



| University

Core APM Complete

APM240

Student Guide

Contents

Introduction to AppDynamics (APM211)	3
Introduction to AppDynamics and APM	5
Enterprise Topology and AppDynamics	20
The Application Dashboard.....	33
Business Transactions.....	42
Troubleshooting Basics (APM212)	59
Troubleshooting Overview	61
Diagnostic Sessions	68
Troubleshooting with Transaction Snapshots	73
Troubleshooting Slow Transactions	89
Troubleshooting Error Transactions.....	97
Advanced Troubleshooting and Tools (APM213).....	112
Too Many / Slow Database Calls	114
Thread State Analysis.....	119
Data Collectors	123
Troubleshooting Node Level Issues.....	127
Memory Management.....	131
Monitoring the Health of Your Application (APM214)	144
Service Endpoints and Information Points	146
Introduction to the Alert and Respond Model	154
Metrics and Baselines.....	156
Health Rules.....	166
Events	172
Actions and Policies.....	178
Custom Dashboards.....	185
Additional Monitoring	189
AppDynamics Strategy and Introduction to Business Transaction Discovery (APM221)	198
Steps to an AppDynamics Implementation	200
Business Transaction Instrumentation	217
Automatic Discovery	223
Best Practice: Select Business Transactions by Hand	234
Custom Match Rules	250
Managing Business Transactions (APM222)	266
Business Transaction Management Strategies	268
Service Endpoints.....	292
Live Preview in Custom Match Rules.....	304
Managing Business Transactions Using Discovery Sessions	312
Configuring Backend Detection	321

Proactive Monitoring and Dashboards (APM223)	331
Customizing Error Detection	333
Configuring Baselines.....	343
Configuring Health Rules.....	355
Policies, Actions, and Runbook Automation.....	363
Custom Dashboards and Reports.....	372
Advanced Troubleshooting and Monitoring (APM224).....	384
Creating and Using Information Points	386
Creating and Using Data Collectors.....	391
Using Development Level Monitoring	402
Configuring JMX Metrics and Windows Performance Counters	406
Using Memory Management Tools	415

Introduction to AppDynamics (APM211)



University

APM211 - Introduction to AppDynamics

Core APM I: Essentials - Module 1

Objectives

Course

After completing this course, you will be able to:

- Explain the value of APM
- Explain the concepts behind AppDynamics
- Describe the fundamental components of the AppDynamics architecture
- Use AppDynamics to evaluate the health of your application
- Identify business transactions in your organization
- Explain what a transaction snapshot is

Labs

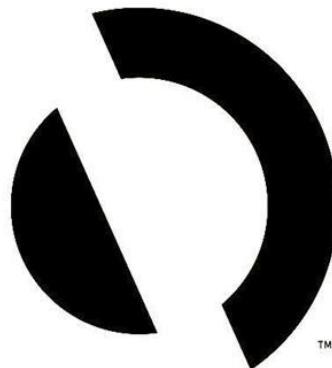
In this course's labs, you will:

- Log in and navigate the User Interface
- Explore the Application Dashboard

Introduction to AppDynamics and APM

Who Are We?

- Founded in 2008
- Purchased by Cisco in March 2017
- Named a Gartner Magic Quadrant leader for APM Suites for 9 straight years



 APPDYNAMICS

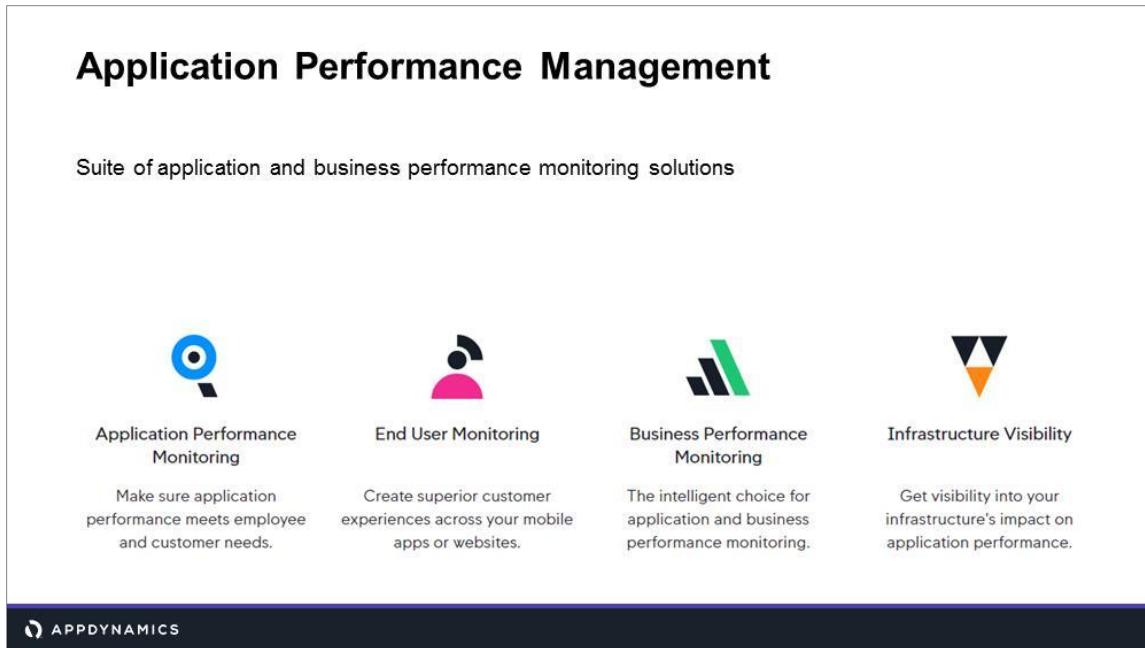
When an application experiences a performance issue, common questions arise: Why is it slow? How long has it been slow? What is causing the slowness?

AppDynamics helps to answer these questions and provides capabilities to:

- Troubleshoot problems such as slow response times and application errors.
- Automatically discover application topology and how components in the application environment work together to fulfil key business transactions for users.
- Measure end-to-end business transaction performance, along with the health of individual application and infrastructure nodes.
- Receive alerts based on custom or built-in health rules, including rules against dynamic performance baselines that alert you to issues in the context of business transactions.
- Analyze your applications at the code execution level using snapshots.

Application Performance Management

Suite of application and business performance monitoring solutions



Application Performance Monitoring
Make sure application performance meets employee and customer needs.

End User Monitoring
Create superior customer experiences across your mobile apps or websites.

Business Performance Monitoring
The intelligent choice for application and business performance monitoring.

Infrastructure Visibility
Get visibility into your infrastructure's impact on application performance.

 APPDYNAMICS

APM at AppDynamics stands for Application Performance Management. We have an entire APM suite of solutions which includes end user monitoring for browser, mobile and IoT, business performance analytics, and logs, server and network monitoring with Infrastructure Visibility.

APM can also stand for Application Performance Monitoring, which is one element of our solution, and the one that we will focus on in this course.

APM vs. APM

Application Performance Management vs. Application Performance Monitoring

The screenshot shows the AppDynamics Home page. The top navigation bar includes Home, Applications, User Experience, Databases, Servers, Analytics, Dashboards & Reports, and Alert & Respond. The main content area is divided into several sections:

- Recently Visited:** Movie2stream - Dashboard
- Applications:** 1 (0 critical, 0 warning, 1 normal). A box highlights "Movie2stream" with a status of "Normal". An orange callout bubble labeled "Monitoring" points to this section.
- Servers:** 1 (0 critical, 0 warning, 1 normal). A box highlights "Movie2streamHost".
- Analytics:** 0 transactions, 0 logs, 0 browser requests, 0 mobile requests.
- User Experience:** Browser Apps: 0. A box highlights "Get Started".
- Databases:** 0. A box highlights "Get Started".
- Dashboards:** 13 (Enterprise Dashboard, Movie2stream - Dashboard, Movie2stream - Development Dashboard, Movie2stream - Help Desk Dashboard, Movie2stream - Operations Dashboard, Movie2stream - QA/Testing Dashboard, Movie2stream - Senior Management Product, Movie2stream - Third Party Payment Provider, M2 Fundamentals Dashboard). A box highlights "Get Started". An orange callout bubble labeled "Management" points to this section.

Supported Languages



Java



.NET



PHP



Node.js



C++



Python



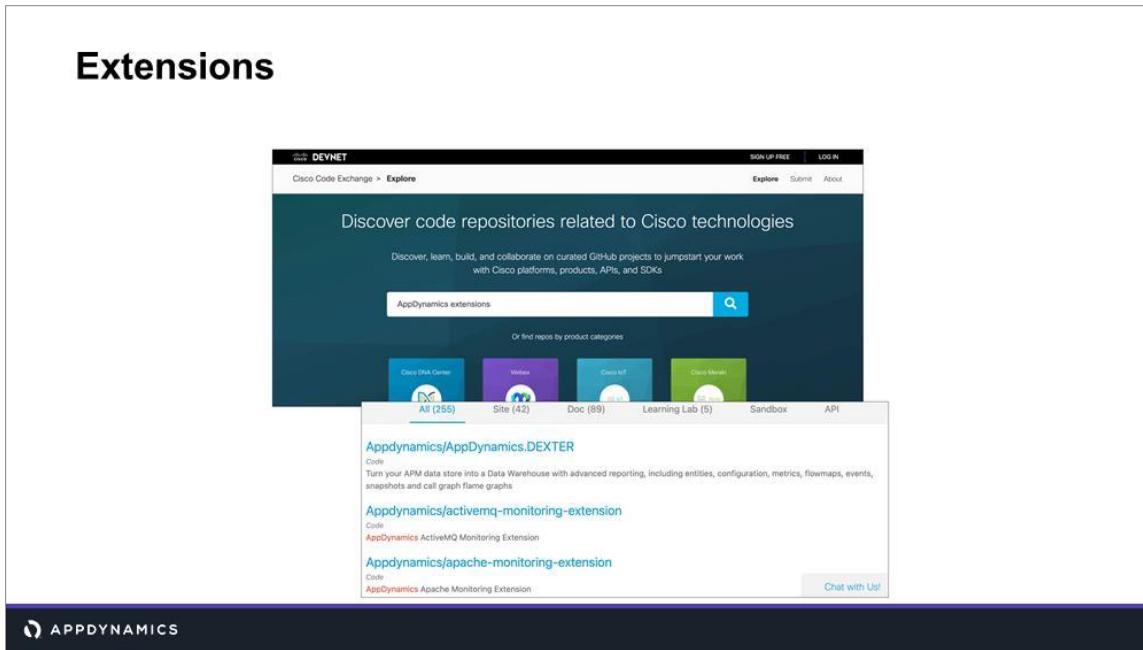
Go



Extensions



AppDynamics currently supports these languages. Extensions (shown on the following slide) allow you to use AppDynamics with an even larger set of technologies.



Extensions provide the ability to use customized metrics or partner with other technologies to extend capabilities. There are monitoring extensions and integration extensions. For monitoring extensions, you can implement custom metrics by using a script, using Java, or using HTTP:

<https://docs.appdynamics.com/display/latest/Extensions+and+Custom+Metrics>

For integration extensions, AppDynamics integrates with 3rd party extensions, for example the AppDynamics ServiceNow REST API Alerting Extension.

You can search for AppDynamics extensions on Cisco DevNet, where you can browse by category and build:

<https://developer.cisco.com/codeexchange/explore/>

Who Uses AppDynamics?



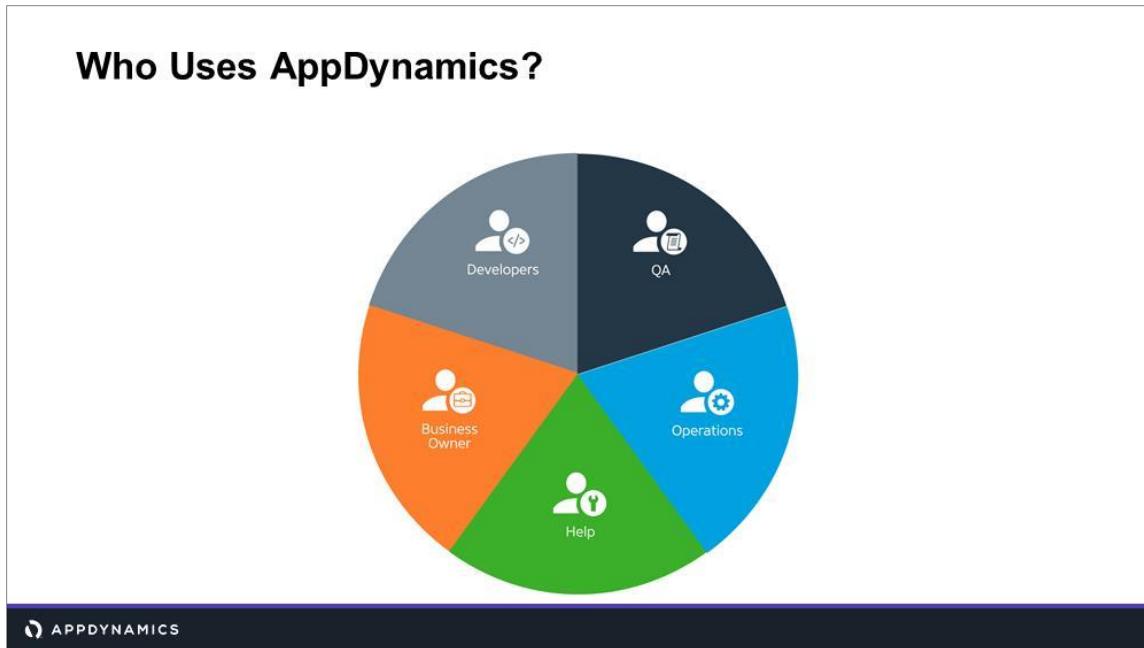
The image contains two circular icons side-by-side. The left icon is dark grey with a white silhouette of a person and a white circle containing '</>' symbols. The word 'Developers' is written in white at the bottom. The right icon is light blue with a white silhouette of a person and a white gear icon. The word 'Operations' is written in white at the bottom. Both icons are set against a white background with a thin black border around the entire image.

Developers

Operations

 APPDYNAMICS

Developers and Operations/Support personnel can get enormous value from AppDynamics. They understand performance, find problems, and can use AppDynamics to see how their code is performing.



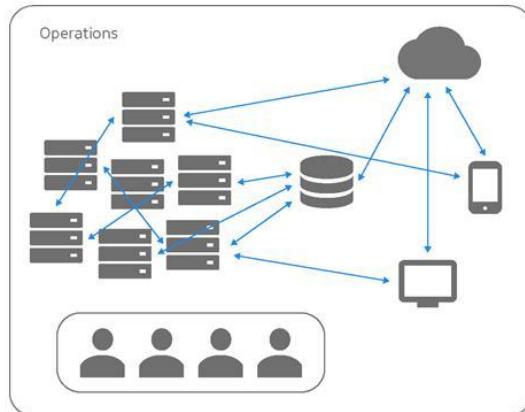
The Customer Experience is the responsibility of everyone in an organization. Using AppDynamics, everyone from the Help Desk to the C-Level executives can and do get value from AppDynamics.

QA and testing can perform load performance tests on sandbox and pre-production environments. They can compare these results to production to see how well production is performing. They can also perform full function testing and if something goes wrong, they can go in and see what went wrong and share that with the development team.

Help Desk and Business Owners can use reporting and dashboards to get a general view of performance, which directly affects customer satisfaction and retention.

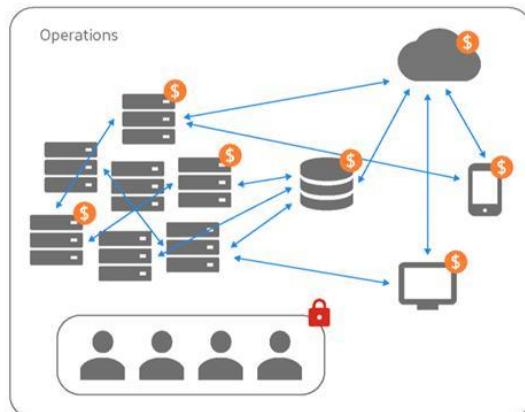
Current State of Affairs

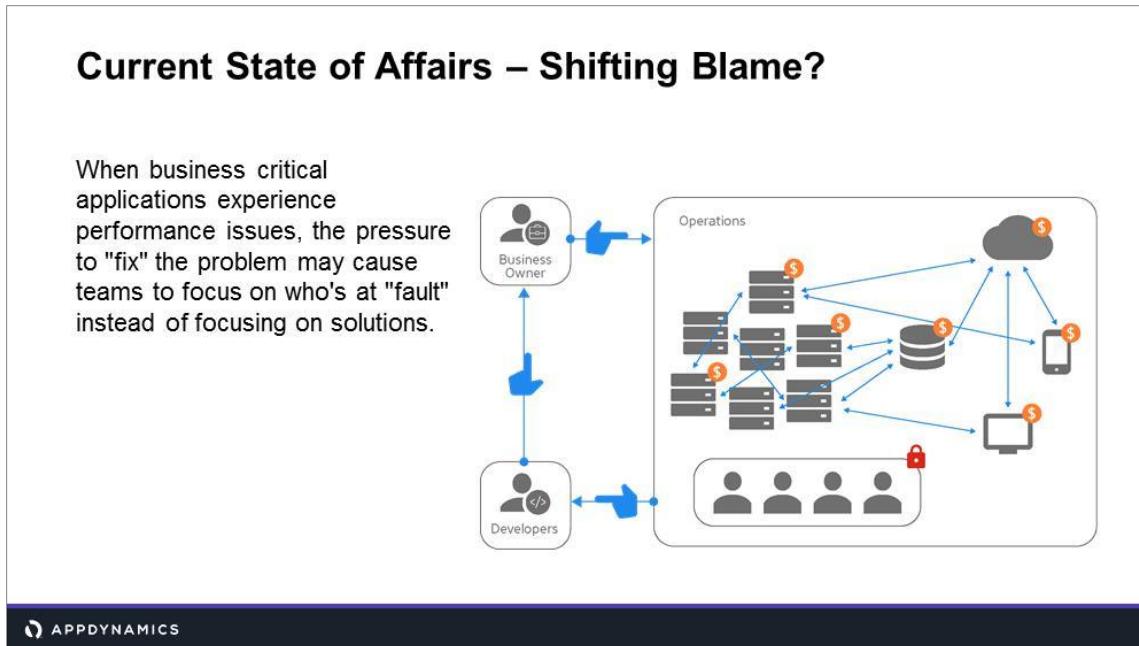
With the rise of the software-defined business, Operations is tasked to support applications that are more and more critical to the business.



