APPDYNAMICS

7 Steps to Maximizing the Value of APM

AppDynamics is now part of Cisco.

CISCO

Introduction

The world of application development and operations is complex.

Application performance management (APM) is an essential solution alleviating some of that complexity, and AppDynamics APM has become the tool of choice for many of the world's leading enterprises.

This guide will help you understand why APM is critical to the success of your business – and how you can maximize AppDynamics APM to successfully improve business performance and customer experience.

Why APM?

Today, businesses are using APM to connect app performance and the customer experience to business outcomes.

APM tracks the behavior of components, such as physical hardware, virtual machines where applications reside, JVM executing the application environment, web containers, associated databases, caches, external web services, and the code running the application itself. APM also provides detailed insight into the end-user experience, helping you understand the quality of service you're providing to your customers.

By monitoring applications and tracking customer behavior, AppDynamics APM can alert stakeholders of any issues related to key Business Transactions — like a broken checkout link that could impact revenue or a slow login page that's hurting customer experience. With AppDynamics, businesses are alerted to application performance issues in real time, allowing business owners to fix the problem before it hurts the company's bottom line.

By using the best practices laid out here, you'll become an expert at making the optimal use of AppDynamics APM. You'll prevent damage to your organization's reputation and delight customers with outstanding performance.

Getting Started With AppDynamics APM

The following 7 steps will help you establish best practices in APM strategy to reach, maintain, and suggest new service levels for the applications you manage.

1. Create an APM Strategy Document
2. Create an Internal APM Wiki
3. Separate Signal From Noise
4. Define Key Performance Indicators (KPIs)
5. Establish Health Rules
6. Create Dashboard Visualizations
7. Automate Everything

These steps will help empower you and your team to deliver on the mandate that IT drives the business forward. The quality, performance, and business value of the applications that you create and operate are essential to the success of your enterprise.

There's never been a more exciting time for IT, and a lot of valuable information awaits, so let's get started.

1 Create an APM Strategy Document

Application performance affects many people — inside and outside your organization. Each person brings their unique set of priorities and, inevitably, some of those priorities will come into conflict. Organizations often use APM to serve those conflicting priorities, which can cause confusion and ineffective action. That's why an APM strategy is essential.

Though every organization is different, your APM strategy should consist of the following sections:

VISION AND GOALS

It's important to know that vision and goals cannot be top-down only. While the vision and goals of the company typically originate with the founders, the first principle of Agile development is: "Active user involvement is imperative." You should be able to articulate the company vision in a way that unites the team. The goals for the application should follow naturally from the vision statement.

PRIORITIES AND PLAN

This is the point at which the lead on the strategy document should shift to the implementation team. Developers and operations engineers need to come to an agreement on what actions should be prioritized to achieve the stated goals. These should include:

- Core applications to be monitored: end user, internal and external services, processes
- Infrastructure components to be monitored: servers, DB, network
- Business and technical dashboard visualizations
- Service-level agreements (SLAs) and alerting

The next step could be as simple as a Gantt chart. Identify the dependencies in meeting your priorities. The last step is assigning tasks in the action plan. A plan built on input from stakeholders eliminates all the time wasted on conflicting priorities and misaligned goals.

ROLES AND RESPONSIBILITIES

AppDynamics APM is a powerful tool, so make sure your team knows how to use it. Decide who is responsible for which features, and what their access level should be.

EDUCATION AND ENABLEMENT

Make sure you set aside adequate time for education and enablement so that the people responsible for interpreting data are proficient with AppDynamics, and not just one little corner of the most-used features.

INTEGRATION

AppDynamics APM must have access to everything that could affect the performance of your application. Build time into your plan for integration across the network, and try to predict areas where integration could run into roadblocks.

SOFTWARE DEVELOPMENT LIFE CYCLE (SDLC)

APM provides benefits across the SDLC, but the focus will differ. Dev and Test will be looking primarily at improving code quality, release velocity, and scalability, whereas Production will be more focused on delivering a great end-user experience (EUE), so it's important to plan your implementation accordingly to leverage the greatest benefit.

With the strategy document created, the next step is to establish an internal APM wiki to enable stakeholder communication and collaboration.

Create an Internal APM Wiki

You should create dedicated wiki content for your APM deployment, which will function as the central communications hub for all stakeholders and subject matter experts who have critical inputs to the project.

It's a home base where your team can share best practices as they develop and evolve essential knowledge for the team.

CONTENTS OF YOUR WIKI

Here are some suggestions that have proven useful for development teams:

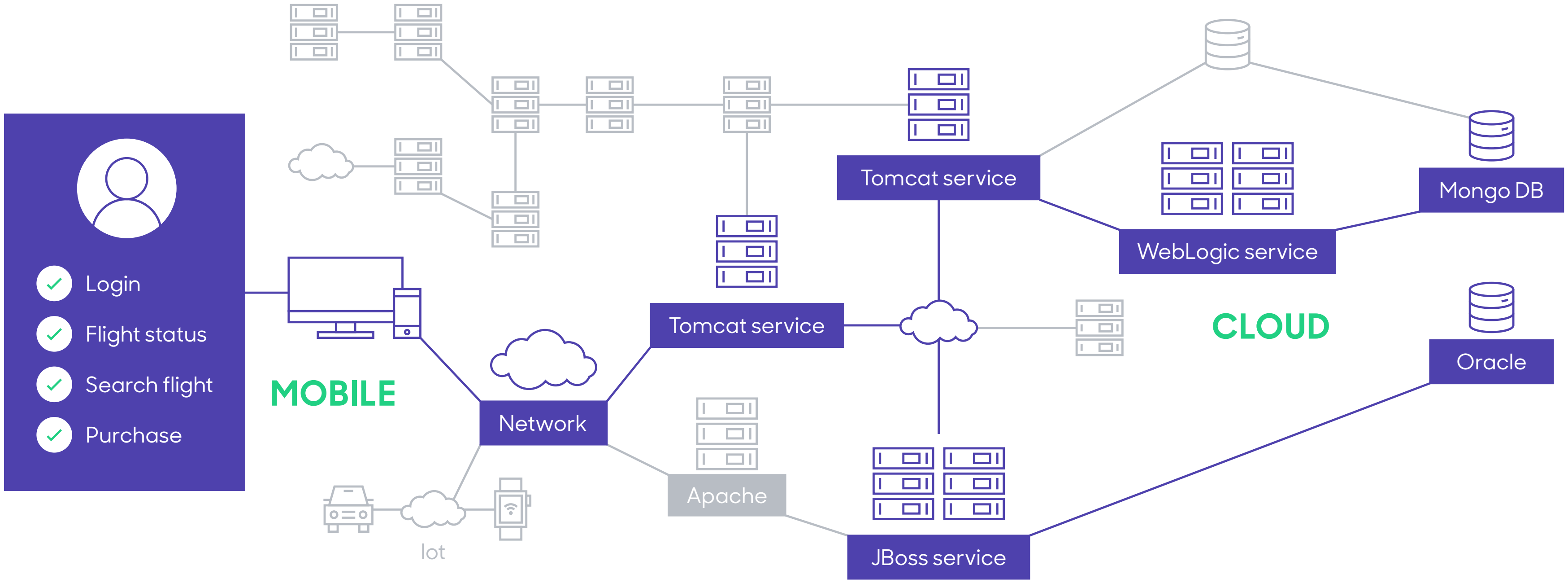
- APM strategy document
- Communications dashboard
- How team members should keep in touch — consider using a messaging solution like Slack or HipChat
- Proposed configurations
- Distributed application architecture examples
- Success stories for solving common development issues
- Links to knowledge base for your APM tools
- APM tutorial videos
- FAQs for onboarding new team members on APM concepts
- Playbooks for critical errors, such as triage procedures for high response times

