as yet have minimal support for tracking MQ interactions as part of application execution. Other examples of integration technologies that could be addressed by this class of products include:

- ESB integrations: ESBs play a major role in SOAs by supporting rollouts of massively componentized software services. They enable businesses to link multiple discrete services together by normalizing across a wide variety of underlying data interchange methodologies.

- API integrations: Today, APIs are playing a central role in integrating companies with providers, suppliers, and customers. The so-called API Economy will likely continue to accelerate as APIs increasingly support the **container-based microservices** and **IoT** deployments of the future. API connections are now part of the execution chain as well, yet many companies have no way to track transactions (and consequently performance) across and through API connections.

## NETWORKED APPLICATION MANAGEMENT

Networked application management solutions are playing an increasing role in providing user experience metrics quantifying Layer 8[6] performance for mobile, componentized, web, and cloud-based applications. These products differ from the standard network monitoring solutions that are part of the data center instrumentation layer by virtue of their visibility into the network traffic that carries application-related information. This visibility delivers higher-level functions such as user experience and performance/availability of external services.

Multiple solutions in this category have extremely varied capabilities and are rapidly evolving beyond their network management–centric roots. As applications are increasingly bound to the network, products that watch transaction flows and application exchanges become treasure troves of data relevant to managing applications.

As a product family, this new generation of networked application management solutions operates on transaction flows to gather application-relevant data related to network performance and transaction performance and, in some cases, the content included in the data flowing across the network. Some products in this category, for example, can perform such tasks as extracting business-relevant information from web traffic. This functionality would enable customers to track incoming sales transactions to maintain a running record of the day's online sales.

**Packet analysis** is another function performed by network facing solutions, and one that supports both performance management and business management capabilities. By utilizing "from-to" packet information, products in this category have the potential to automate the process of identifying and mapping transactions, regardless of the path the transaction takes as it traverses the execution environment. Automated **transaction tracing**, a technique utilized by a subset of products in the APM category, is one example of state of the art in this space.

Transactional reconstruction is done based on tagging transactions, then tracking them through the execution process. Solutions in this space utilize a combination of instrumentation and analytics to track transactions across diverse technology elements in real time, essentially automating the process of end-to-end transaction performance/availability management. Transaction tracing is useful for automating monitoring and management of complex transactions and mission-critical transactions such as those characteristic of equities trading and other financial systems.

Since these applications are network-centric, the challenge for vendors is to combine network-focused functionality with application-focused insights. The next few years will be pivotal for this market because its success relies on the ability of network vendors to make the paradigm shift to applications and to

[6]While the OSI model of networking has seven layers, Layer 8 refers to a hypothetical layer above Layer 7 (application) layer, often called the user layer. In reality, there is no Layer 8; however this is a sort of technical shorthand describing the applications that developers write and with which humans interact. One source for additional information on the OSI model is searchnetworking.techtarget.com/definition/OSI.

partner with application management vendors with similar vision. This will likely be a case where the most flexible and visionary vendors will be the most successful in the marketplace.

Theoretically, if all network connection points could be instrumented, this category of products could provide a way to automate 99% of the work involved in discovering, modeling, mapping, and monitoring virtually every transaction traversing a company's network. Observation of network traffic at wire-speed provides a basis for analysis of virtually all traffic interchanges for relevant content.

Today, a handful of network-focused solutions utilizes such analysis to support real-time application optimization, real-time reporting, troubleshooting, and **transaction management**. As analytics become increasingly sophisticated and as physical storage and processing power substantially escalate to support real time analytics, this product segment will likely be a primary data/metrics source for the fully automated data centers of the future. This type of real-time analysis will be essential for tracking and monitoring cloud bursts, dynamic deployments supported by **software-defined networking**, and similar automated functions.

### ADVANCED IT ANALYTICS LAYER

AIA is EMA's term for analyzing operational big data for a variety of use cases. These include optimizing service performance, minimizing security issues, managing change, and optimizing capacity across internal IT and the extended enterprise (partners, service providers, suppliers, etc.). This data is also ultimately utilized for governing IT more effectively in support of the business it serves. This layer provides the "secret sauce" that distinguishes one vendor from another in virtually all the other layers. While it's not difficult to collect metrics and data from SNMP, WMI, logs, and similar technical sources, the difficulty lies in how that data is analyzed, once it is available. Analytics govern the way each vendor's solution correlates, combines, understands, and processes available data. And since analytics reflect the vendor's primary intellectual property supporting APM and UEM functions, many of the algorithms and heuristic capabilities utilized in these types of solutions are patented to prevent their use by competing vendors.