Current State of Affairs – Complicating Factors

- Applications are becoming increasingly complex
 - Expensive to run
 - Expensive to troubleshoot
- The degree of interaction between applications is increasing
- Security concerns are locking down environments
- It's difficult to easily see the "Big Picture"





User feedback on application issues are typically anecdotal, and do not necessarily help to determine the root cause.

With real-time visibility and insight into IT environments, AppDynamics allows your organization to stop the "blame game" and instead align DevOps and business owners around information that helps to deliver flawless customer experiences at scale.



One of the results of applications that experience slow performance or downtime is an angry customer.

Downtime Costs

- For the Fortune 1000, the average total cost of unplanned application downtime per year is \$1.25 billion to \$2.5 billion
- The average hourly cost of an infrastructure failure is \$100,000 per hour
- The average cost of a critical application failure per hour is \$500,000 to \$1 million
 - *Source: IDC Study - The cost of downtime*



© APPDYNAMICS

Source: IDC Study - The cost of downtime <https://www.techrepublic.com/resource-library/whitepapers/idc-study-the-cost-of-downtime/>

What About Your Downtime?

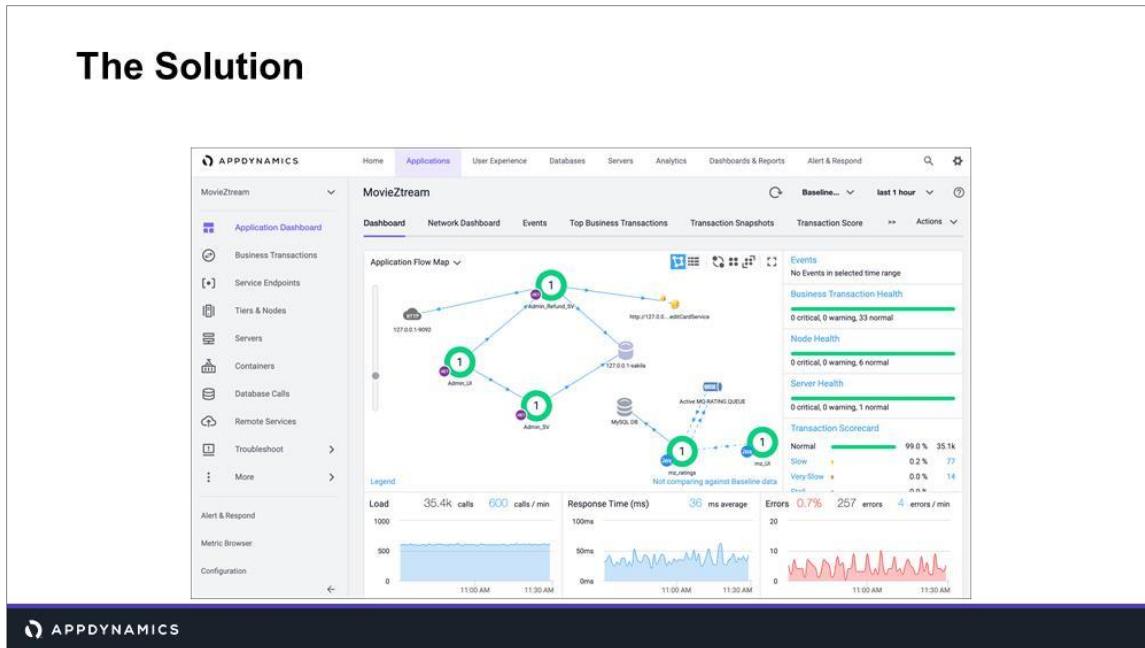
- What is your current Mean Time to Detect?
- What is your current Mean Time to Diagnose?
- What is your current Mean Time to Repair?



© APPDYNAMICS

You may not currently know the answer to these questions, but they are important questions that you should be asking about your application.

AppDynamics not only can help answer them, it can help reduce the amount of time to detect, diagnose and repair.



AppDynamics can take this locked down, complex application that communicates with other applications and databases, and provide meaningful insight by bringing in all of the performance information from production. It allows users both see the big picture and provides them with the ability to zoom in to a more detailed look at specific problems.

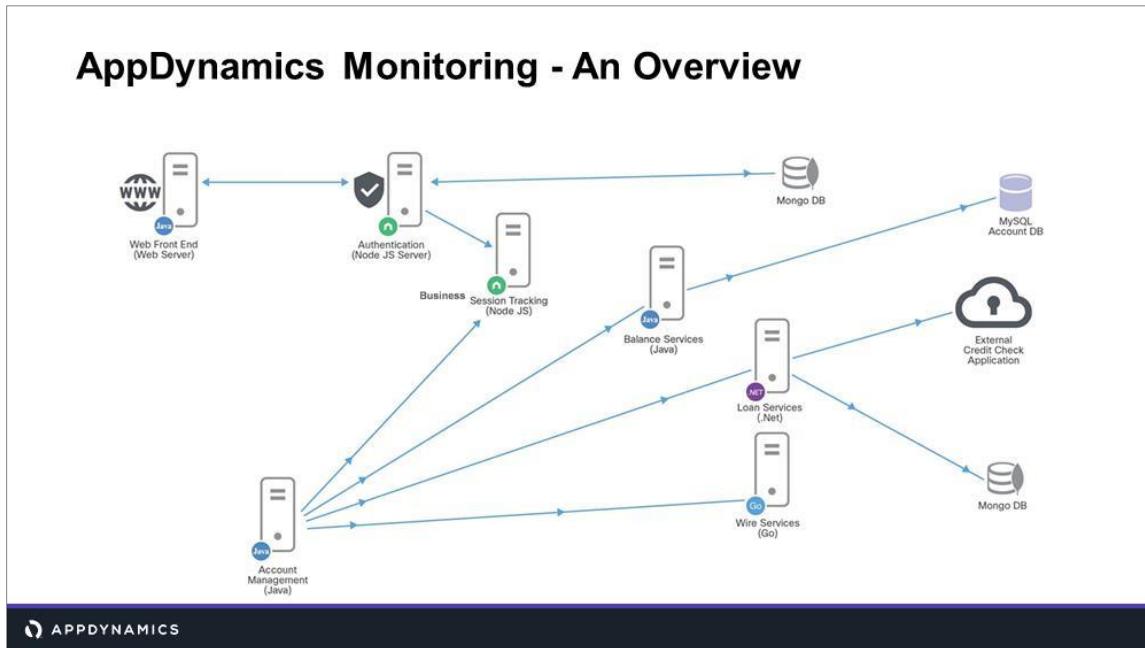
AppDynamics provides a single view of the application where everyone can see exactly what is going on via the Controller. App Agents are installed on JVMs, .NET applications, PHP web servers and Node.js. These agents send performance data to the controller.

Review Question - Answer

AppDynamics Application Performance Management suite includes tools that allow you to:

- A) Monitor application performance.
- B) Track user experience starting with the browser or mobile app.
- C) Analyze business metrics.
- D) Assess how your hardware and network are impacting the performance of your application.
- E) Track how search engines rank your application pages.

Enterprise Topology and AppDynamics

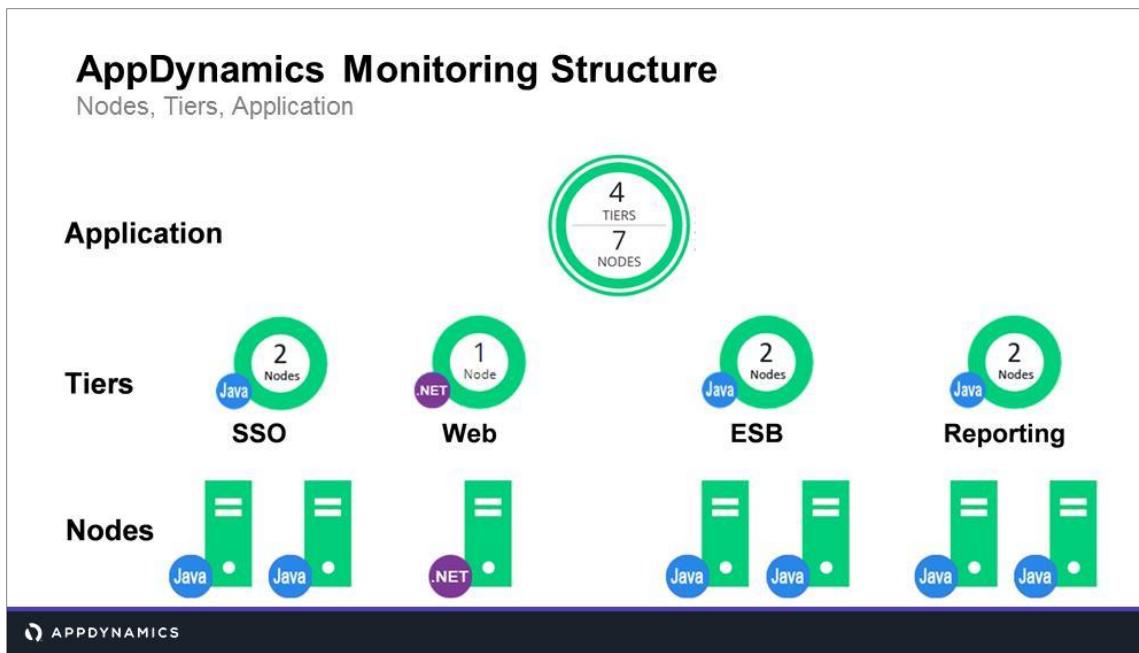


Let's walk through how it works at a high level using the example of a user logging on to your application.

A user accesses the login page on the web or intranet. When the login request is sent to the webserver, Application Performance Monitoring tags and follows the request for its duration. It looks at the communication stream of that request – how much time did the request take to process? Were there any downstream calls? How many? Were there any errors? What were they?

If a database is involved, metrics around database communication and processing can be captured as well. Calls to external web services are also tracked.

Before we look at how AppDynamics does this, let's look at the Application – Tier – Node structure within AppDynamics.



Nodes:

A node is the basic unit of processing that AppDynamics monitors. Nodes belong to tiers and a node cannot belong to more than one tier.

In AppDynamics, a node is instrumented and monitored by an AppDynamics agent.

An AppDynamics tier represents an instrumented service (such as a web application) or multiple services that perform the exact same functionality, and may even run the same code. In the AppDynamics model, a tier is composed of one node or multiple redundant nodes. For example the inventory tier may have one node whereas the ecommerce tier may use 3 nodes. Each node in a multi-node tier typically provides identical functionality. One example of a multi-node tier is when you have a set of clustered application servers or services.

Tiers:

Tiers, along with applications, are solely logical (not physical) grouping mechanisms.

In order to “map” your application environment correctly to the AppDynamics model, you need to understand what an application means in AppDynamics.

Application:

An application is a grouping of one or more related customer applications that are closely related by business function.

You can map multiple applications that are connected as a single AppDynamics monitored application, or keep them separate. AppDynamics can track cross-application requests, so if a transaction makes a call to another application, you can still trace the call and troubleshoot.

Notes From the Field

Applications, Tiers, and Nodes

Business Problem

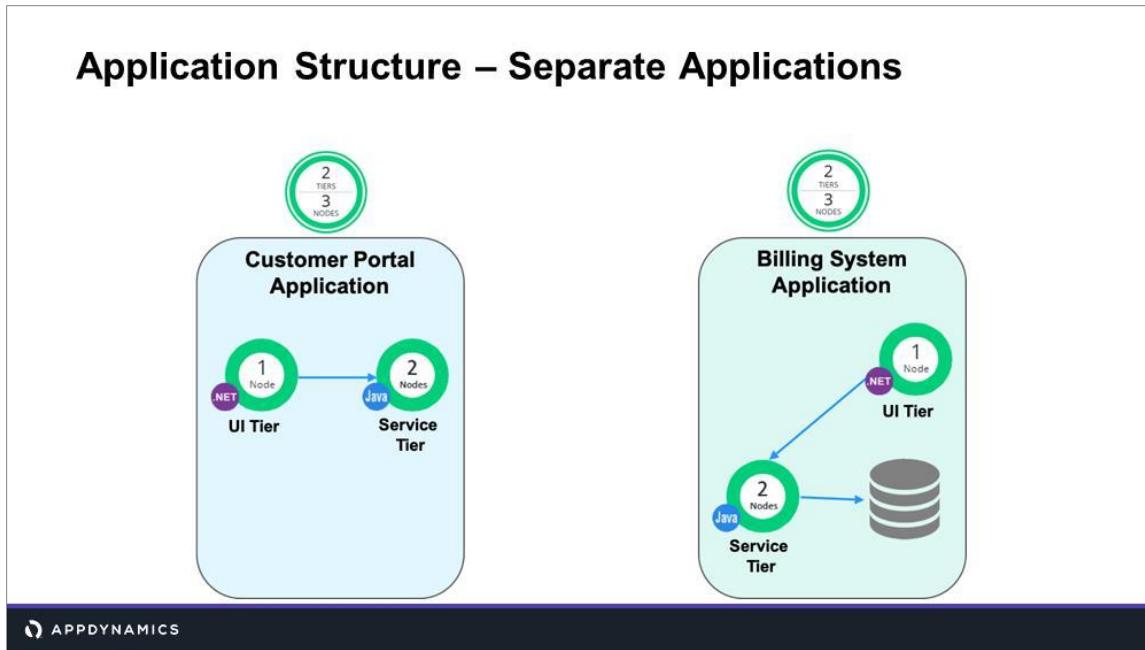
A customer originally instrumented a number of key systems (Customer Portal, Billing, etc.) separately, inadvertently making troubleshooting more difficult.

Solution

They combined these AppDynamics applications into one application called Enterprise. To make management easier, they prefixed the tier names with the system names. E.g.,

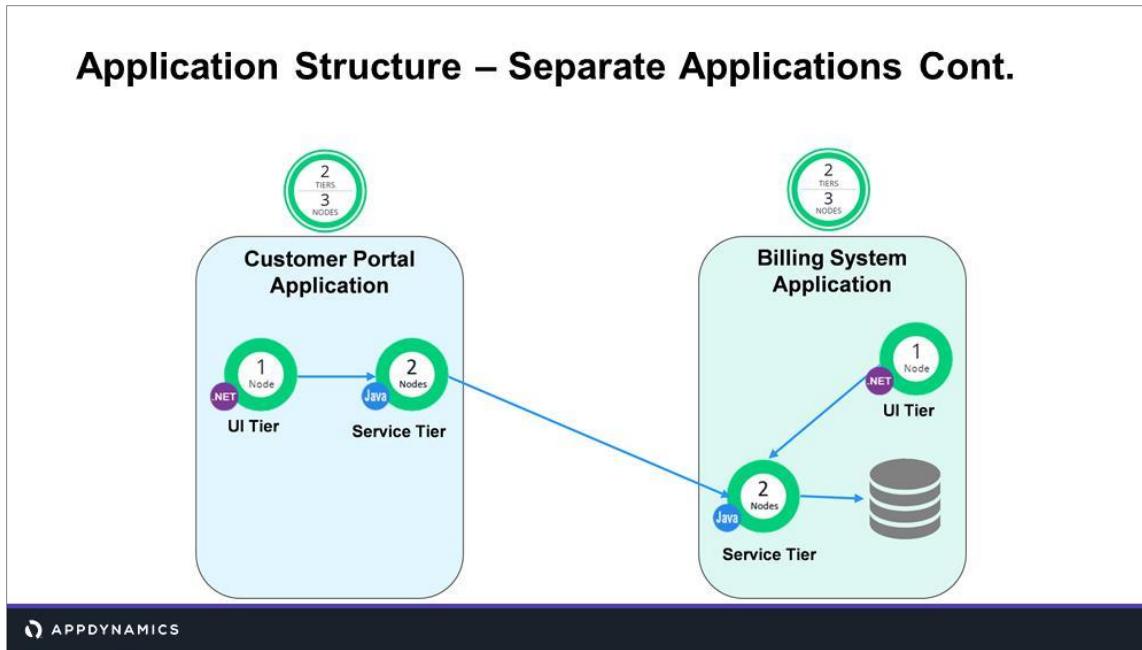
- Customer Portal - Web
- Customer Portal - Service
- Billing - Web, etc.