3 Separate Signal From Noise

AppDynamics APM will track and visualize a vast number of data points to help resolve issues and tune performance. However, it's important to decipher which data is "signal" and which is "noise." Think of signal as your critical application traffic and noise as other background activity happening in your IT estate.

DEFINING BUSINESS TRANSACTIONS

In APM terms, Business Transactions are the top-level metric. Each defines a logical set of user actions related to application functionality as demonstrated in the following graphic, i.e., login, flight status, search flight, purchase. Each user action can then be directly associated with application traffic flow across each monitored tier.



Business Transactions carry the highest level of risk to your business if they don't perform as expected, so it's very important to ensure that they are all correctly identified and defined as part of your APM implementation.

AppDynamics APM automatically identifies the most common Business Transactions and assigns names based on URL, entry point, HTTP operations such as POST, or even functional names based on GET parameters.

Many organizations allow AppDynamics APM to automatically discover a wide range of Business Transactions. Then they simply sort by calls per minute or average response time to find the transactions with the greatest traffic or highest response time. Those are commonly the ones most in need of monitoring, and for many customers they may be limited to a few dozen, even for sophisticated applications.

RECOMMENDED STEPS IN DEVELOPING BUSINESS TRANSACTIONS:

- Create a list of critical Business Transactions that need to be monitored.
- Validate your list throughout your organization.
- Review the automatic Business Transaction identification to ensure:
    - All Business Transactions on your list are covered.
    - The names allocated are appropriate — you can amend as required.
- Collect all other Business Transactions in a "catch-all-others" or "All Other Traffic" category.

NEXT STEPS:

- Set monitoring and alerting for each Business Transaction based on key performance indicators (KPIs) (covered in Step 4).
- Set health rules for each Business Transaction (covered in Step 5).

Define Key Performance Indicators (KPIs)

KPIs are data metrics baselining the performance of hardware and software.

As you deploy AppDynamics APM, you must find the most intelligent way to choose from among hundreds of thousands of data points to be measured — or possibly millions of data points at larger enterprises.

COMMON KPIs

The three most important measures for maintaining excellence throughout the Business Transactions are:

- Load: measuring total number of calls per minute
- Response time: with average (or percentile) responses measured in milliseconds
- Error rate: including both the total number and the error types per minute

THE RIGHT NUMBER OF KPIs

Keep narrowing your area of concentration down from hundreds of thousands of data points to the smallest number of factors that impact customers directly. You may be tempted to start with a large amount of KPIs, because an automated system, like AppDynamics, can alert you to any number of metrics that exceed KPI thresholds or agreed on SLAs. That may not be the best move.

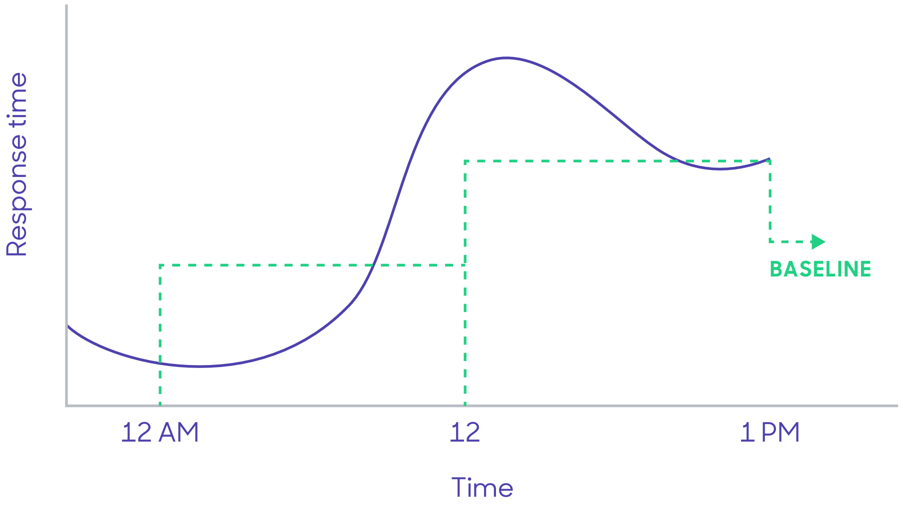
Begin by establishing a response time KPI for each core Business Transaction. AppDynamics will automatically learn what's normal for your core Business Transactions in terms of response time, so it's a matter of curating this value and adjusting alerting accordingly. Once this important step is complete, you can look to adjust or add further KPIs as required.

Put a framework in place to start isolating and incorporating KPI expansion. Iterative improvement and carefully measured growth will deliver far more reliable results.

5 Establish Health Rules

One of the key concepts in APM is that your application is healthy when it performs at or above normal behavior as identified in the baseline measurements.

Health rules are alerts that notify you whenever conditions move outside the established KPI baseline or other health indicators point to a malfunction. For example, one of the most common health rules is, "Create a critical event when average response time increases above three standard deviations of the baseline."