## USER EXPERIENCE MANAGEMENT

### INTRODUCTION

The expanding diversity of execution ecosystems has elevated **UEM** in prominence. As these ecosystems increasingly incorporate cloud, hybrid, virtual technologies, **SaaS**, and so on into transaction execution, UEM becomes one of the few constant and meaningful barometers for successful application delivery. EMA research bears this out. One study showed that UEM accelerated benefits from **cloud computing** across the board; benefits included accelerated service deployments, reductions in operational costs, and reductions in management complexity. It also found that service providers were nearly twice as likely to increase revenue from their cloud-related services by using UEM-related technologies for service monitoring.

There are diverse opinions regarding where UEM solutions fit on the enterprise management spectrum. Often the distinctions between APM and UEM definitions are very vendor-specific. Some view UEM as part of APM, while others view it as an adjunct, add-on function. The right answer is that UEM is both; indeed, multiple vendors have incorporated UEM functionality into APM solutions, providing customers with a consolidated view of application performance as experienced by the user in context to underlying hardware and software infrastructure elements.

From the perspective of the EMA Application Management Semantic Model shown in Fig. 12.4, UEM solutions are classified as part of the network layer because they focus on the performance of a single transaction or application as it traverses the network. Instead of simply doing silo monitoring focused on a single infrastructure element, UEM solutions track performance of a given transaction or user action across silos at the execution level. Depending on the type of product, UEM solutions may or may not link end-to-end execution with underlying execution elements. As a result, while some do provide information relevant to root-cause analysis, none provide the comprehensive visibility and analytics supporting root-cause analysis that APM solutions are built to deliver.

Because some UEM solutions—including those leveraging synthetic transactions—require no instrumentation on the platforms they monitor, they are the preferred monitoring platform for services delivered via the public cloud. More than 40% of today's IT organizations indicate that cloud application deployments are causing them to reevaluate management solution portfolios, and most of those without UEM solutions in place are considering purchases of these solutions. UEM solutions range from relatively inexpensive, compared to full-fledged APM solutions, to very expensive. Similarly, while some solutions require only minimal skills to manage and use, others require highly skilled coders and/or network engineers. Regardless of the specifics, these solutions (as a group) provide the number one way IT organizations plan to monitor and manage end-to-end transactions that access the cloud.

UEM solutions are specifically designed to monitor availability, as well as performance as experienced by the end user. They encompass a range of technology types. For example, active (synthetic transaction) and passive (**real user monitoring**) monitoring technologies, as well as client-side instrumentation and web server/browser instrumentation, all deliver visibility to specific aspects of overall performance and therefore provide insight on the user experience.

While each of these perspectives yields a partial view of application performance, most companies prefer to monitor mission-critical applications from multiple perspectives. The closer a company can get to end-to-end monitoring via automation, the less time is spent on triage and root-cause analysis. Suite solutions, encompassing both APM and UEM capabilities, do all of this by correlating and analyzing metrics from multiple locations into a single view of application performance. Often, these solutions pull in insights from infrastructure components, such as load balancers and middleware, to complete the end-to-end picture.

## TYPES OF USER EXPERIENCE MANAGEMENT SOLUTIONS

Table 12.2 summarizes the key UEM product types and the primary functions of each type. The following section covers each product type in additional detail.

**Active or synthetic transaction monitoring (synthetic transactions)**: Synthetic transactions, sometimes called robot transactions, are script-based tests that run at specified intervals against specified execution environments. They monitor transaction performance by automating the process of running tests against the application being monitored. These types of products are widely available, are capable of monitoring performance across the Internet, and provide an excellent entry point into UEM.

Synthetic transactions test performance and availability regardless of whether users are on a system or not. However, since they have little or no visibility of the underlying technology supporting the transaction, they are primarily used for high-level performance/availability analysis versus root-cause analysis. They yield two types of information. First, they test whether an actual transaction can be completed, in other words, whether the system is available or not. They also quantify the time spent for

**Table 12.2 User Experience Management Products and Features**

| UEM Product/Capability Type | Function |
|---|---|
| Active or synthetic transaction monitoring | Synthetic transactions, sometimes called robot transactions, are script-based tests that run against an application at regular intervals |
| Passive or observed transaction monitoring | Also known as Real User Monitoring (or RUM), these tools leverage appliances placed at strategic points across the data center and/or end-user workstation to watch network or data flows, typically HTTP/HTTPS and similar |
| Browser injection | Monitor interactions between the web server and the browser by injecting code during execution |
| Client-side monitoring | Monitor performance and availability from the perspective of the user workstation |

the transaction to complete, in other words, performance of the application. Unlike RUM, they do not require that anyone is actually working on the system. In other words, they monitor actual execution time, not the actual user experience. For this reason, they are often used for unattended testing. For example, many IT organizations run synthetic transactions around the clock regardless of whether or not users are on the system. This means that if something goes down overnight, the problem can be detected and fixed before users arrive and sign on in the morning.