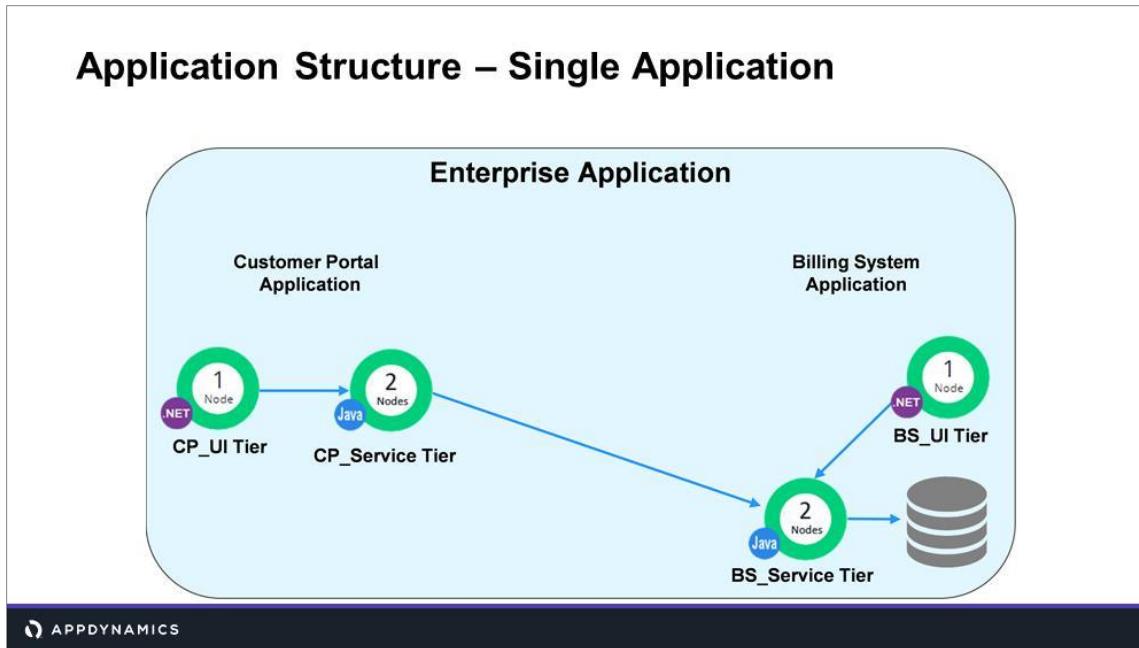
Let's look at how this might be set up in AppDynamics.

Let's say you have two applications – a Customer Portal Application and a Billing System Application. Each of these has a UI Tier and a Service Tier. The Billing System Application also connects to a database.



A user wants to pay their bill. That request starts in the UI tier of the Customer Portal and then makes a call to the Billing System Service Tier.

In this configuration, if the “Pay Bill” request has an error, you would need to check both applications in AppDynamics in order to determine the root cause of the error.



A better way might be to create one AppDynamics application and then prefix the tiers with the system names.

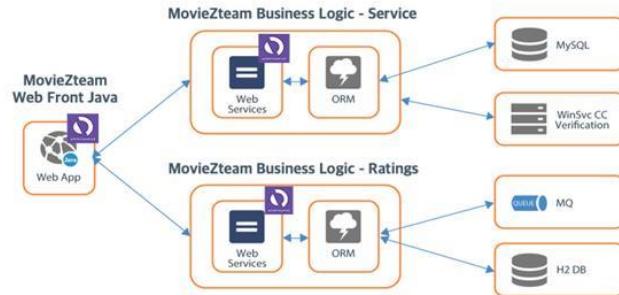
This setup means that you only have to check one application to find the error in the 'Pay Bill' request. Generally, you would be able to more easily see the full lifecycle of transactions.

One potential downside is that you will end up with a potentially more complicated AppDynamics application with many monitored transactions. And there is a 200 transaction limit per Application.

Now that we know how applications are structured, let's look at how AppDynamics monitors your application.

What is an Agent?

- A code snippet
- Plug-ins or extensions that monitor code, runtime and behavior
 - GUIDs are assigned to every request



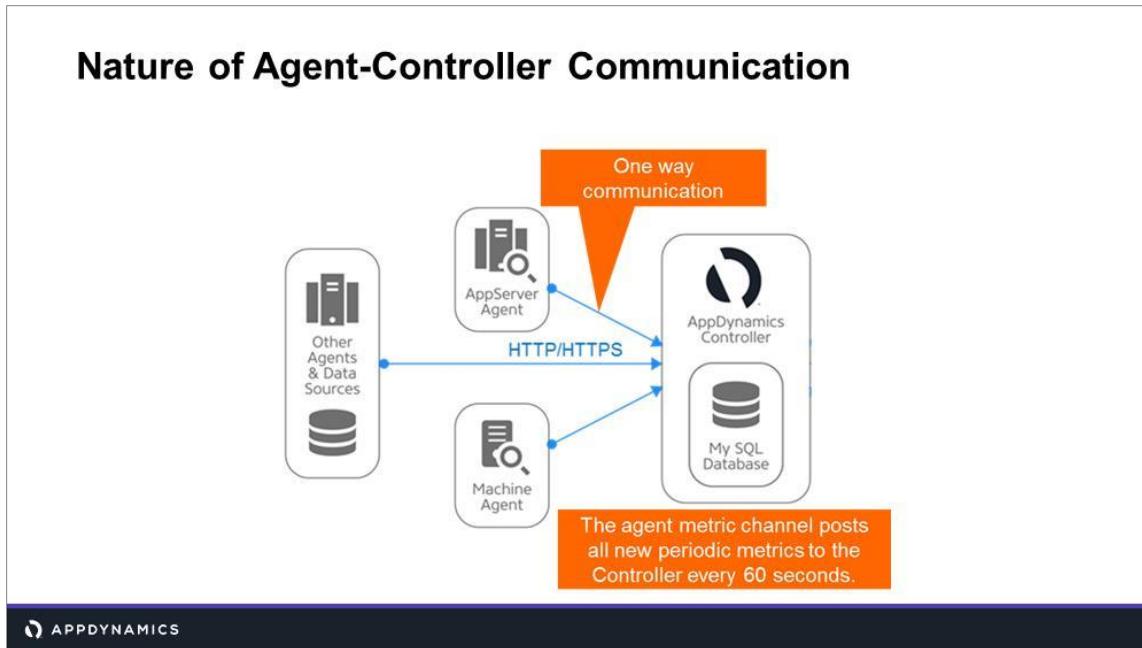
Agents are plug-ins or extensions that monitor the performance of your application code, runtime, and behavior.

There are a number of different agents in Appdynamics - including the Analytics Agent and the EUM Agent. We're going to focus on Application Agents, or App Agents.

Once deployed, App agents immediately monitor every line of code.

Unique tags called GUIDs or (Global Unique Identifiers) are assigned to every request and passed along to every method call and every downstream request header.

This allows AppDynamics to trace every transaction from start to finish—even in modern, distributed applications.

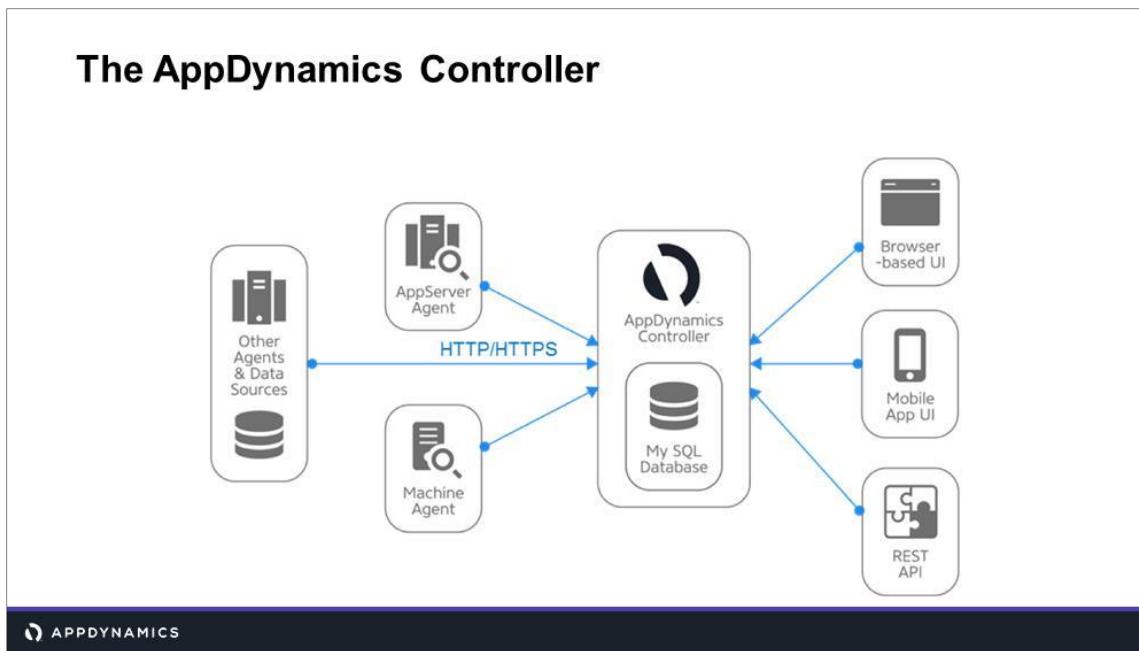


Notice that the communication arrows only go one way - from the Agent to the Controller. However, please note that while the communication is always initiated by the Agent, the communication can go both ways once the communication channel has been established. Once the communication channel is established, the Agent can pull new property configurations from the Controller.

Agents communicate with the controller in the following manner:

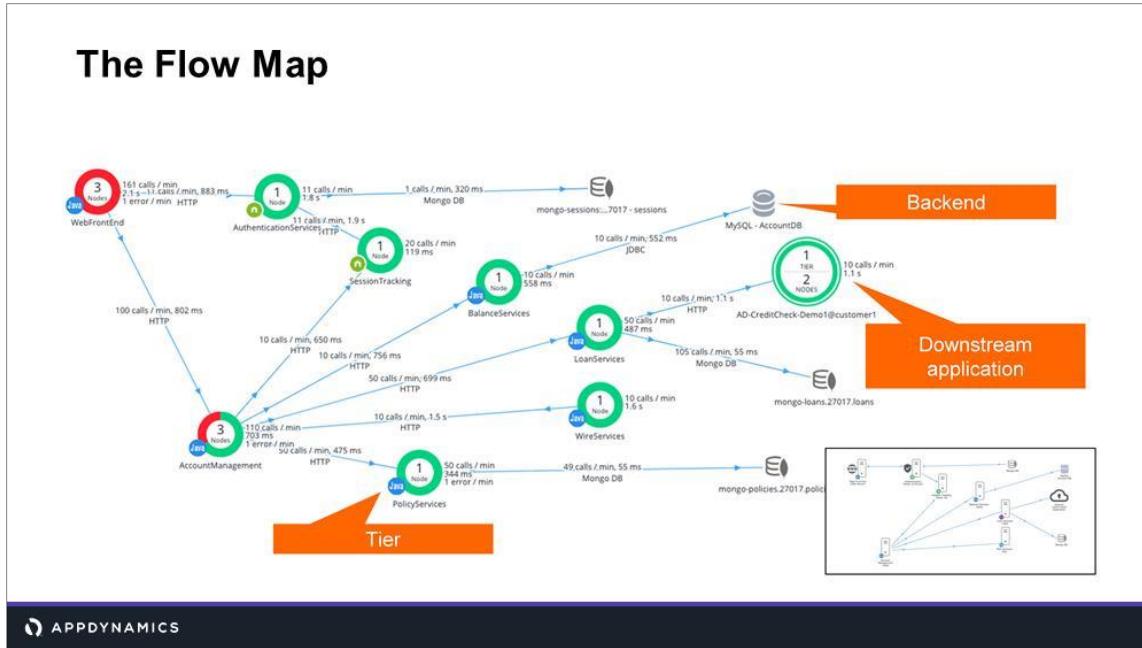
- The agent configuration channel queries the Controller for any new configuration changes every 60 seconds and downloads the changes when they are available.
- The agent metric channel posts all new periodic metrics, including JMX, Windows performance counters, and business transaction metrics to the Controller every 60 seconds.
- If there are new business transactions that haven't been seen before by the agent, they are posted to the Controller for registration every 10 seconds.
- If the agent has collected any new snapshots or events, they are posted to the Controller every 20 seconds.

Configuration changes can sometimes take 2 to 3 minutes for the changes to actually show up.



Agents capture performance activity and send it to the Controller, which is updated in real-time. The Controller helps monitor, troubleshoot and analyze your entire application landscape—from backend infrastructure to the end user—in one simple interface.

AppDynamics supports the ability to trace calls to your applications even if these calls jump from one application to another. Sometimes calls can cross Controller boundaries, where one Controller gets call data from the next app in the call chain. AppDynamics supports the concept of “drill-down” if you need to track a call-chain across this Controller boundary.



The Controller displays your application structure in a flow map. A flow map is an interactive, graphical display of your application, tiers, nodes, and backends. It shows the tiers and nodes traversed by the transaction, the number of calls, and the processing time. It also uses color to denote the health of the tiers, as well as any downstream applications - green for healthy, yellow for warning, and red for critical.

With flow maps, you can:

- Quickly determine the health of your application and tiers
- Access any additional linked applications
- View metric call data for tiers and nodes
- Click items to drill down into the next level

You can also create custom flow maps, for example, if you were only interested in a particular set of functions, you could create a flow map to only show those tiers.

Flow maps exist throughout the Controller. All applications, a single application, tier and node flow maps, and transaction flow maps show you varying levels of granularity.

Review Question - Answer

An Application in AppDynamics is:

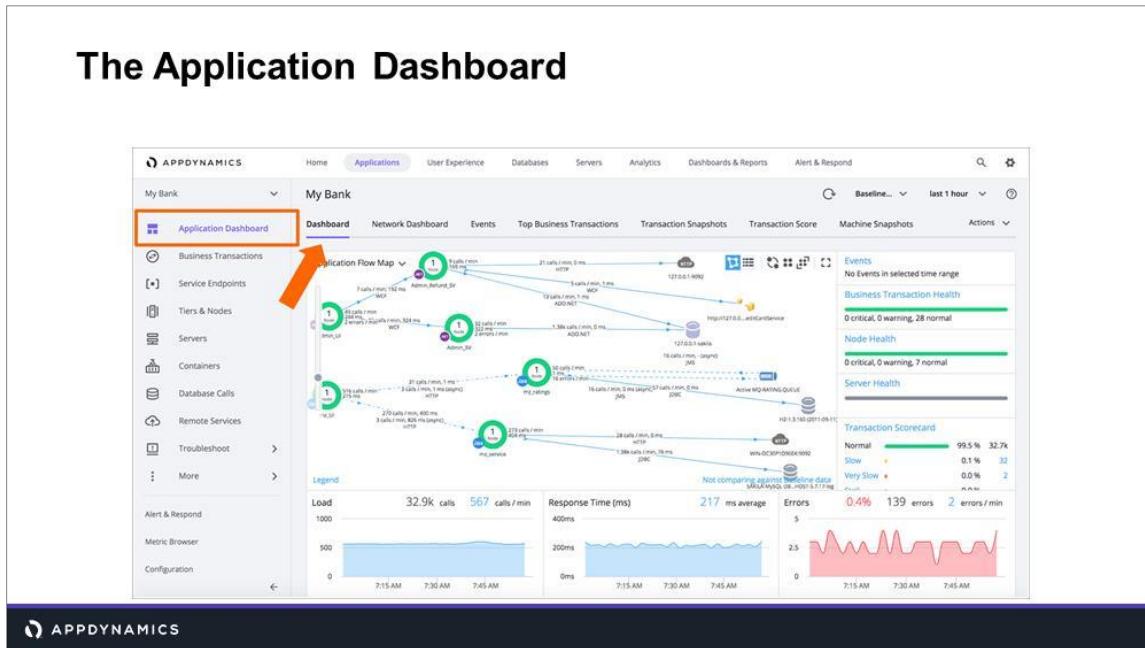
- A) The visual display of your tiers, nodes, and backends.
- B) A grouping of one or more related customer applications that are closely related by business function.**
- C) An agent that monitors every line of code with minimal overhead.
- D) An instrumented service (such as a web application) or multiple services that perform the exact same functionality.
- E) The basic unit of processing that AppDynamics monitors.

Review Question - Answer

A Tier in AppDynamics is:

- A) The visual display of your application, nodes, and backends.
- B) An agent that monitors every line of code with minimal overhead.
- C) A named collection of nodes, typically representing a service (such as a series of Web Servers).**
- D) The basic unit of processing that AppDynamics monitors.