RULE TIERS

Three areas to concentrate on health rules are at the tier level, Business Transaction level, and the infrastructure level.

- Tier health rules: Looks at a specific tier for spikes or drops in response time, error rates, miscellaneous exception, calls, and call drops. It takes time to set up for a single tier, but that process can be copied for smaller organizations or automated for larger ones.
- Business Transaction health rules: Compares data against the baseline (or a static value) for the same anomalies mentioned above as data moves through each Business Transaction.
- Infrastructure health rules: This keeps an eye on the underlying hardware/VMs with CPU usage rates, memory usage rates, and, for example, time spent by JVM to complete Garbage Collection (GC).
- Node health rules: Other commonly measured node examples are the thread pool utilization rate, queue sizes, wait time to acquire a connection, and thread contention.

RULE TRIGGERS

Whenever an event exceeds the conditions of your health rules, you can configure AppDynamics APM to trigger a policy, which initiates an action that you have defined, such as sending out an email alert, sending an alert to a service management deployment, or running a script. The health statuses are typically normal, warning, and critical.

The health rule wait time setting allows you to decide how long you want to wait before the action is triggered again. Each health rule is evaluated every minute, but you decide how often you want to get notified if a health rule is violated.

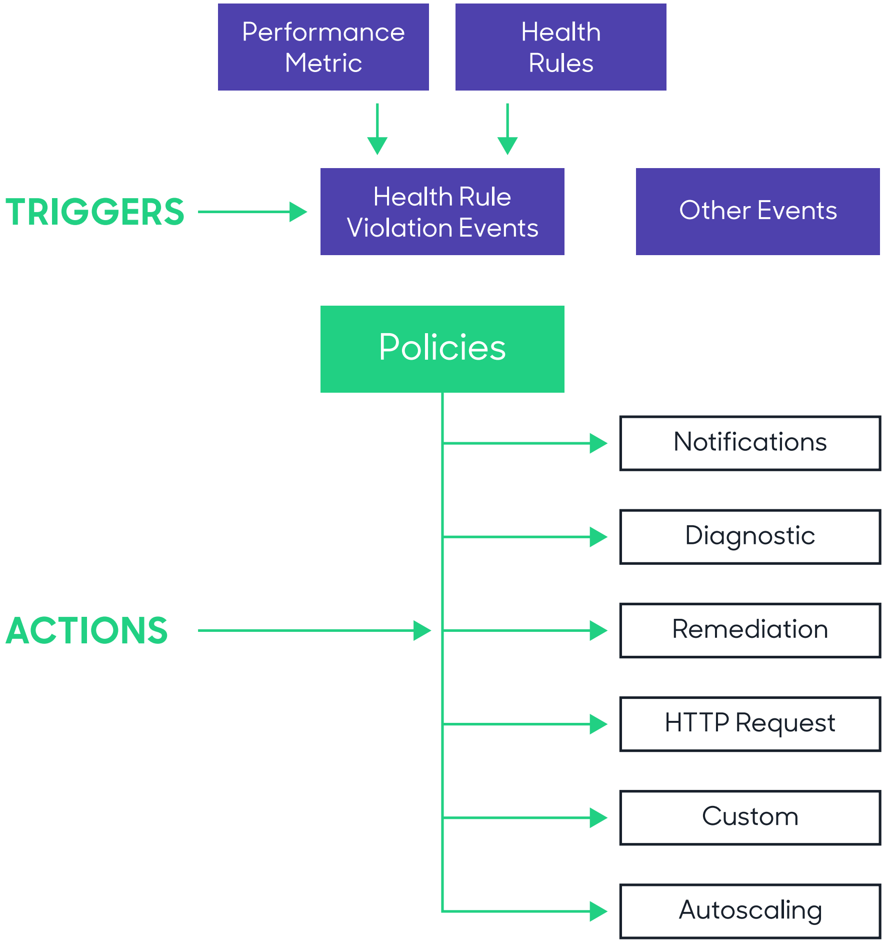
If the health rule violation still exceeds your parameters, the violation remains open in the same status. You can then set up a different action to be triggered based on the "Continues Critical" or "Continues Warning" status.

RULE DATA

The other important setting of a health rule is to configure how much data is considered for the health rule validation. Let's say the CPU of a node spikes for one minute to 95%. Is this a valid violation of the rule? Maybe you want only to receive an alert if the spike continues for five minutes. In that case, set the time for the health rule to evaluate data to five minutes. This setting is a "sensitivity trigger," as a longer time reduces the sensitivity of the health rule.

There are many ways to go from here, but once you have the essential health rules in place, you'll want to visualize them, which we'll cover in Step 6.

ALERT AND RESPONSE OVERVIEW



Create Dashboard Visualizations

Appropriate dashboard visualizations are one of the most important aspects of a successful APM implementation. Without providing relevant data to relevant stakeholders, you are severely limiting the value of your APM investment.