Synthetic transactions start with scripting. Scripts are developed using programming languages or by point and click graphical interfaces, depending on the product. The scripts simulate a sequence of actions that a user might perform in the process of performing a task, such as entering a purchase order. The script is then run at given intervals against the system where purchase orders are entered—the company's enterprise resource planning system, for example.

These types of solutions are available as on-premises or cloud-based services. For example, cloud-based synthetic monitoring solutions have been available in the marketplace for quite some time from vendors such as Keynote and Gomez (now part of Dynatrace).

- Limitations: While these solutions advise IT support teams about basic performance and availability, the limitation is that most cannot pinpoint the source of a problem and why it is occurring. In addition, scripts often require maintenance if the underlying application changes, and some basic solutions even require script updates when a given web page changes. In addition, performance or availability issues vary significantly based on the origin of script execution. For example, if scripts are run within a company's **firewall**, they give little indication of what the user experience might look like from a remote office or on a mobile device.
- Use cases: These types of transactions are ideal for specific use cases. Since they are capable of monitoring performance whether or not users are on the system, they are often deployed as an early warning system to notify IT teams of execution issues occurring during off hours. Unlike real-user monitoring (RUM) solutions, they can test web-based applications running over the **WAN** or the Internet. Since they can also be run from or to multiple external locations, they are also useful for comparing performance across locations in corporate environments with multiple worldwide offices. IT executives often want to compare performance across geographies to ensure that contracted service levels are being maintained across the globe. Finally, synthetic transactions can be used to monitor internal or customer users, since no instrumentation is required to be installed on the endpoint.

**Passive or observed transaction monitoring (RUM from the network):** Products in this category are at the high end costwise and typically require network-focused skills to deploy and operate. RUM solutions monitor end-to-end transaction performance (for web and/or nonweb applications) from the perspective of the network, often collecting metrics from network taps[7] or span ports.[8] These types of solutions have visibility of the actual user actions, underlying network traffic, and interactions occurring during transaction execution; users must be on the system for these types of solutions to yield value.

RUM solutions see all network traffic traversing the monitored port. Virtually all are smart enough to analyze and report on specific types of Layer 7 traffic, including well-known protocols such as **FTP** and **HTTP**. However, an increasing number are also able to recognize and track Layer 8 traffic generated by commercial software packages developed by well-known business application vendors. They recognize these solutions using a combination of built-in analytics and the execution fingerprints of the commercial application. Some such solutions go deeper, utilizing packet analysis to actually break open the packets traversing the network to see the data contained in the packet, or even the payload. In other words, these solutions would be able to watch transactions to intelligently track sales information incoming from a website in real time, and so on.

RUM solutions give very deep insight into actual information flows, such as messaging and call sequences traversing the network. Since these message flows are the glue that binds transactions together, these network-oriented products can yield very useful application-related information. They can contribute valuable insight into the automated creation of topology models, for example, since they can see which nodes are talking to others and how they interact.

Because they have visibility to actual network traffic, these solutions can be very effective for general troubleshooting purposes. For example, one company experiencing application slowdowns every day at noon used a product of this type to trace the source to the fact that employees were turning on Internet radio while they ate lunch at their desks, leaving business-critical applications bandwidth-constrained in terms of network availability.

- Limitations: Network monitoring delivers visibility to network traffic and device-to-device interactions within a company's internal boundaries. Once the transaction exits organizational borders to the public Internet, for example, transaction visibility is lost. Some products in this class can track egress and reentry via transaction tagging; however, this capability varies by vendor. In contrast, server monitoring agents can passively observe transactions and interactions on any manageable endpoint. Cloud-based virtual servers, for example, can be included in the end-to-end monitoring ecosystem, as long as the cloud customer has the ability to provision an agent on the cloud-based system. In other words, once a transaction exits organizational boundaries, agents installed on the application server can take over end-to-end monitoring, if this feature is built into the monitoring solution provided by the vendor.