The Application Dashboard



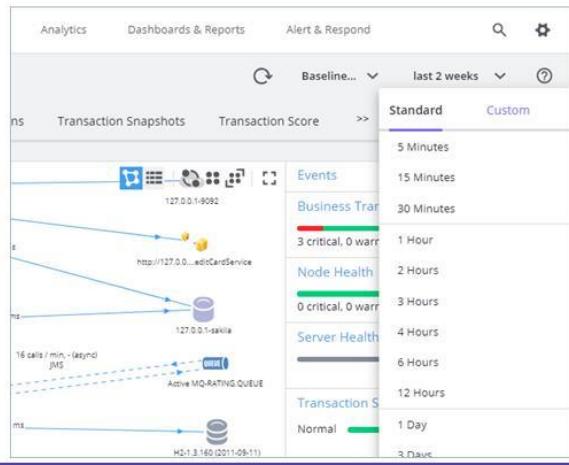
The Application Dashboard is the first thing you see when you open your app, and is sort of grand central station for health and metric data for your application.

It contains your application flow map, as well as key metric charts that can show you the general health of your application.

Across the top, you can access other tabs that contain information specific to this application – events, top business transactions, and transaction snapshots, for example.

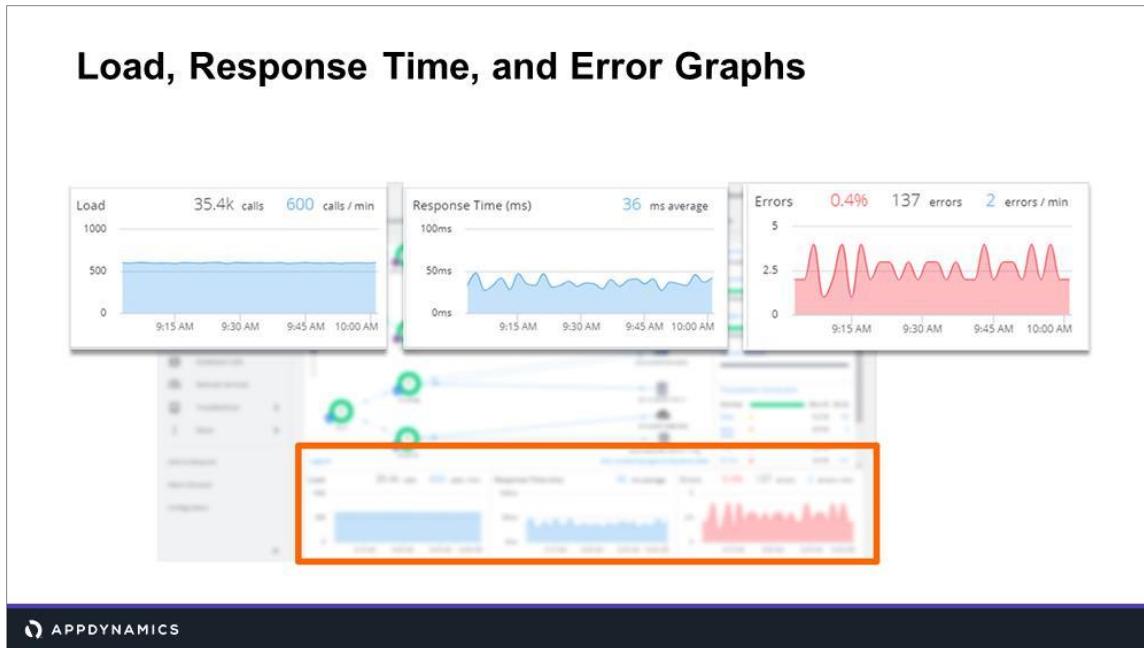
Time Period

- Select from a list of standard time periods
- Create your own custom time period



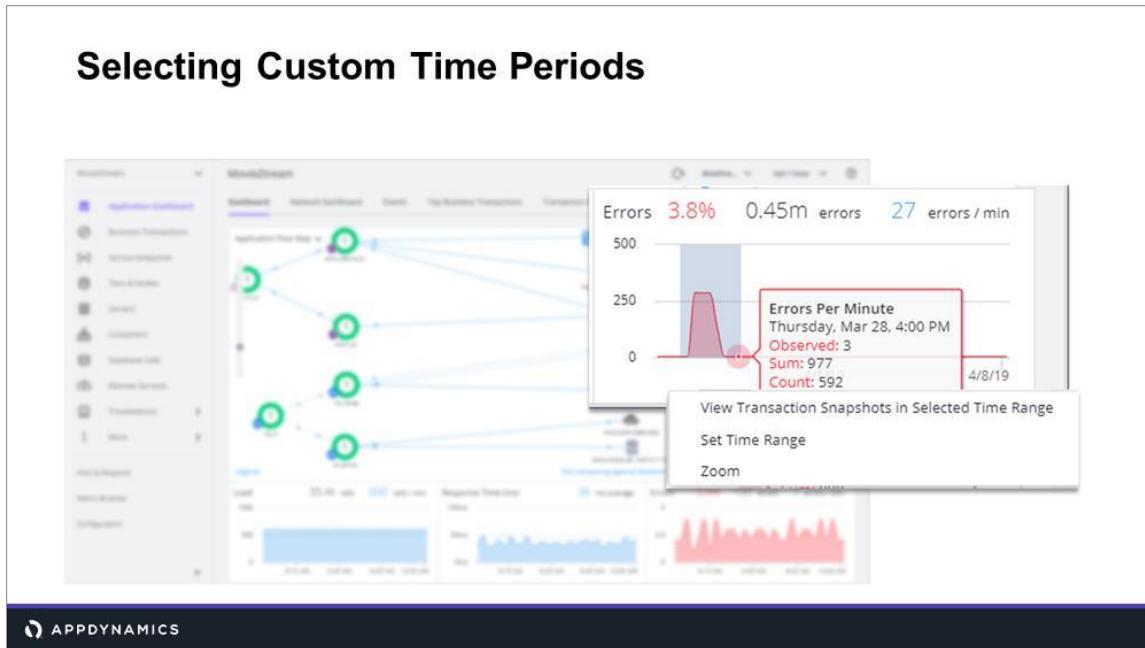
One of the important things to keep in mind about what you're seeing on the application dashboard is that the metrics are show for the selected time period. You can change the time period by selecting from a list of commonly used standard time periods, or you can create your own custom time period.

Time Zones - Rather than translate the timestamp on snapshots and metrics to another user's time zone, you can adjust the time zone setting to collaborate and troubleshoot issues with more accuracy. Align your time zone with other users by going to the Gear Icon in the top right > My Preferences, and use the Display Time Zone dropdown to change your time zone. You will then need to reload the browser.

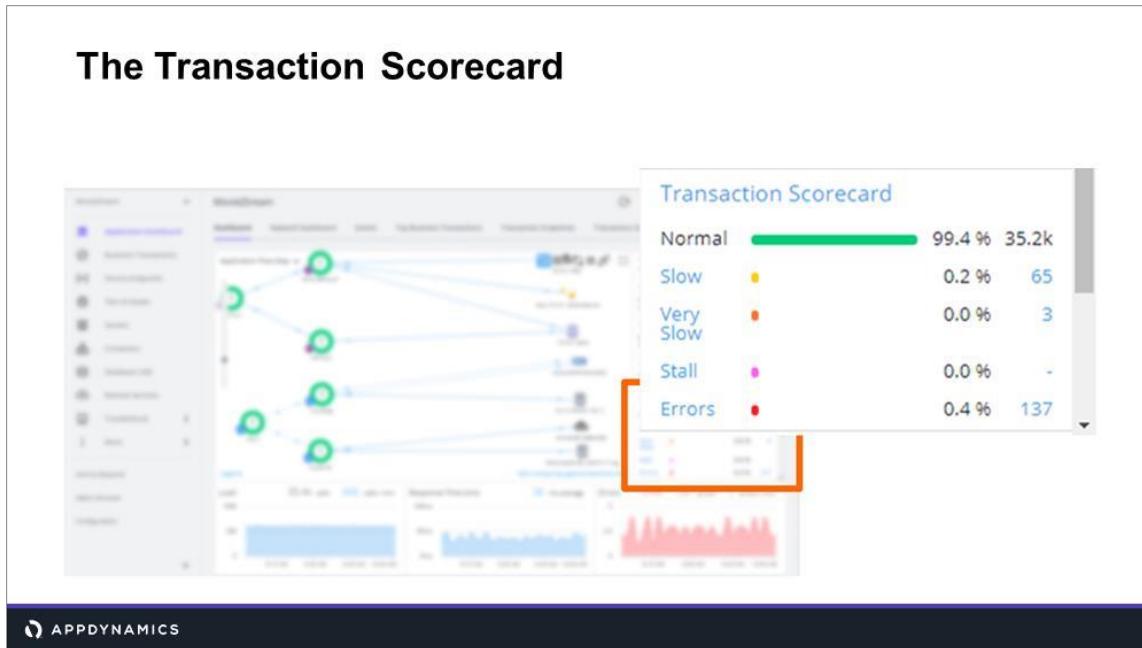


There are three graphs across the bottom of the Application Dashboard.

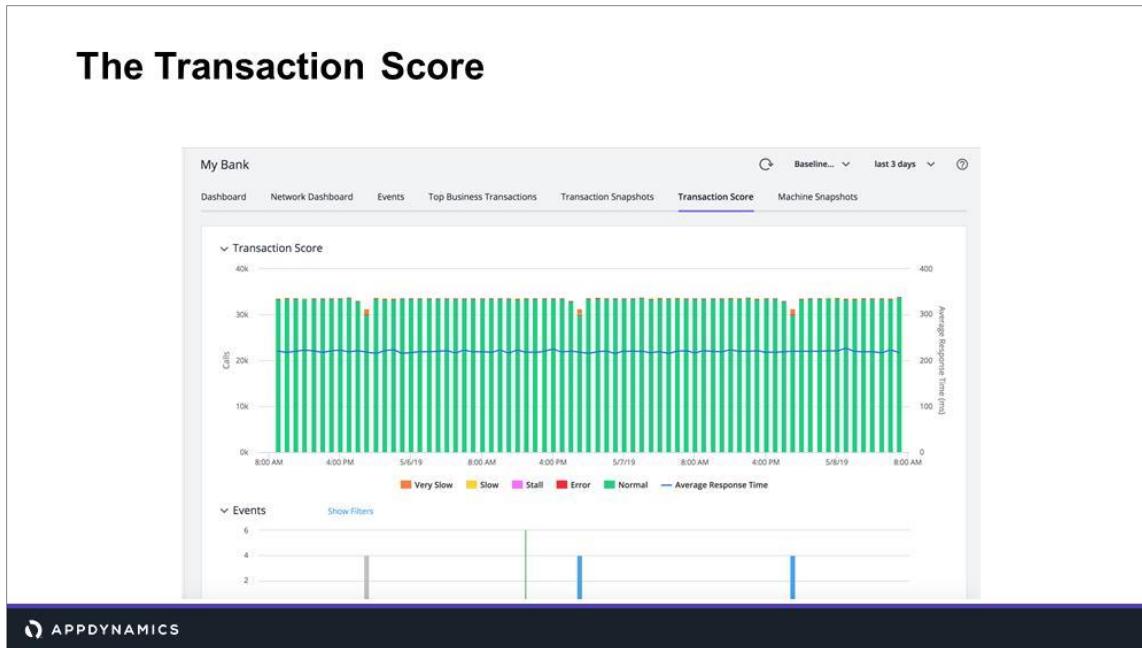
They show you the overall load, response time, and number of errors for the entire application for the selected period of time. It's a good place to look for spikes, or even trends or patterns.



You can also click and drag your mouse over areas of interest in these graphs, and AppDynamics will ask if you want to set the time range to match your selection. This is extremely useful in focusing on issues.



The Transaction Scorecard is both a visual representation of how your transactions are performing based on your thresholds. It's also interactive, and allows you to click the different thresholds to open a list of those transactions.



The Transaction Score is opens a detailed view of the performance of your transactions over time. You can click the items in the legend to add or remove that performance level. For example, if you want to remove the normal transactions so that you're just looking at slow, very slow, stalled and error, you would click Normal.

Notes From the Field

Using Time Period to Find Issues

Issue

An Analyst's Manager would call and just say "It's slow. Can you fix it?"

Solution

The Analyst would open the application dashboard and look at the Response Time widget. She would then drag her cursor across the widget to focus on the point where response times increased. She was able to look at the slow and very slow transactions at the specific time when they started to be slow.



Review Question - Answer

What are ways of setting the application dashboard time range?

- A) Select a standard time range from the time range drop down.
- B) You can't - the time range is always set to the current hour of data.
- C) Create a custom time range from the time range drop down.
- D) Click and drag your mouse across an area on the Load, Response Time, or Error charts at the bottom of the dashboard.

Review Question - Answer

Application Dashboard panels and flow maps allow you to drill down into an increasing level of detail.

True

False

Business Transactions

Business Transactions

Application traffic is organized into Business Transactions

Each represents a distinct logical user activity (login, search, checkout, etc.)

Most performance monitoring and troubleshooting activities occur at the Business Transaction level



© APPDYNAMICS

Application traffic is organized into business transactions. In AppDynamics, a business transaction represents a distinct logical user activity (login, search, checkout, etc.) , or something that has been invoked.

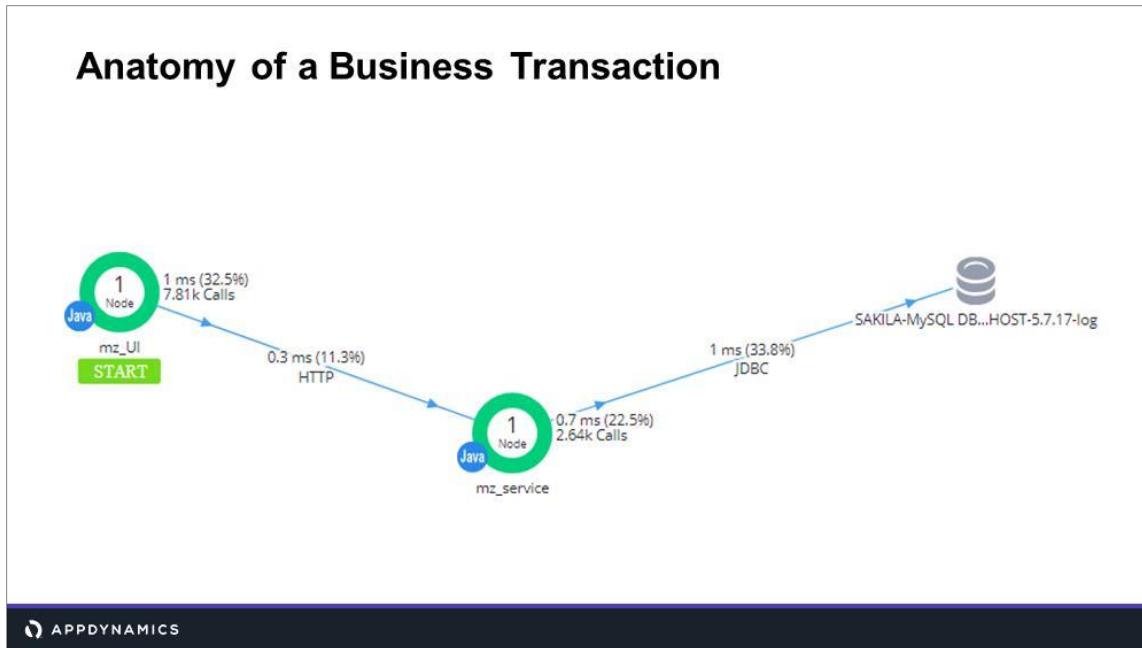
For example, on this banking website, you have a menu of tasks that you can perform. You can sign in, check your balance, or check the current savings account interest rates.

Each of these tasks is a business transaction in AppDynamics. AppDynamics traces the transaction context across all tiers, including JVMs, CLRs, and calls to HTTP, JMS, SOAP, databases, third-party web services, etc. The traced activities are bundled together to make a business transaction. So for example, all the activities that happen as a user logs in are grouped together and represented by one business transaction called Login.

Business Transactions allow you to view your application traffic in a way that aligns with the primary functions of your web business. This allows you to examine the performance of your application as your users experience it.

Most performance monitoring and troubleshooting activities occur at the Business Transaction level, so it's important to understand how they work.

Let's take a closer look at what this looks like in a flow map format.



Let's look at the high level anatomy of this in a flow map format.transaction

The transaction starts in the mz_UI tier, which took 1 ms to process or 32.5 percent of the total transaction time. It then made an HTTP call to the mz_service tier which took .3 ms. The mz_service tier then took .7 ms to process the request, and made a database call. That call took 1 ms. You can add up the total time of each piece to determine the total transaction time.

Managing Business Transactions

Business Transactions that are displayed under the main tab are referred to as "First Class" Business Transactions.