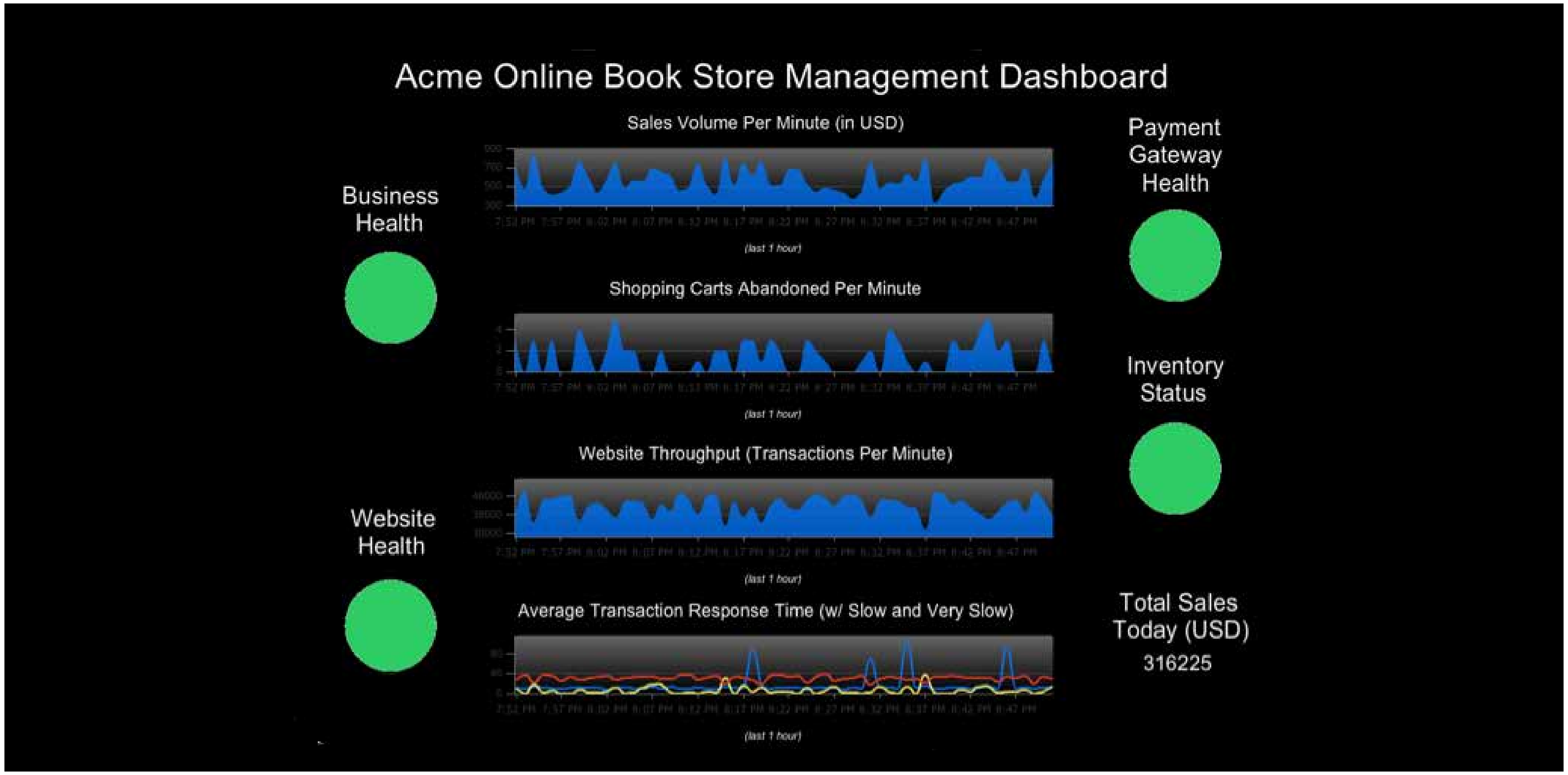
So, who are the stakeholders that should be leveraging dashboards — and what metrics do they care about?

C-LEVEL MANAGEMENT

Executives may want to easily view and understand the impact of application performance on revenue.

Metrics they care about may include:

- Conversion rate (completed sales) vs. Business Transaction response time
- Sales volume
- Abandoned shopping carts

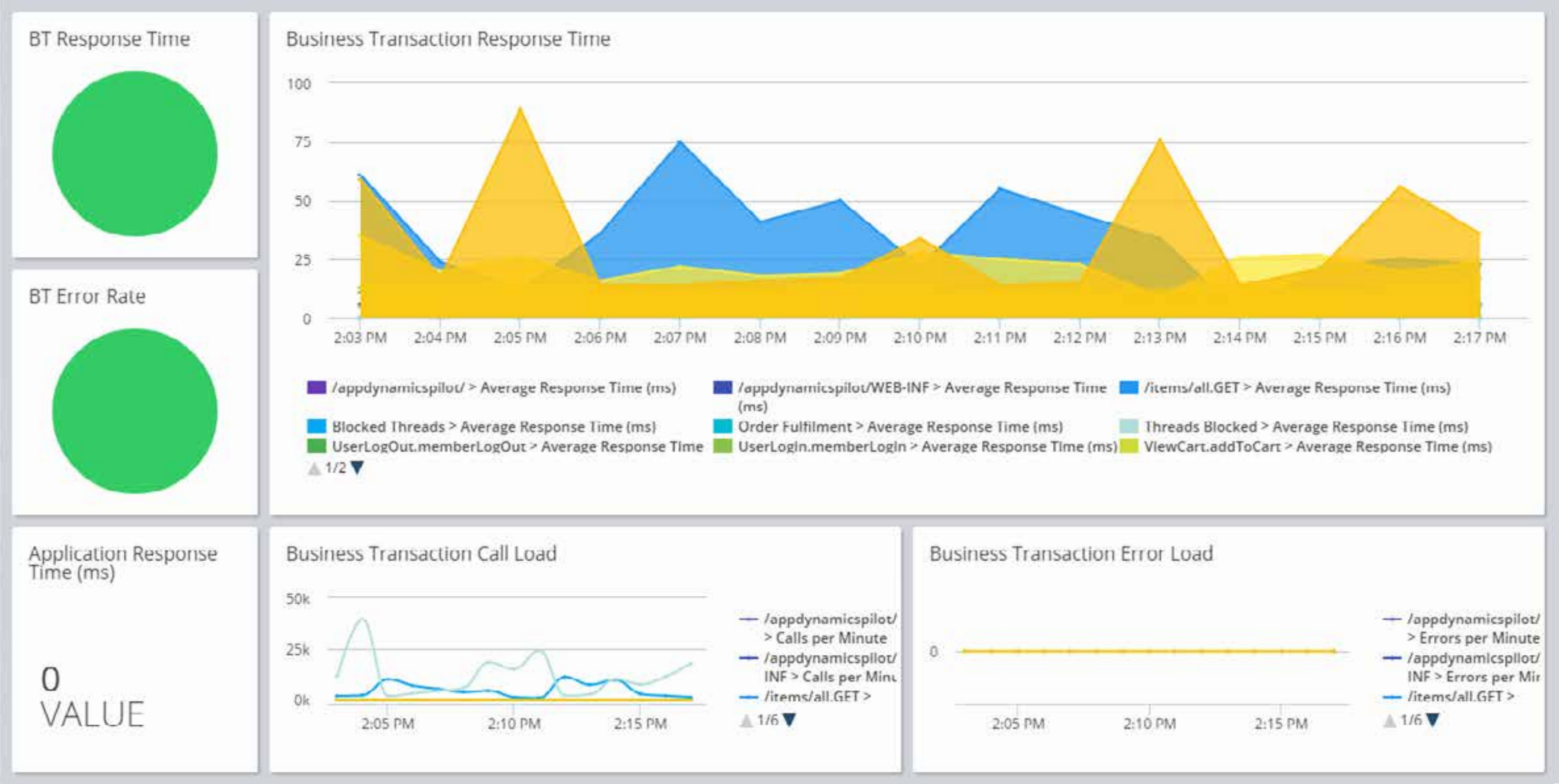

Acme Online Book Store Management Dashboard
Sales Volume Per Minute (in USD)
(last 1 hour)
Shopping Carts Abandoned Per Minute
(last 1 hour)
Website Throughput (Transactions Per Minute)
(last 1 hour)
Average Transaction Response Time (w/ Slow and Very Slow)
(last 1 hour)
Business Health
Website Health
Payment Gateway Health
Inventory Status
Total Sales Today (USD)
316225

DEV TEAMS

Development teams can use dashboards to measure code quality and track performance defects. They want to see how performance is trending against SLAs of overnight builds.