[7]A network tap is an external monitoring device that mirrors the traffic that passes between two network nodes. A tap (test access point) is a hardware device inserted at a specific point in the network to monitor data (from searchnetworking.techtarget.com/definition/Network-tap, accessed 2/15/2016).

[8]Port mirroring, also known as SPAN (switched port analyzer), is a method of monitoring network traffic. With port mirroring enabled, the switch sends a copy of all network packets seen on one port (or an entire VLAN) to another port, where the packet can be analyzed (from www.miarec.com/faq/what-is-port-mirroring, accessed 2/15/2016).

- Use cases: Since this class of solutions gathers data from network spans and taps, they are most useful for monitoring applications executing within the customer's own network. However, they can monitor performance of external clouds accessed by applications by watching egress (when a transaction leaves the data center) and ingress (when the transaction returns from the cloud). Comparing these two gives a quantification of time spent off premises. This is a useful metric when troubleshooting performance problems for **hybrid cloud** application configurations in which execution spans on premises and cloud. Finally, network-focused monitoring is best suited to monitoring internal or customer users, since, in most cases, these solutions have no visibility of external or public networks.

**Passive or observed transaction monitoring (RUM from the application server)**: This set of solutions monitors network and user interactions from the perspective of the application server. Code-focused agent capabilities watch applications as they traverse Java or .NET platforms, while network interactions can be tracked by intercepting information from the network interfaces on each server. Although most such solutions are instrumented via agents installed on application servers, in some cases other types of servers and devices can also be instrumented depending on the types of agents provided by the RUM solution vendor.

- Limitations: While agent-based management systems are in wide use and such systems deliver very deep insight into applications, transactions, dependencies, and code, many companies find that the process of installing and maintaining hundreds or thousands of agents requires a significant amount of staff time. The resource overhead involved in managing the management system is also higher than that of network-focused RUM, for example, since installed agents talk to a centralized analytics and reporting server. So, in addition to installing agents, customers are also required to install and maintain the server. This has been alleviated by some vendors of agent-based solutions that moved analytics and reporting to the cloud. In this scenario, the link between agent and server is achieved via a virtual private network link, which adds security by shielding sensitive IP-address and topology data from unwanted intruders.
- Use cases: The network-focused RUM solutions described above monitor network and device traffic within a given company's operational boundaries. Once the transaction exits organizational borders to the public Internet, for example, transaction visibility is lost. In contrast, server monitoring agents can passively observe transactions and interactions on any manageable endpoint. Cloud-based virtual servers, for example, can be included in the end-to-end monitoring ecosystem, as long as the cloud customer has the ability to provision an agent on the cloud-based system.

**Browser injection**: This type of end user monitoring monitors interactions between the web server and the browser and, often, between the user and the application. In this scenario, the web server injects code into the browser, enabling monitoring of web traffic and performance as experienced by the user.

- Limitations: Products with injection capabilities are rarely available as standalone solutions. Instead, they add value to APM or UEM solutions by providing a depth of insight into actions on the endpoint that are not readily available via any other monitoring technique.
- Use cases: Since only the web server is instrumented, browser injection is one of the best ways to monitor web interactions as experienced by customers and other external users. Depending on the analytics capabilities delivered by the vendor as part of the analytics/reporting server, solutions of

this nature can yield detailed insights into web interactions impacting customer satisfaction or revenue. Browser injection provides accurate insights into user interactions with a web application, which can assist developers and marketing personnel in identifying user responses to the various steps of a sales transaction. Browser injection can, for example, track user actions to determine the points at which transactions are abandoned within a shopping cart application. This type of information can be used to make price adjustments and/or application redesigns, either of which may lead customers to complete future transactions.

**Client-side monitoring**: Client-side monitoring provides the best way to deliver comprehensive visibility to what has been called "the last two feet" of a transaction. This class of solutions delivers visibility to metrics and events on the desktop or mobile device, which other solutions lack. Like other products in the UEM/APM categories, these solutions also rely on a central server to aggregate, analyze, and correlate metrics and deliver real-time alerting and after-the-fact reports on the findings.

Client-side UEM solutions use actual client-side instrumentation to gather information about all things related to the internal customer and his or her workstation. They rely on agents installed on the endpoint to provide deep monitoring of the actions and experiences of internal users. This solution category is particularly useful for diagnosing application problems with sources originating on the endpoint. For example, old, underpowered PCs, software conflicts, and poorly configured devices can all impact perceived performance and are best diagnosed at the endpoint.