Name	Health	Response Time (ms)	Calls / min	% Errors	% Slow Transactions
/customer/rate.htm	Red	36	32	0	0.1
/customer/rent.htm	Green	43	58	0	0.2
/customer/rents.htm	Green	24	29	0	0.7
/movie/search.htm	Green	52	86	0	0.7
/mybank/ui/balance.htm	Green	1	90	0	0
/mybank/ui/cPay.htm	Green	5	132	0	0.1
/mybank/ui/loanApply.htm	Green	1,377	2	100	0

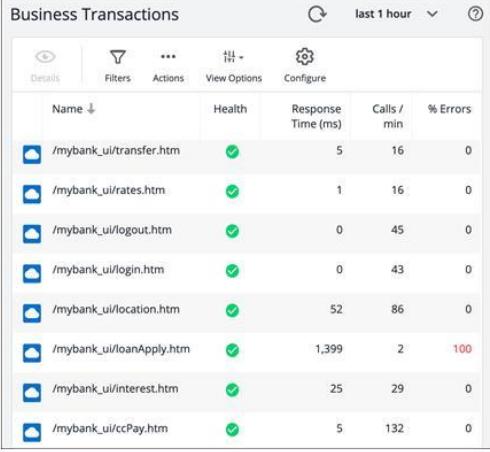
When AppDynamics is installed, it begins to detect business transactions using default automatic detection rules. Configuring transaction detection is covered in the advanced course.

You can access and manage your business transactions from the left menu.

The Business Transactions page allows you to see all of your currently monitored business transactions as well as their performance metrics. You can view summary information for each transaction's performance. You can sort, filter, and search for specific transactions. And you can use the more actions to rename, delete, exclude, and configure thresholds for each transaction.

Business Transaction Strategy - Identify Key Transactions

- Identify the 10-20 most critical functions of the application
 - Business value
 - Customer impact
- This will be covered in depth in the Core APM Advanced course



APPDYNAMICS

It's important to identify the key transactions that are the most important to you in terms of business value and customer impact. This is especially true with larger applications that have a lot of transactions, but even with smaller applications, determining which transactions are key to your business helps keep you focused.

Identify the 10-20 most critical functions of the application—the ones that prompt a midnight call if they fail.

Setting Performance Thresholds



How do you define slow? Very slow? Stalled?

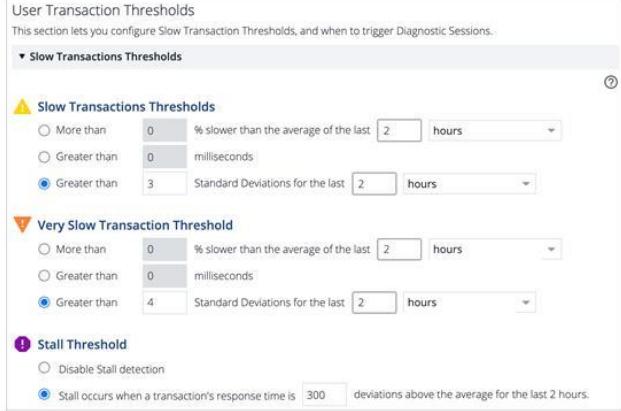
Are these the same for every transaction?

What should happen if these transactions fall below their thresholds?

Develop KPIs and Thresholds

Configure thresholds for each key transaction:

- In milliseconds
- Percentage relative to average
- A number greater than standard deviations for a specific time period



User Transaction Thresholds
This section lets you configure Slow Transaction Thresholds, and when to trigger Diagnostic Sessions.

Slow Transactions Thresholds

Slow Transactions Thresholds

More than 0 % slower than the average of the last 2 hours

Greater than 0 milliseconds

Greater than 3 Standard Deviations for the last 2 hours

Very Slow Transaction Threshold

More than 0 % slower than the average of the last 2 hours

Greater than 0 milliseconds

Greater than 4 Standard Deviations for the last 2 hours

Stall Threshold

Disable Stall detection

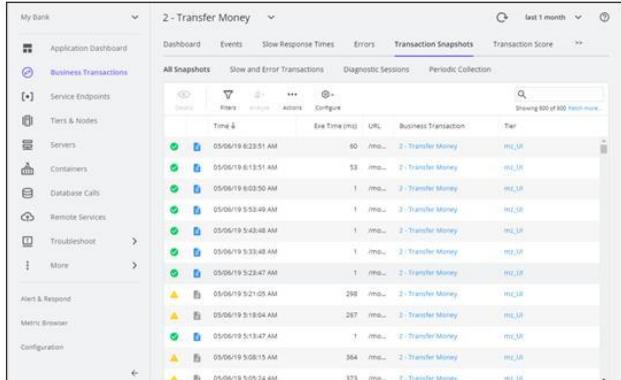
Stall occurs when a transaction's response time is 300 standard deviations above the average for the last 2 hours.

You can configure thresholds for each key transaction, which means you have control over the definition of slow, very slow, or stalled for each key transaction.

Thresholds can be set in milliseconds, percentage relative to average, and the number of standard deviations for a specific time period.

Transaction Snapshots

- Taken for specific instances of a transaction
 - Periodically
 - When something significant happens
- Contains details on that one request



The screenshot shows the AppDynamics interface with the 'Transaction Snapshots' tab selected. The main pane displays a table of transaction snapshots for a transaction named '2 - Transfer Money'. The table includes columns for Time, Execution Time (ms), URL, Business Transaction, and Tier. The data shows multiple executions of the transaction over a period of time, with execution times ranging from 53 ms to 373 ms. The tier for all entries is 'mt2_SR'.

Until now, we've talked about business transactions at the top level. AppDynamics monitors every execution of a business transaction in the instrumented environment, and the metrics reflect all executions.

However, for troubleshooting purposes, AppDynamics takes snapshots of specific instances of a transaction. A transaction snapshot gives you a view of the processing flow for a single invocation of a transaction.

Snapshots can be taken periodically (every 10 minutes), or when something significant happens such as a slow, stalled, or error transaction. Diagnostic Sessions can also be used to capture snapshots. Diagnostic Sessions will be covered in detail later.

Snapshot collection limits prevent excessive resource consumption at the node level:

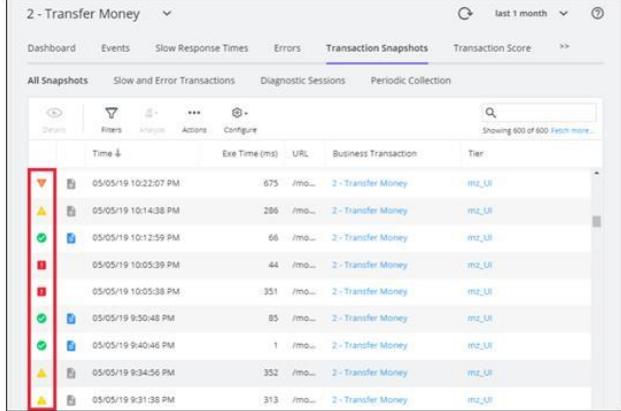
- Originating transaction snapshots are limited to a maximum of 20 originating (5 concurrent) snapshots per node per minute.
- Continuing transaction snapshots are limited to a maximum of 200 (100 concurrent) snapshots per node per minute.

AppDynamics applies snapshot collection limits to error transactions as well. As a result, not every error occurrence that is represented in an error count metric, for example, will have a corresponding snapshot. For error transactions, the following limits apply:

- For a single transaction, there is a maximum of two snapshots per minute.
- Across transactions, the maximum is limited to five snapshots per minute. (Specified by the node property max-error-snapshots-per-minute.)

Transaction Snapshots

Transaction Snapshots include the User Experience



Time	Exec Time (ms)	URL	Business Transaction	Tier
05/05/19 10:22:07 PM	675	/mo...	2 - Transfer Money	mtz_UI
05/05/19 10:14:38 PM	286	/mo...	2 - Transfer Money	mtz_UI
05/05/19 10:12:59 PM	66	/mo...	2 - Transfer Money	mtz_UI
05/05/19 10:05:39 PM	44	/mo...	2 - Transfer Money	mtz_UI
05/05/19 10:05:38 PM	351	/mo...	2 - Transfer Money	mtz_UI
05/05/19 9:50:48 PM	85	/mo...	2 - Transfer Money	mtz_UI
05/05/19 9:40:46 PM	1	/mo...	2 - Transfer Money	mtz_UI
05/05/19 9:34:56 PM	352	/mo...	2 - Transfer Money	mtz_UI
05/05/19 9:31:38 PM	313	/mo...	2 - Transfer Money	mtz_UI

APPDYNAMICS

Transaction Snapshots give you a cross-tier view of the processing flow for a single invocation of a transaction. They include the user experience based on thresholds that are default in AppDynamics and which you can customize.

Transaction Snapshot Information

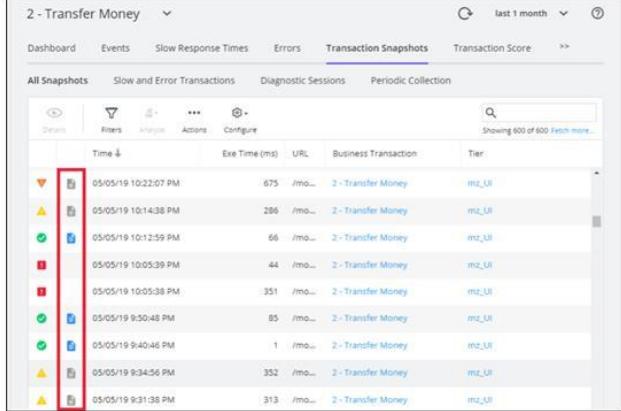
Full vs. Partial Call Graphs

Indicates a full call graph

- Lists methods in call stack
- Contains info about each call
- Periodic Snapshots
- Diagnostic Sessions

Indicates a partial call graph

- Represents subset of call sequence
 - From the point of slowness / error
- Slow and Error* Snapshots
 - *May contain no call graph



Time	Exec Time (ms)	URL	Business Transaction	Tier
05/05/19 10:22:07 PM	675	/mo...	2 - Transfer Money	mtz_UI
05/05/19 10:14:38 PM	286	/mo...	2 - Transfer Money	mtz_UI
05/05/19 10:14:38 PM	66	/mo...	2 - Transfer Money	mtz_UI
05/05/19 10:05:39 PM	44	/mo...	2 - Transfer Money	mtz_UI
05/05/19 10:05:38 PM	351	/mo...	2 - Transfer Money	mtz_UI
05/05/19 9:50:48 PM	85	/mo...	2 - Transfer Money	mtz_UI
05/05/19 9:40:46 PM	1	/mo...	2 - Transfer Money	mtz_UI
05/05/19 9:34:56 PM	352	/mo...	2 - Transfer Money	mtz_UI
05/05/19 9:31:38 PM	313	/mo...	2 - Transfer Money	mtz_UI



Transaction snapshots can contain full or partial call graphs. All call graphs list the methods in a call stack and provide information about each call.

A full call graph lists the methods in a call stack and provides information about each call.

A partial call graph represents the subset of the call sequence, typically from the point at which the transaction has been determined to be slow or have errors.

Both periodic transaction snapshots and those snapshots captured as part of a diagnostic session will contain full call graphs.

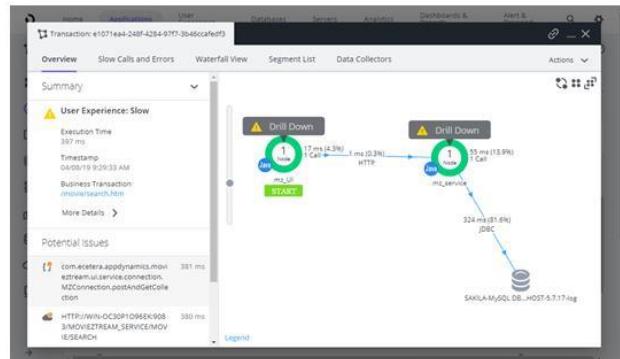
All slow, very slow, and stalled transactions will have either a partial or full call graph in the snapshot.

Error transactions may or may not have call graph information but will contain exception information. That's why you sometimes won't see the icon next to them.

Transaction Snapshot Details

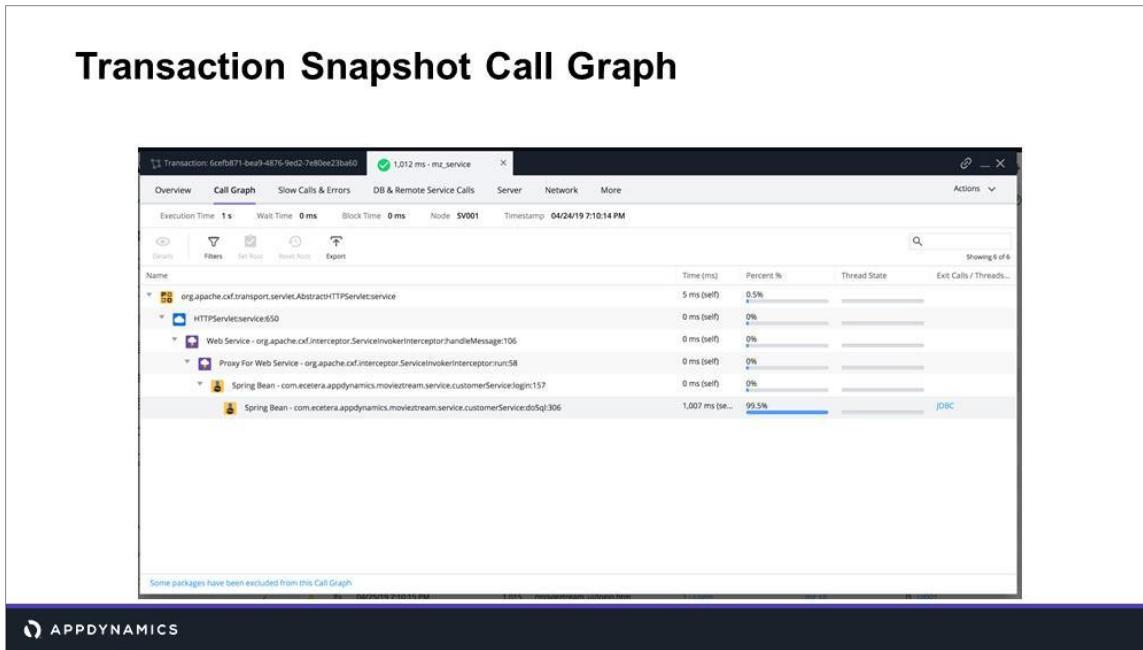
Details depicts a set of diagnostic data, including:

- An overview, with summary information about what was going on with that transaction at the time
- Additional tabs for slow calls and errors, a waterfall and segment list view, and any data collector data



A transaction snapshot depicts a set of diagnostic data, for an individual transaction across all app servers though which the transaction has passed. It is often the vehicle for troubleshooting the root causes of performance problems.

Call drill downs, where available, detail key information including slowest methods, errors, and remote service calls for the transaction execution on a tier. A drill down may include a partial or complete call graph. Call graphs reflect the code-level view of the processing of the business transaction on a particular tier.



This is the call graph view for this transaction and tier.

ATD - Automated Transaction Diagnostics

- Uses machine learning to identify snapshots with anomalies that could help identify future production issues
- Available on individual Business Transaction dashboard
- SaaS only feature
- Click to drill into details



Events

Transaction Scorecard

Category	Count	Percentage	Total
Normal	35.3k	89.4 %	
Slow	664	1.7 %	
Very Slow	3.5k	8.9 %	
Stall	< 1	0.0 %	
Errors	6	0.0 %	

Transaction Diagnostics

2 performance issues identified

⚠ Warning (05/01/21, 2:11 AM)
Duration: 6m

⚠ Warning (04/30/21, 1:10 AM)
Duration: 12m

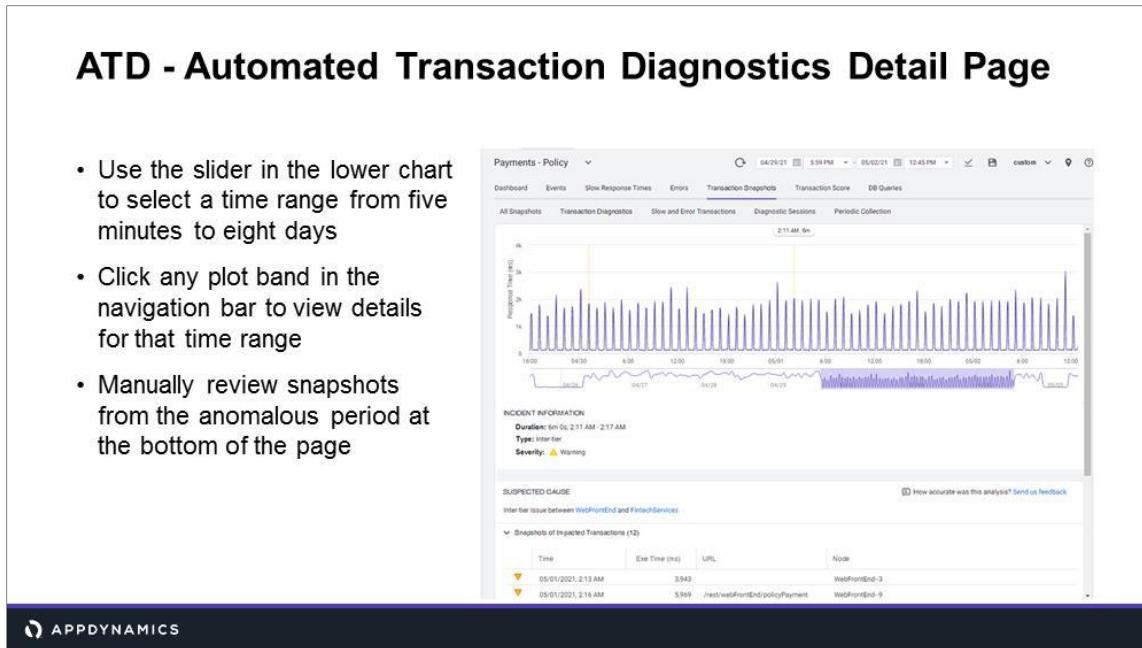
APPDYNAMICS

Automated Transaction Diagnostics uses machine learning to identify transaction snapshots for anomalies that could cause issues in the future. These snapshots may not have triggered a Health Alert.