Metrics they care about may include:

- Business Transaction response times
- Business Transaction error rates
- Business Transaction call loads
- Business Transaction error loads

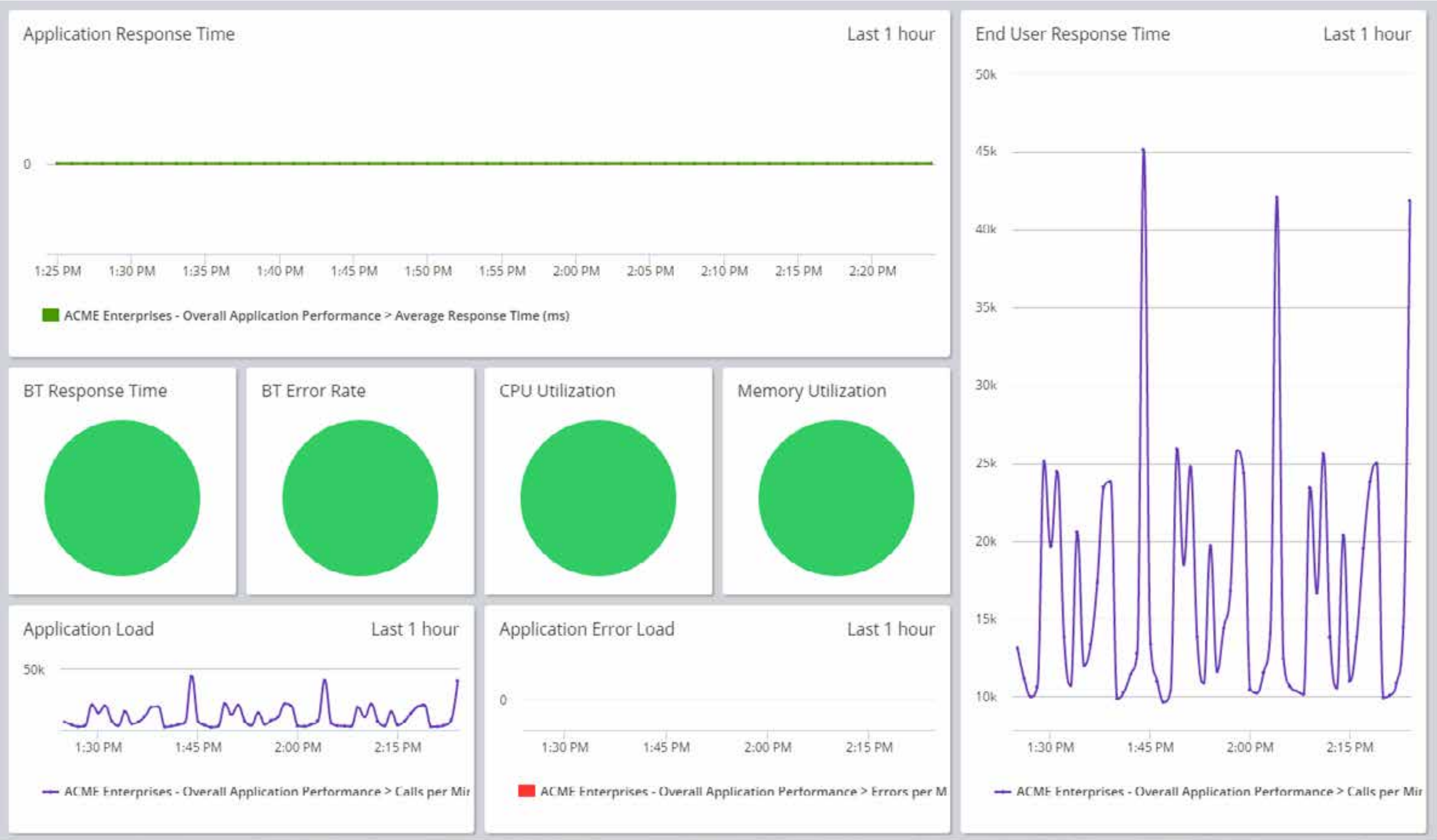

BT Response Time
BT Error Rate
Application Response Time (ms)
0
VALUE
Business Transaction Response Time
100
75
50
25
0
2:03 PM
2:04 PM
2:05 PM
2:06 PM
2:07 PM
2:08 PM
2:09 PM
2:10 PM
2:11 PM
2:12 PM
2:13 PM
2:14 PM
2:15 PM
2:16 PM
2:17 PM
/appdynamicspilot/ > Average Response Time (ms)
/appdynamicspilot/WEB-INF > Average Response Time (ms)
/items/all.GET > Average Response Time (ms)
Blocked Threads > Average Response Time (ms)
Order Fulfilment > Average Response Time (ms)
Threads Blocked > Average Response Time (ms)
UserLogOut.memberLogOut > Average Response Time
UserLogin.memberLogin > Average Response Time (ms)
ViewCart.addToCart > Average Response Time (ms)
▲ 1/2 ▼
Business Transaction Call Load
50k
25k
0k
2:05 PM
2:10 PM
2:15 PM
/appdynamicspilot/ > Calls per Minute
/appdynamicspilot/ INF > Calls per Minu
/items/all.GET >
▲ 1/6 ▼
Business Transaction Error Load
0
2:05 PM
2:10 PM
2:15 PM
/appdynamicspilot/ > Errors per Minute
/appdynamicspilot/ INF > Errors per Mir
/items/all.GET >
▲ 1/6 ▼

OPS TEAMS

Operations teams can leverage dashboards to monitor application health and track Business Transaction performance 24/7.