Due to their deep visibility to users and endpoints, as well as the fact that they are typically installed on every user's workstation, they provide unparalleled support for user management as well. They may, for example, reveal the need for user training in the use of an application, or for a device or network upgrade.

- Limitations: While these solutions provide deep visibility to the endpoint and some visibility back into the data center, they lack full visibility into end-to-end execution as the transaction traverses the data center. This means that while they are very good at quantifying the user experience, as solo solutions they lack the comprehensive breadth required to diagnose the root cause of the majority of problems not originating on the endpoint. In an effort to close this gap, endpoint monitoring vendors often partner with traditional APM vendors. Integrations between traditional APM and endpoint monitoring can provide the best of both worlds, yielding detailed correlations of endpoint behavior with application performance that cannot be equaled without endpoint instrumentation.
- Use cases: These solutions are ideal for supporting applications running on any physical or virtual Windows-based desktop. Depending on the vendor, they may also support mobile devices and native **mobile applications**. Granular insight into Citrix XenApp, **VDI**, Rich Client, Java and **Rich Internet Applications (RIAs)**, and asynchronous capabilities are available, depending on the specific product.

## SUMMARY

Analytics, UEM, and APM are taking center stage as companies find that cloud, mobile, and container-supported software deployments are complicating the task of application support. With growing on-premises complexity and the march to the public cloud, APM and UEM solutions have become essential tools for development, DevOps, and operations personnel. These groups need faster, better, and cheaper ways to test, monitor, and manage application performance.

While APM and UEM solutions can help to ensure a high-quality user experience, the capabilities of a given tool or solution depend largely on the instrumentation and proprietary analytics delivered by the vendor. Products capable of observing and learning from their environments can minimize hands-on application support requirements, maximize staff and resource utilization, reduce business risk, and optimize application quality.

From a capability perspective, the combination of APM and UEM functions provides flexible, comprehensive coverage for transactions spanning cloud, mobile, and web. Analytics operating on combined metrics from real user interactions, infrastructure monitoring, and synthetic monitoring provide performance visibility and root-cause analysis regardless of whether users are on the system or not. This combination of capabilities enables an APM solution to preemptively monitor transactions, detect topology changes on an ongoing basis, and analyze real user metrics for UEM optimization. These are differentiating capabilities that are available primarily in high-end APM solutions.

In recent years, those APM solutions capable of supporting collectors and metrics from third-party solutions have become particularly attractive to medium-sized to enterprise-sized companies. Interoperability—the ability to share metrics with and consume metrics from third-party systems—maximizes the value proposition of existing management tools investments.

## KEY TAKEAWAYS

- Traditionally, applications were supported with a combination of silo monitoring and tribal knowledge—knowledge built from the experience of the hands-on specialists supporting a company's enterprise applications. However, IT ecosystems have now become so massive and complex that it is no longer possible to manage them with silo knowledge alone.
- Analytics capable of transforming data and metrics into a human readable application or transaction perspective are what separates actual application monitoring from relatively simple silo monitoring.
- Analytics add the context necessary to understand the role of each moving part in end-to-end execution, a viewpoint that is absolutely critical for rapid—and eventually automated—problem determination and resolution. Without this context, these solutions would simply be operational data stores versus true tools capable of intelligent insight into application ecosystems.
- APM and UEM are not one-size-fits-all terms. There is a wide variety of permutations and tools options in each group.
- APM tools can be a unifying factor providing a common language and viewpoint for cross-functional teams with diverse training and skills. Particularly when such tools are capable of gathering data from multiple sources (data center hardware and software, network-focused metrics, application execution data, etc.), they create a centralized application perspective that is virtually impossible to duplicate manually or via tribal knowledge across teams.
- The expanding diversity of execution ecosystems has elevated UEM in prominence. As these ecosystems increasingly incorporate cloud, hybrid, virtual technologies, SaaS, and such into transaction execution, UEM becomes one of the few constant and meaningful barometers for successful application delivery.

**Examples of vendors with products in this space**:

Apica
AppDynamics
AppFirst
ASG
Appnomic
Aternity
BlazeMeter
BMC
Catchpoint
CA Technologies
Dyn (Acquired by Oracle)
Dynatrace
Dell Software
EMC
ExtraHop
HP
IBM
Idera
ManageEngine
Nastel
New Relic
NetScout
Netuitive
Push Technology
Riverbed
SmartBear
SOASTA
Splunk
Stackify
Sumo Logic