ATD delivers a prioritized list of causal factor transactions that you can use to manage problematic business transactions before they become actual production issues or outages. ATD's guided root-cause analysis clearly exposes the offending anomalies along with the contributing tiers, exit calls, or inter-tier network issues.

ATD - Automated Transaction Diagnostics Detail Page

- Use the slider in the lower chart to select a time range from five minutes to eight days
- Click any plot band in the navigation bar to view details for that time range
- Manually review snapshots from the anomalous period at the bottom of the page



Payments - Policy

04/29/21 5:59 PM - 05/02/21 12:45 PM

Dashboard Events Slow Response Times Errors Transaction Snapshots Transaction Score DB Queries

All Snapshots Transaction Diagnostics Slow and Error Transactions Diagnostic Sessions Periodic Collection

INCIDENT INFORMATION

Duration: 05:21.1 AM - 217 AM

Type: Inter-tier

Severity: Warning

SUSPECTED CAUSE

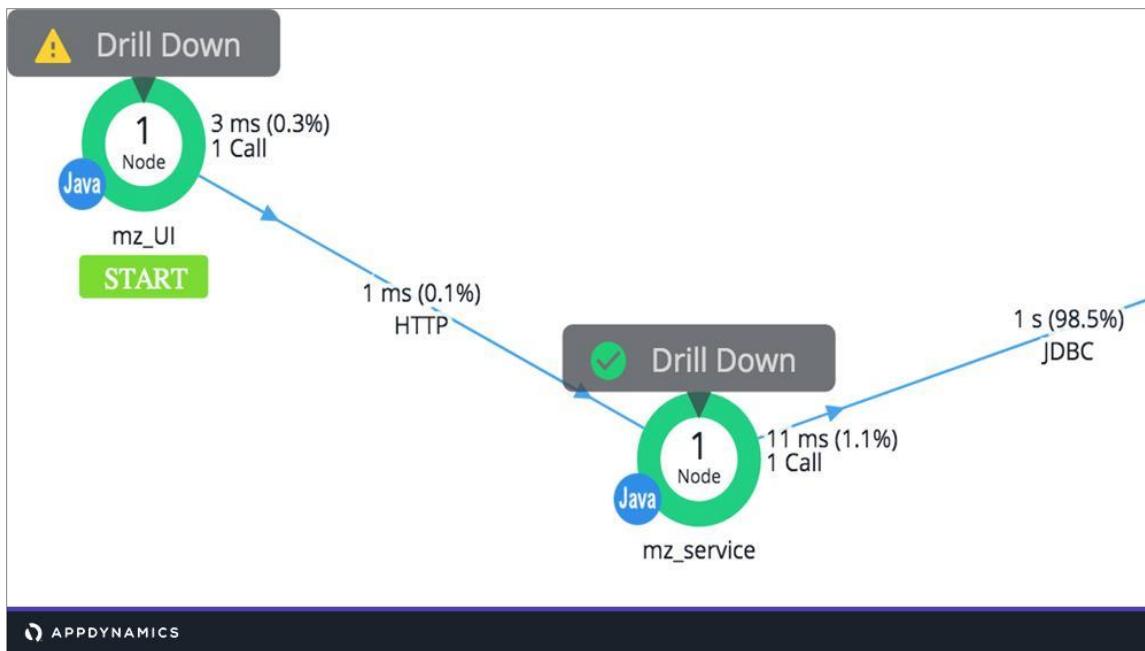
Inter-tier issue between WebFrontEnd and FinchServices

How accurate was this analysis? Send us feedback

Snapshots of Impacted Transactions (12)

Time	Exec Time (ms)	URL	Node
05/01/2021, 2:13 AM	3,943	/rest/webfrontEnd/policyPayment	WebFrontEnd-3
05/01/2021, 2:16 AM	5,969	/rest/webfrontEnd/policyPayment	WebFrontEnd-9

APPDYNAMICS



Review Question - Answer

In this transaction flow map:

- A) How many tiers are involved? **2**
- B) Where is the majority of the time spent? ***JDBC connection***
- C) Where would you focus your efforts to speed up this transaction? ***DB***

What You Learned

- The value of APM
- The concepts behind AppDynamics
- The fundamental components of the AppDynamics architecture
- How to use AppDynamics to evaluate the health of your application
- How to identify Business Transactions in your organization
- What a Transaction Snapshot is

Differentiate yourself and your company with AppDynamics Certification

APPDYNAMICS | Certification Program

General Certification Information

<https://learn.appdynamics.com/certifications>

AppDynamics Certified Associate Performance Analyst

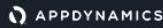
<https://learn.appdynamics.com/certifications/performance-analyst>

AppDynamics Certified Associate Administrator

<https://learn.appdynamics.com/certifications/administrator>

AppDynamics Certified Implementation Professional

<https://learn.appdynamics.com/certifications/implmenter>



To differentiate yourself and your company in an increasingly competitive market, please consider becoming AppDynamics certified.

For more information, please copy and paste this URL into a separate tab in your browser: <https://learn.appdynamics.com/certifications>

For information on specific certification tracks, please copy and paste the appropriate URLs below:

AppDynamics Certified Associate Performance Analyst:

<https://learn.appdynamics.com/certifications/performance-analyst>

AppDynamics Certified Associate Administrator:

<https://learn.appdynamics.com/certifications/administrator>

AppDynamics Certified Implementation Professional:

<https://learn.appdynamics.com/certifications/implmenter>

Troubleshooting Basics (APM212)



University

APM212 - Troubleshooting Basics

Core APM I: Essentials - Module 2

Objectives

Course

After completing this course, you will be able to:

- Explain what a diagnostic session is and describe the different ways you can run a diagnostic session
- Troubleshoot different application issues using transaction snapshots
- List the errors that AppDynamics captures

Labs

In this course's labs, you will:

- Work with diagnostic sessions
- Troubleshoot slow Business Transactions
- Troubleshoot Business Transaction errors

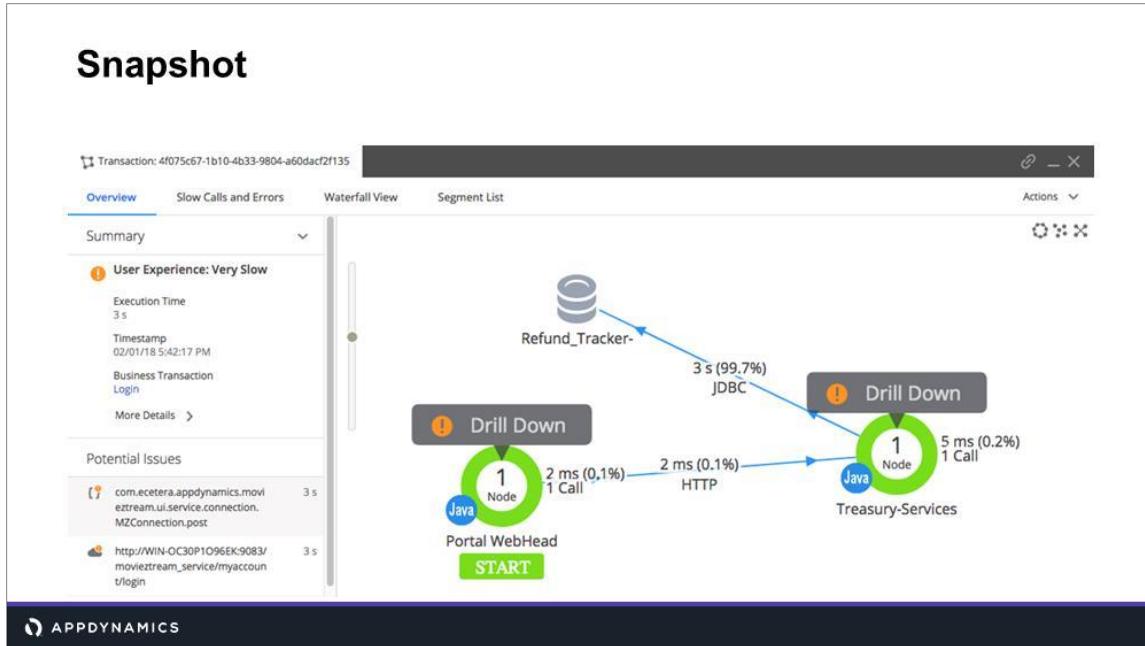
Troubleshooting Overview

The War Room

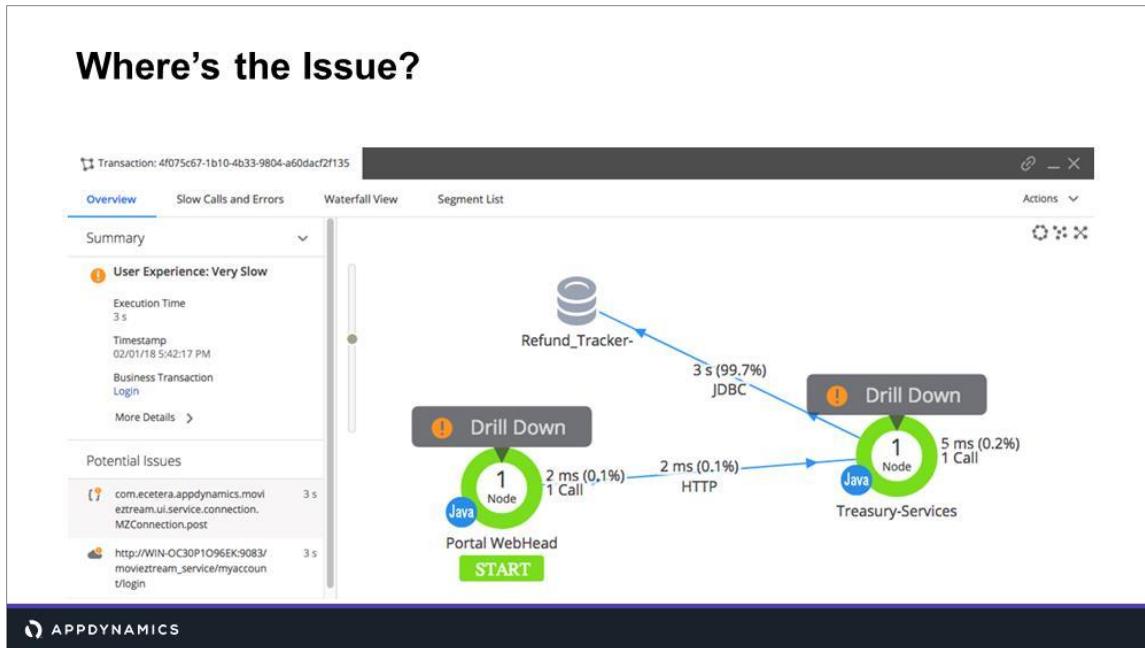


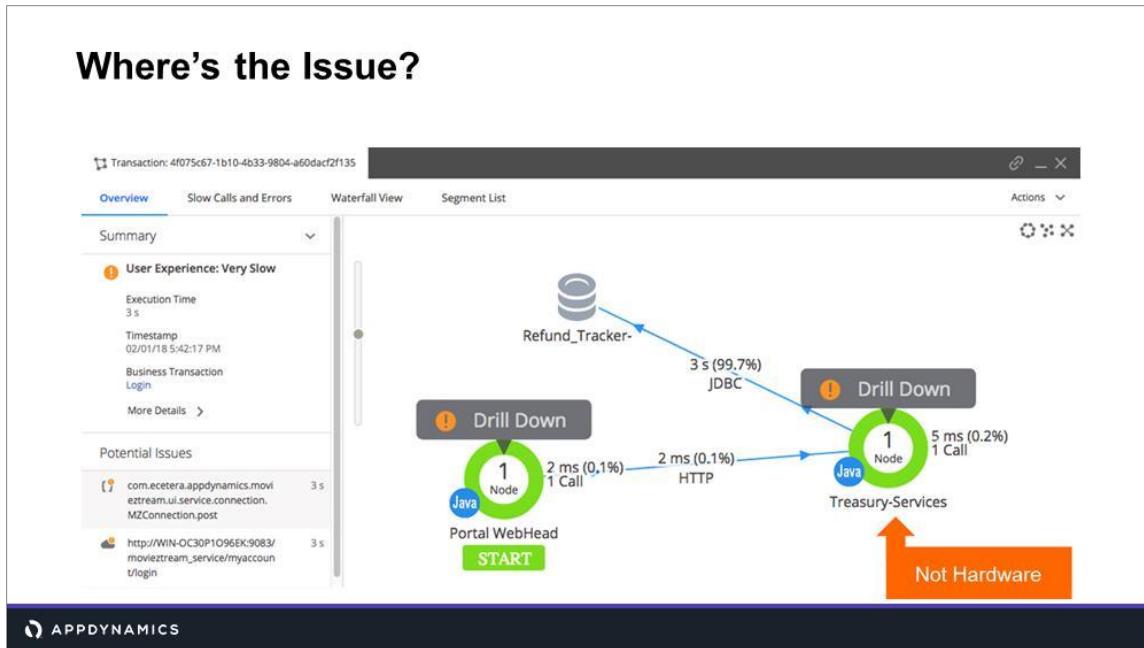
 APPDYNAMICS

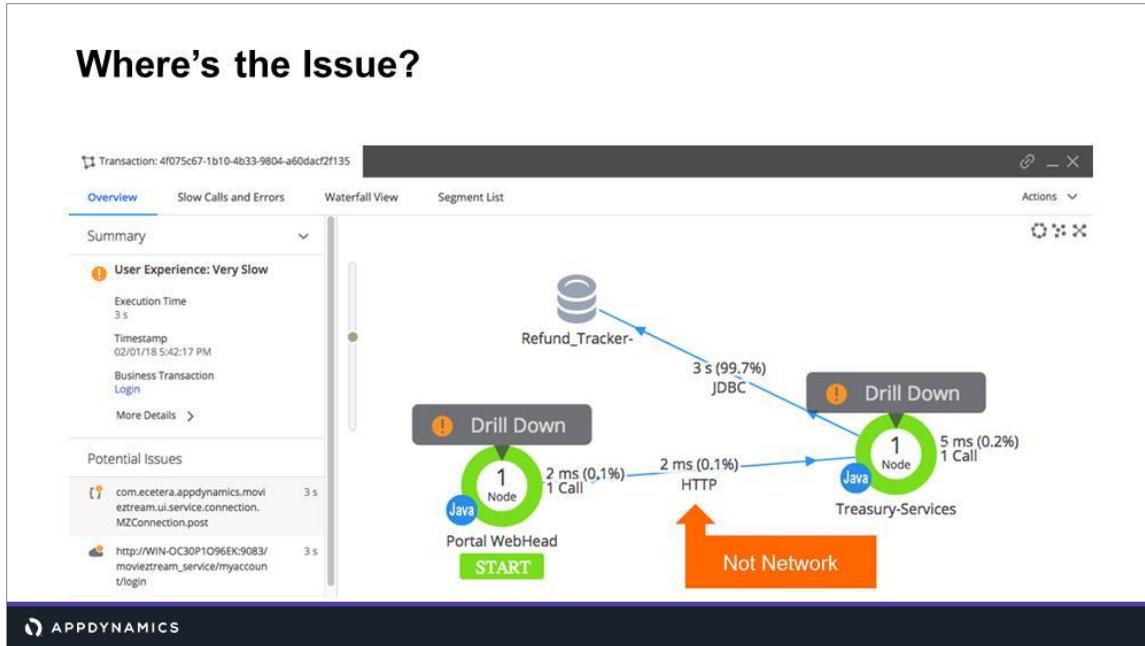
Everybody has been in one of those meetings where something has gone wrong with the application, and everyone is arguing about which component is at fault. Transaction snapshots can take the guesswork out of this, and put you on the path to fixing the issue.