Metrics they care about may include:

- Application response time
- Application load
- End-user response time



DASHBOARD CREATION

The end goal is a range of dashboard visualizations that immediately informs even non-technical stakeholders when something isn't right. Something as simple as green-yellow-red color assignments based on tolerance levels moves you in the right direction.

In creating dashboards, you want to get as much input as possible from those most likely to use the dashboard. You'll typically aggregate data from various sources and combine it all into a visualization of live and historical data.

In terms of dashboard best practices, consider the following:

- Plan rather than create ad-hoc dashboards.
- Decide on the different audiences and the essential information they require.
- Don't try and cram too much information into a single dashboard view, as this only makes it harder to understand what the dashboard is intended to present.
- Keep the number of dashboards to a minimum.
- It makes sense to design your dashboard content so that from the top level, which is typically Business Transaction focused, you can easily drill down into more technical views to troubleshoot problems.
- Rather than create something entirely new, is there already a similar out-of-the-box dashboard that's good enough?
- Tier/node dashboards: While custom dashboards typically aggregate information from various sources, a tier/node dashboard is specific and can easily be cloned across tiers and nodes. You may use them for technical information for engineering teams, while the custom dashboard is more management-oriented.

7 Automate Everything

There's a popular DevOps framework developed by John Willis, Damon Edwards, and Jez Humble known as CALMS. It directs DevOps teams to devote their time to the following core values:

- Culture
- Automation
- Lean
- Measurement
- Sharing

While defining the core values of DevOps culture is a topic worthy of its own eBook, the principles of automation and sharing deserve your attention right now.

AppDynamics APM will help you set up workflows to automate commonly repeated tasks, such as scripts and configuration changes. You'll be able to set up rules that instantly trigger cloud-based scaling and creating or deleting instances in the cloud in response to metrics.

AUTOMATION BEST PRACTICES

Best practices suggest there are four areas where automation can be especially valuable:

- Agent deployment
- Monitoring for on-premises controllers
- Configuration deployment
- Integration with event systems

The temptation is to set aside automation as a "nice-to-have" feature that takes up too much time, but the opposite is true. Many teams find themselves falling behind at moments like these, and having automated agent deployment is one area where you can save a great deal of time right away.

AUTOMATING AGENT DEPLOYMENT

Automating the agent deployment and upgrade will dramatically reduce time and mistakes during installation. Even small shops should try to automate their agent deployment as soon as possible. Instrumenting an application adds an agent to the application's runtime process. It's common to deploy 10 to 20 agents or more during a major release or enhancement. Unified agents do not take the place of automated agents.

AUTOMATING MONITORING FOR ON-PREMISES CONTROLLERS

The bigger your controller is, the more important it is to monitor this critical asset. Straight out of the box, AppDynamics' controller comes with a self-monitoring capability. You'll be able to add more configuration functions as you become proficient, but you need to start with the basics. Automated monitoring is the most direct way to proactively troubleshoot the controller. If you need to contact the AppDynamics support team for any reason, data collected in automated controller monitoring will help them point you in the right direction.

AUTOMATING CONFIGURATION DEPLOYMENT

This is a much more advanced technique that mature APM users have used after they have built up absolute confidence in automating components. With this, you'll be able to automate the entire configuration deployment. What this means is that a developer can set up a Business Transaction in the Dev environment, then automatically migrate that Business Transaction to a Test environment. After it's been tested out, the Business Transaction can automatically be moved directly into production.

Consider a continuous deployment scenario, where a developer writes a new functionality. Together with the code, they deploy the Business Transaction with data collectors and a health rule. This makes it easy for Ops to monitor the new functionality.

INTEGRATION WITH EVENT SYSTEMS

Finally, automation allows us to integrate with internal or external event systems. Examples could include:

- Raising tickets in defect tracking solutions like JIRA as a result of detecting errors or performance regression in overnight Dev builds
- Sending alert notifications to infrastructure monitoring frameworks as a result of AppDynamics health alerts (discussed in Step 5)
- Initiating provisioning of additional web server resources in AWS as a result of detecting increasing traffic volumes

DO NOT AUTOMATE WHAT YOU DO NOT UNDERSTAND

In the drive for efficiency, some teams end up automating processes before the effects across the system are fully explored. These bad processes become automated bad processes, hiding the error and generating more work in the end.

Take a step back and discuss with team members how any new automation, script, or module is likely to force changes to other current processes. While automation is necessary to be able to respond or remediate issues in time to help customers, you cannot allow automation to slow down adaptation to rapid changes. Do not allow automation to intimidate team members from running new scripts in response to an unusual event.

Automation should free up your time to improve collaboration across the enterprise. The 21st century is characterized by toppling functional silos and making traditional IT specialty knowledge available to less technical colleagues.

AppDynamics Key Benefits

APPLICATION PERFORMANCE MANAGEMENT

Get real-time, end-to-end management made for the most complex and distributed applications. AppDynamics APM helps you focus on what matters with features like application mapping, dynamic baselining, and code-level diagnostics.

INFRASTRUCTURE VISIBILITY

Make sure your application performance is being fully supported by your infrastructure with server and database performance visibility. Infrastructure visibility easily lets you drill down from Business Transactions to troubleshoot server, database, and network issues before they affect customers.

END USER MONITORING

Ensure each tap, swipe, click, or interaction is flawless and quickly solve issues by automatically capturing errors, crashes, network requests, page load details, and other metrics. Synthetic monitoring agents establish performance baselines, manage third-party SLAs, and benchmark the competition.

BUSINESS MONITORING

Our business monitoring solution provides clear, actionable correlations between the quality of your customers' experiences with your applications and the eventual impact on your bottom line — helping you spot new opportunities to improve outcomes and drive customer loyalty.

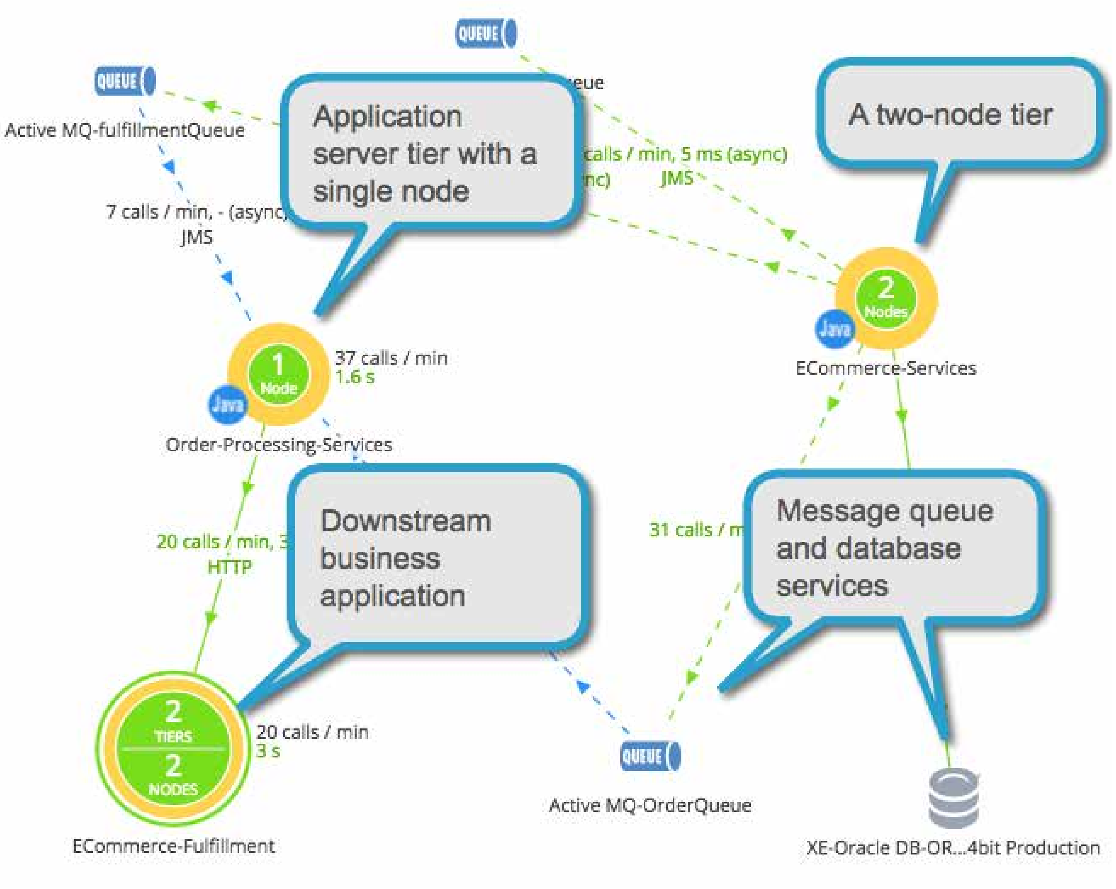
Essential Features

In order to leverage AppDynamics APM to its fullest extent, you'll want to become familiar with these three essential features that will help you maximize deployments:

FLOW MAPS

Flow maps are visualizations generated when app agents installed on your application runtime (e.g., JVM or CLR) report back to AppDynamics. Those reports are stitched together to create a map of the logical and technical flow of data and traffic throughout your distributed environment.

AppDynamics agents automatically inject instrumentation in application bytecode at runtime so they can continuously discover and map Business Transactions, services, backends, etc., in your application architecture.

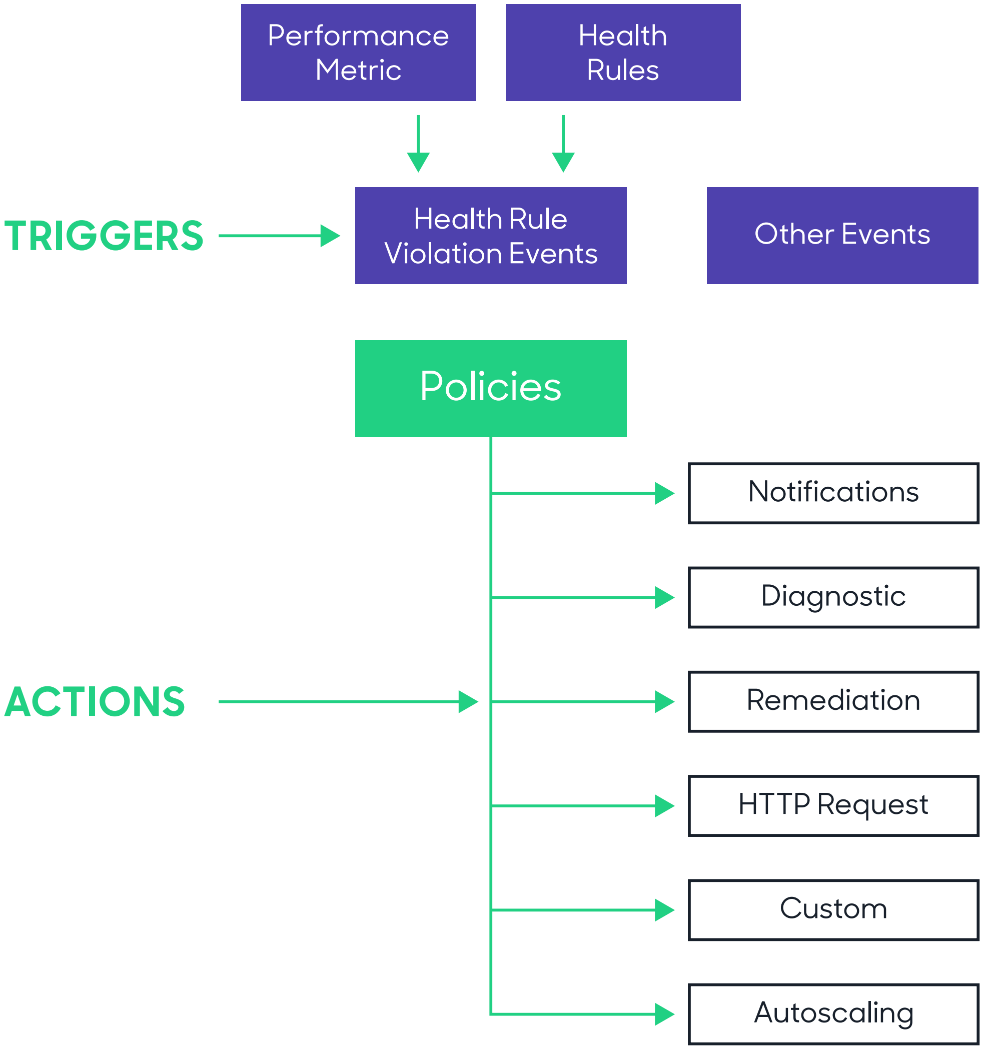


Application server tier with a single node
A two-node tier
Downstream business application
Message queue and database services
QUEUE
Active MQ-fulfillmentQueue
7 calls / min, - (async)
JMS
calls / min, 5 ms (async)
JMS
1 Node
Java
Order-Processing-Services
37 calls / min
1.6 s
2 Nodes
ECommerce-Services
20 calls / min, 3
HTTP
31 calls / m
2 TIERS 2 NODES
ECommerce-Fulfillment
20 calls / min
3 s
Active MQ-OrderQueue
XE-Oracle DB-OR...4bit Production