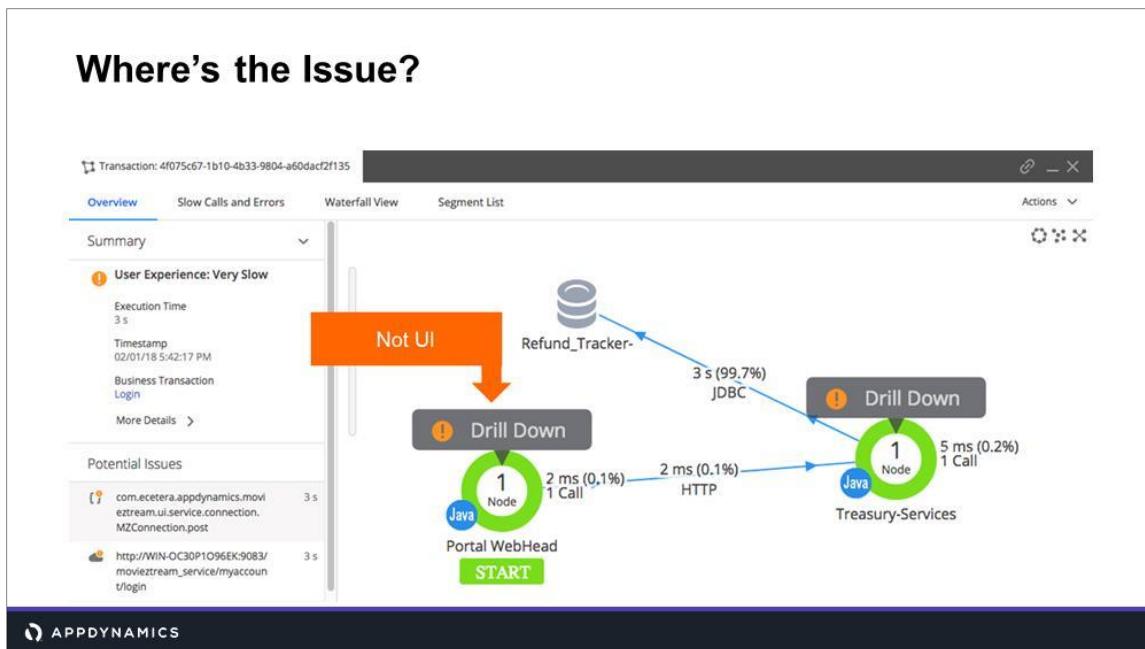
So here we're looking at a snapshot. We can see that it's a very slow snapshot. Let's see how we can find out more information.

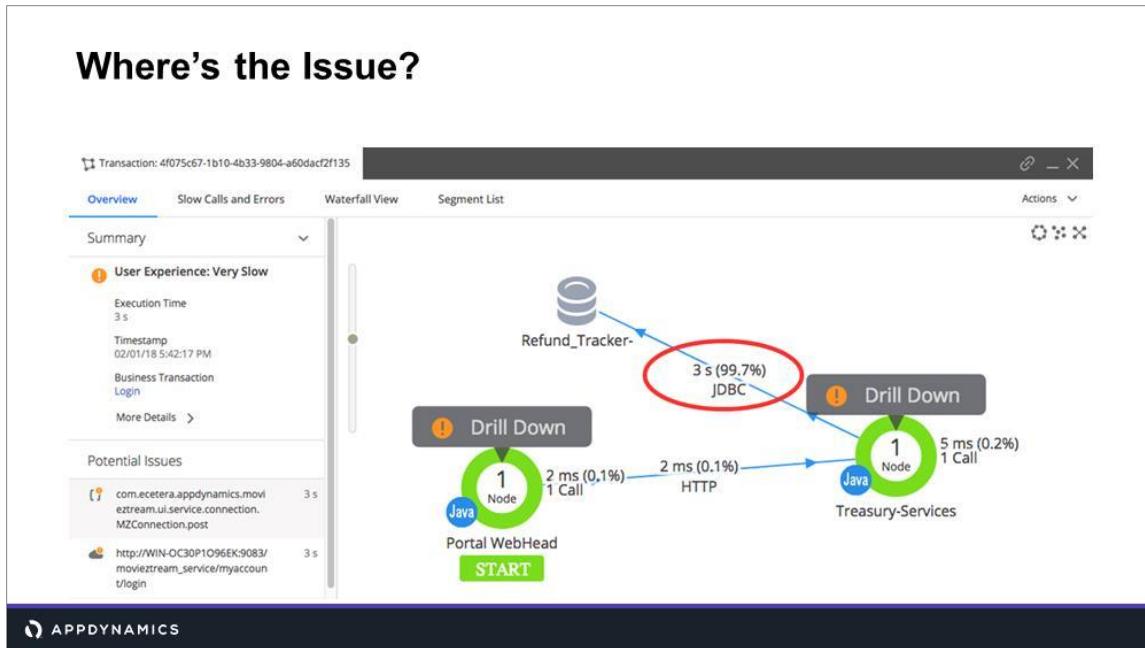






Where's the Issue?

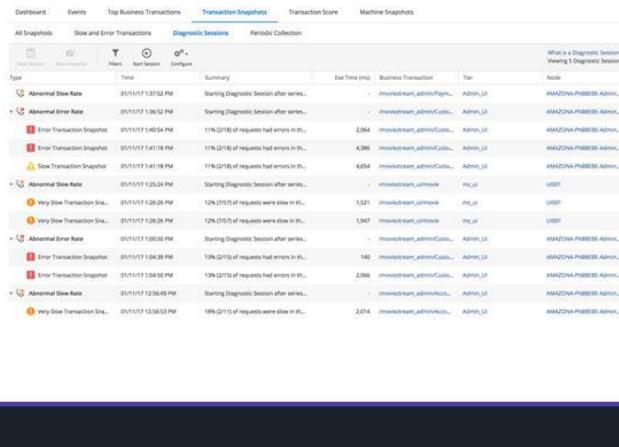




Diagnostic Sessions

Diagnostic Sessions

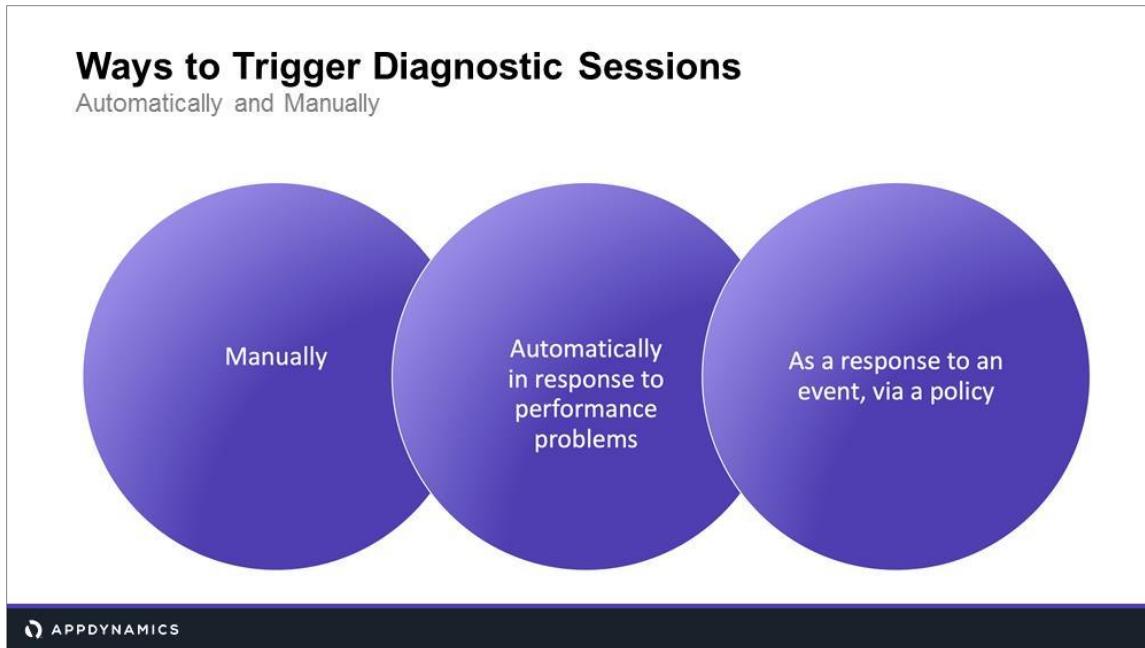
- A **Diagnostic Session** directs AppDynamics to collect extra transaction snapshots for one or more Business Transactions for a period of time
- Frequency of snapshot collection and duration of the diagnostic session are configurable
- Snapshots contain full call graphs and appear under **Transaction Snapshots > Diagnostic Sessions**



Diagnostic sessions allow you to collect extra snapshots on Business Transactions to help you troubleshoot issues within your application. These “extra” snapshots can be helpful because not only do they allow you to see more instances of a business transaction, they also provide “full call graphs” within the snapshot. A full call graph captures the entire call sequence for the business transaction invocation, and contains information on each monitored node involved in the processing of the business transaction.

Under normal circumstances, snapshots are only collected periodically or when there is a performance issue; many of these snapshots contain partial call graphs, or no call graphs at all. Snapshots with full call graphs are more likely to help diagnose performance issues, allowing you to optimize the flow of a complex Business Transaction.

The system can trigger diagnostic sessions manually through the user interface or you can configure them to start automatically when thresholds for slow, stalled, or error transactions are reached. If the diagnostic session is triggered manually, the diagnostic session collects snapshots on each node that the selected Business Transaction passes through. If the diagnostic session is triggered to start automatically, the diagnostic session collects snapshots on the triggering node only.

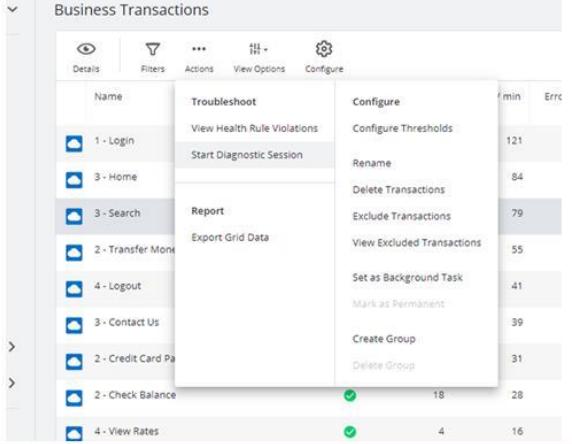


Diagnostic sessions can be started manually, or automatically, either in response to a performance issue, or as a response to an event.

Let's take a closer look at each of these.

Manual Diagnostic Session

- Triggered manually via **Business Transactions Dashboard**. Select the Business Transaction, then **Actions > Start Diagnostic Session**
- Specify duration and frequency of snapshot collection
- Collected snapshots are available at **Transaction Snapshots > Diagnostic Sessions**



Business Transactions

Name	Troubleshoot	Configure	fmin	Err
1 - Login	View Health Rule Violations	Configure Thresholds	121	
3 - Home	Start Diagnostic Session	Rename	84	
3 - Search	Report	Delete Transactions	79	
2 - Transfer Moni	Export Grid Data	Exclude Transactions	55	
4 - Logout		View Excluded Transactions	41	
3 - Contact Us		Set as Background Task	39	
2 - Credit Card Pa		Mark as Permanent	31	
2 - Check Balance		Create Group	18	28
4 - View Rates		Delete Group	4	16

You can manually trigger a diagnostic session when you need one or configure them to start up automatically.

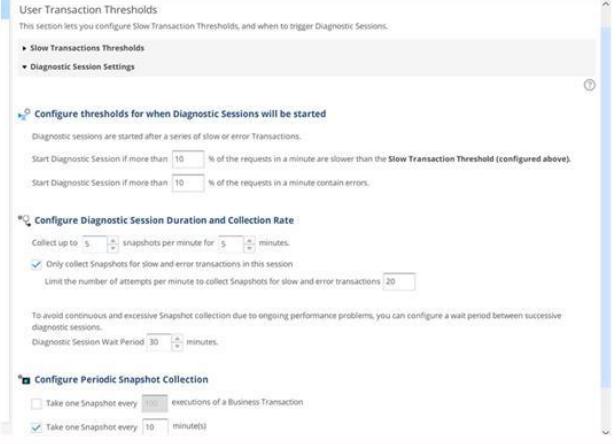
To start a diagnostic session manually, go to the Business Transaction dashboard and select the Business Transaction you want to run the session on.

Click **Actions > Start Diagnostic Session**, then specify the snapshot collection duration at the bottom of the **Start Diagnostic Session** window. The captured snapshots are available from the **Transaction Snapshots** tab > **Diagnostic Sessions**.

When configuring a manual or automatically triggered diagnostic session, the duration and frequency to take snapshots must be specified. The duration can be from 1 to 10 minutes, and the frequency can be from 1 to 10 snapshots. You should not run a diagnostic session continuously. You can set a wait period between sessions and increase the time as needed.

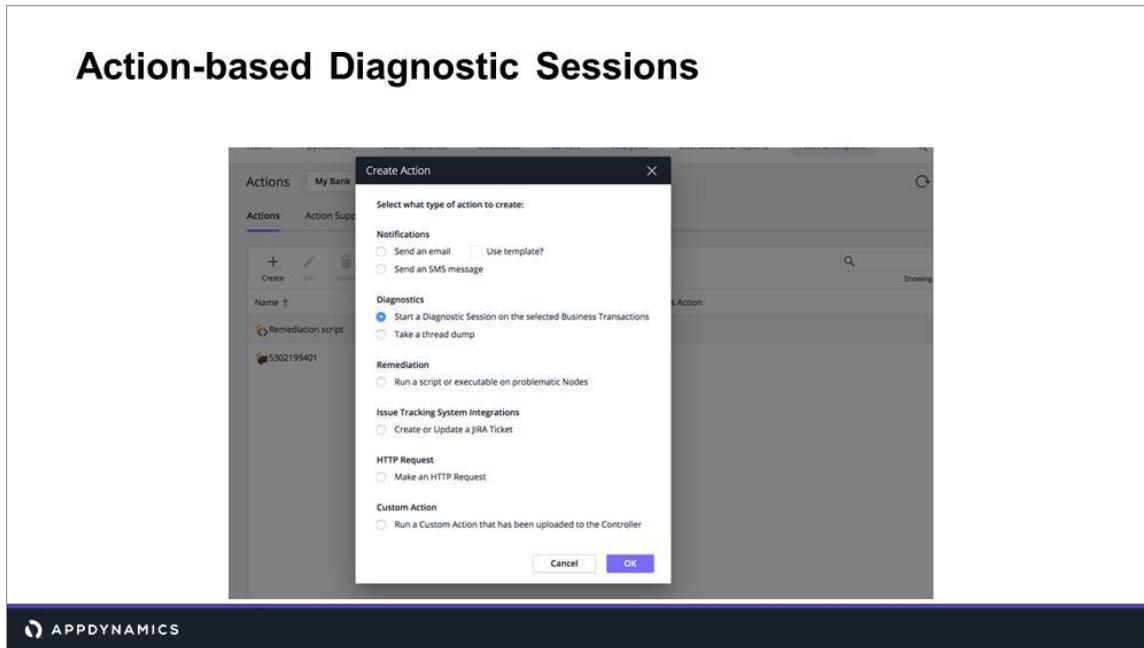
Slow/Error Transaction Diagnostic Session

- Triggered automatically when thresholds are crossed
- Thresholds configured in: **Configuration > Slow Transaction Thresholds > Diagnostic Session Settings**
- Default thresholds are**
 - More than 10% of calls in a minute have errors
 - More than 10% of calls in a minute are slow | very slow | stalled
- Result:** A diagnostic session is started for affected Business Transactions



APPDYNAMICS

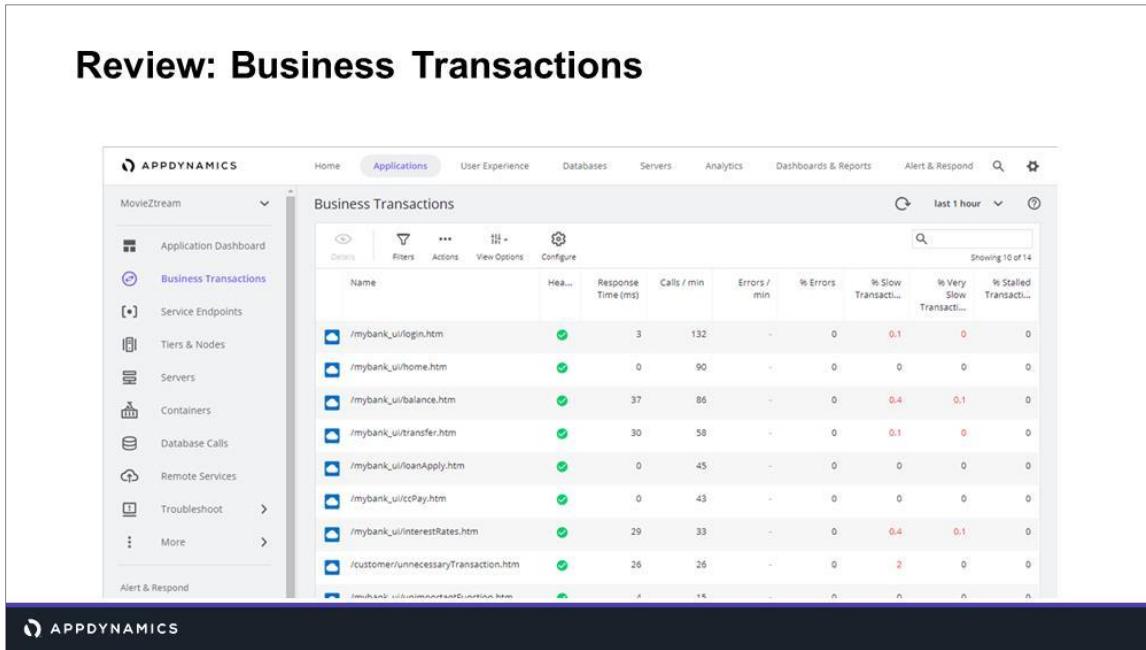
Diagnostic sessions are automatically triggered when there are a high number of slow or error transactions. For example, if more than 10% of the requests in a minute are slow or have errors, a diagnostic session begins. And like other features in AppDynamics, you can adjust the specific thresholds based on your own environment. To configure the threshold value, go to **Configuration > Slow Transaction Thresholds > Diagnostic Sessions Settings**.