ALERTS / BASELINES

Alerts and baselines are then automatically created, establishing the historical app performance "normal." AppDynamics compares actual versus historical normal for a baseline reading. You can then set thresholds for alerting when performance wavers from the baseline. AppDynamics' baselining is also dynamic, meaning that it is ongoing, and can change and update itself over time, as "normal" changes.

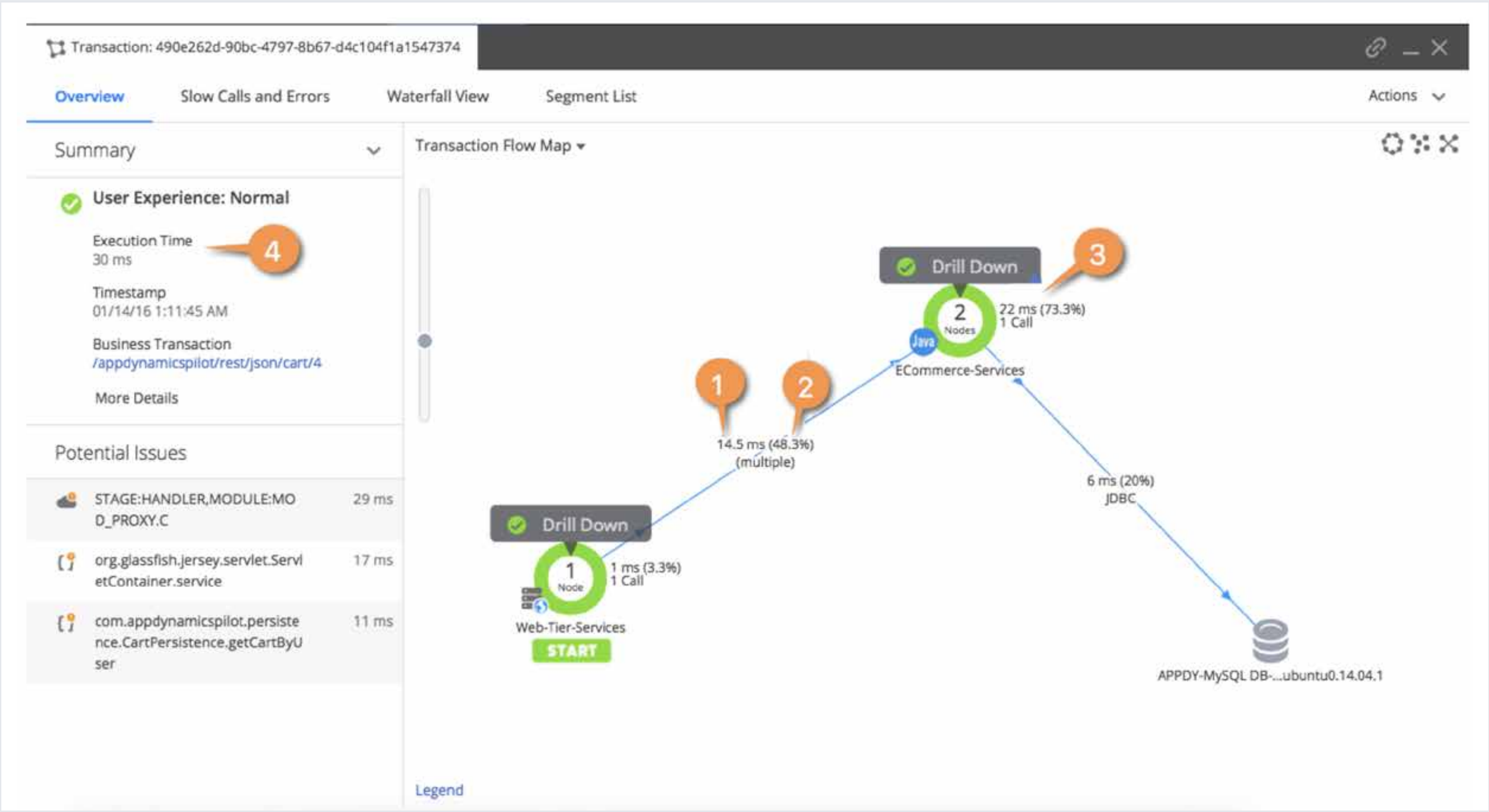
ALERT AND RESPONSE OVERVIEW



TRANSACTION SNAPSHOTS

Transaction Snapshots are a set of diagnostic data for an individual Business Transaction across all servers through which the Business Transaction has passed. This will give you a detailed view into the transaction for deep diagnostics, one slice at a time.

Though diagnosis is ongoing, watching every Business Transaction, AppDynamics APM maintains a low resource overhead, only triggering deep-level diagnostics when performance issues are detected.



Transaction: 490e262d-90bc-4797-8b67-d4c104f1a1547374

Summary

User Experience: Normal

Execution Time
30 ms

Timestamp
01/14/16 1:11:45 AM

Business Transaction
/appdynamicspilot/rest/json/cart/4

More Details

Potential Issues

STAGE:HANDLER,MODULE:MOD_PROXY.C	29 ms
org.glassfish.jersey.servlet.ServletContainer.service	17 ms
com.appdynamicspilot.persistence.CartPersistence.getCartByUser	11 ms

Conclusion

AppDynamics APM has redefined the role of IT, giving technology a seat at the strategic table. By putting the customer experience first, the next logical step is seeing applications in terms of the essential Business Transactions the company depends on for revenue. Everyone in the organization can clearly see the significance of Business Transactions and the financial impact of performance issues that affect them.

WANT TO LEARN MORE ABOUT APPDYNAMICS?

Take a guided tour or schedule a demo today.

GUIDED TOUR

SCHEDULE A DEMO