Diagnostic sessions are also available as an action in AppDynamics. We'll look at policies in a later module. Policies ultimately allow you to take an event in AppDynamics and determine a response or action to that event. Running a diagnostic session is one of those available actions, and can be useful, because you're telling AppDynamics to collect full call graph snapshots when something happens, for example, a transaction breaches a performance threshold.

Troubleshooting with Transaction Snapshots

Review: Business Transactions



The screenshot shows the AppDynamics interface with the 'Business Transactions' dashboard selected. The left sidebar shows 'MovieZteam' and various monitoring options. The main table displays 14 business transactions, with the first 10 visible. The columns include Name, Headers, Response Time (ms), Calls / min, Errors / min, % Errors, % Slow Transaction, % Very Slow Transaction, and % Stalled Transaction. The transactions listed are: /mybank_ui/login.htm, /mybank_ui/home.htm, /mybank_ui/balance.htm, /mybank_ui/transfer.htm, /mybank_ui/loanApply.htm, /mybank_ui/ccPay.htm, /mybank_ui/interestRates.htm, /customer/unnecessaryTransaction.htm, and /customer/unnecessaryTransaction.htm (repeated).

Name	Headers	Response Time (ms)	Calls / min	Errors / min	% Errors	% Slow Transaction	% Very Slow Transaction	% Stalled Transaction
/mybank_ui/login.htm	✓	3	132	-	0	0.1	0	0
/mybank_ui/home.htm	✓	0	90	-	0	0	0	0
/mybank_ui/balance.htm	✓	37	86	-	0	0.4	0.1	0
/mybank_ui/transfer.htm	✓	30	58	-	0	0.1	0	0
/mybank_ui/loanApply.htm	✓	0	45	-	0	0	0	0
/mybank_ui/ccPay.htm	✓	0	43	-	0	0	0	0
/mybank_ui/interestRates.htm	✓	29	33	-	0	0.4	0.1	0
/customer/unnecessaryTransaction.htm	✓	26	26	-	0	2	0	0
/customer/unnecessaryTransaction.htm	✓	4	14	-	0	0	0	0

Remember that a Business Transaction represents a distinct logical user activity (login, search, checkout, etc.).

Business Transactions allow you to examine the performance of your application as your users experience it, and they are at the root of most monitoring and troubleshooting activities.

Review: Transaction Snapshots

Transaction Snapshots include the user experience

! Error

! Very Slow

! Slow

✓ Healthy

✗ Stalled

📄 Indicates a full call graph

📄 Indicates a partial call graph

Time	Exec Time (ms)	URL	Business Transaction	Tier
05/05/19 10:22:07 PM	675	/m...	2 - Transfer Money	mz_UI
05/05/19 10:14:38 PM	286	/m...	2 - Transfer Money	mz_UI
05/05/19 10:13:59 PM	66	/m...	2 - Transfer Money	mz_UI
05/05/19 10:05:39 PM	44	/m...	2 - Transfer Money	mz_UI
05/05/19 10:05:38 PM	351	/m...	2 - Transfer Money	mz_UI
05/05/19 9:50:48 PM	85	/m...	2 - Transfer Money	mz_UI
05/05/19 9:40:46 PM	1	/m...	2 - Transfer Money	mz_UI
05/05/19 9:34:56 PM	352	/m...	2 - Transfer Money	mz_UI
05/05/19 9:31:38 PM	313	/m...	2 - Transfer Money	mz_UI

⌚ APPDYNAMICS

And transaction snapshots give you a cross-tier view of the processing flow for a single invocation of a transaction.

Transaction snapshots can contain full or partial call graphs. All call graph list the methods in a call stack and provides information about each call.

A partial call graph represents the subset of the call sequence, typically from the point at which the transaction has been determined to be slow or have errors.

Both periodic transaction snapshots and those snapshots captured as part of a diagnostic session will contain full call graphs.

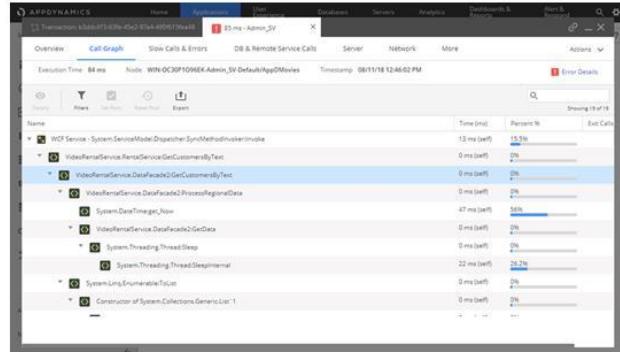
All slow/very slow/stalled transactions will have either a partial or full call graph in the snapshot.

Error transactions may or may not have any call graph information but will contain exception information. That's why you sometimes won't see the icon next to them.

Review: Transaction Snapshot Drill Down

The drill-down button opens the code level details for this transaction on the particular tier.

By default, drill-down opens to the Call Graph tab.



Drilling down on a tier using the **Drill Down** button on the **Overview** tab flow map on the **Transaction Details** screen opens the code-level details for the Business Transaction execution on that particular tier. This is how you access the call graph information.

- The **Overview** tab includes a problem summary, execution times, CPU time stamp, tier, node, process ID, thread name, etc.
- The **Call Graphs** tab lists call graphs showing the execution flow for the transaction on a given tier. Full snapshots have complete call graphs. Partial snapshots can have partial call graphs or none at all.
- The **Slow Calls and Errors** tab lists all the slow method calls and calls that resulted in an error. You can use the **Hot Spots** slider to sort calls by execution time with the most expensive calls in the snapshot at the top.
- The **Error Details** tab exposes exception stack traces and HTTP error codes.
- The **DB & Remote Service Calls** tab shows all SQL query exit calls to databases and exit calls to other remote services such as web services, message queues or caching servers.
- The **Server** tab displays graphs for hardware (CPU Memory, Disk IO, Network IO), Memory (Heap, Garbage Collection, Memory Pools), JMX, and more. If you have Server Visibility, you'll have access to full performance details for the server hardware and operating system

- For customers using Network Visibility, the **Network** tab shows charts related to the impact of the network on the transaction and other pertinent data.
- The **More** tab shows hows metrics for the node that deviate the most from the established baselines as Node Problems. It also shows all the Service Endpoints invoked during the snapshot and the Servlet URI and Process ID of the transaction.

The Two Things Customers Complain About



It's broken!

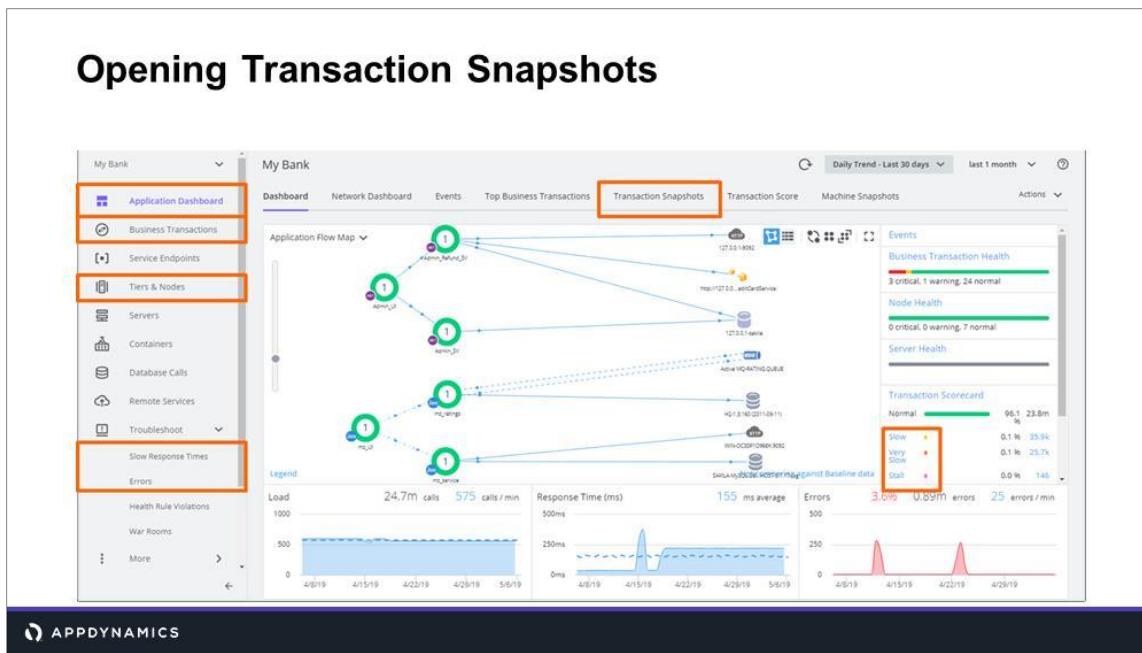


It's slow!

 APPDYNAMICS

One of the results of applications that experience slow performance or downtime is an angry customer.

And a great first place to look is Business Transactions via transaction snapshots.

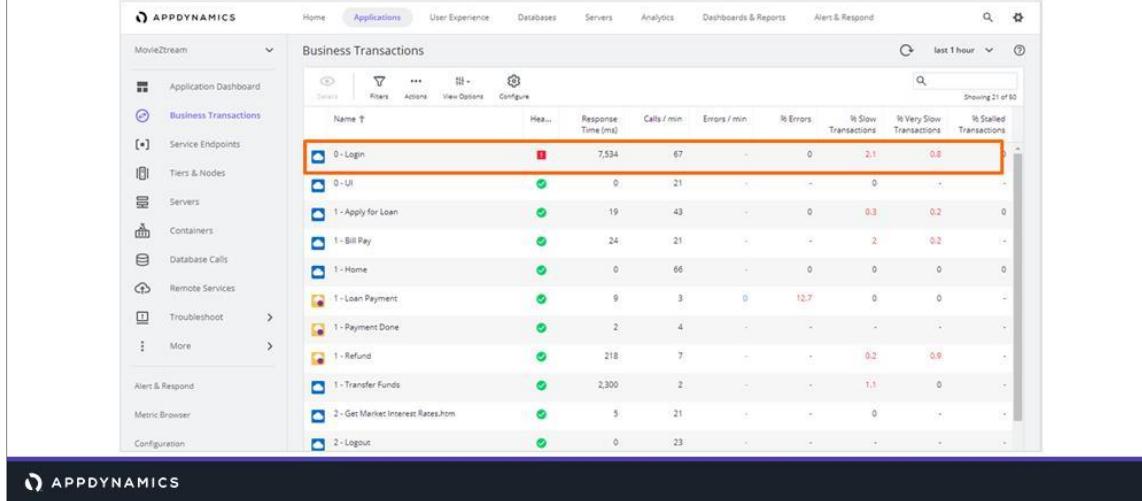


Let's walk through how you might use Business Transactions and transaction snapshots to find the root cause of an issue. Each of the highlighted areas will get you to transaction snapshots although some take a couple of clicks, such as Business Transactions and Tiers and Nodes in the left nav.

In this case, imagine that support has contacted you about users receiving login errors.

The methods for opening transaction snapshots on this page all eventually open a general transaction snapshots page, where you could then filter and search for the specific transaction - in this case, Login. But there's an even faster way to get to the transaction snapshots for this transaction.

Opening Transaction Snapshots

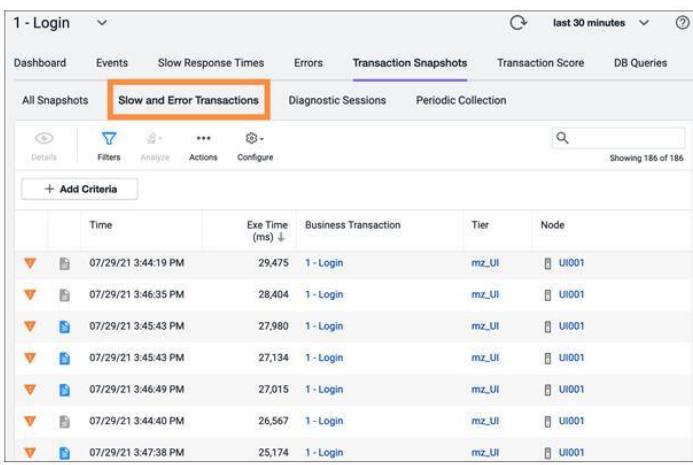


Name	Health	Response Time (ms)	Calls / min	Errors / min	% Errors	% Slow Transactions	% Very Slow Transactions	% Stalled Transactions
0 - Login	Red	7,534	67	-	0	2.1	0.8	-
0 - UI	Green	0	21	-	-	0	-	-
1 - Apply for Loan	Green	19	43	-	0	0.3	0.2	0
1 - Bill Pay	Green	24	21	-	-	2	0.2	-
1 - Home	Green	0	66	-	0	0	0	0
1 - Loan Payment	Green	9	3	0	12.7	0	0	-
1 - Payment Done	Green	2	4	-	-	-	-	-
1 - Refund	Green	218	7	-	-	0.2	0.9	-
1 - Transfer Funds	Green	2,300	2	-	-	1.1	0	-
2 - Get Market Interest Rates.htm	Green	5	21	-	-	0	-	-
2 - Logout	Green	0	23	-	-	-	-	-

You can click **Business Transactions** on the left menu to see a list of transactions and then drill down from there.

Here's the **Business Transactions** screen, and we can see that something's going on with the Login transaction. So let's open that up by double-clicking it.

Finding the Right Snapshot(s)

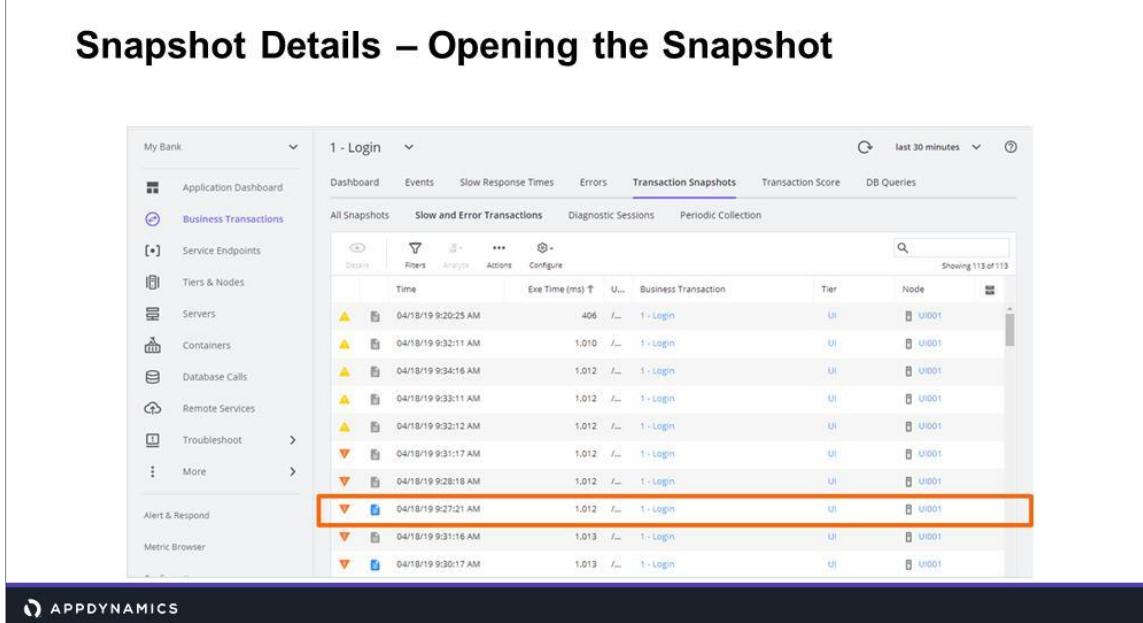


	Time	Exe Time (ms)	Business Transaction	Tier	Node
▼	07/29/21 3:44:19 PM	29,475	1 - Login	mz_Ui	UI001
▼	07/29/21 3:46:35 PM	28,404	1 - Login	mz_Ui	UI001
▼	07/29/21 3:45:43 PM	27,980	1 - Login	mz_Ui	UI001
▼	07/29/21 3:45:43 PM	27,134	1 - Login	mz_Ui	UI001
▼	07/29/21 3:46:49 PM	27,015	1 - Login	mz_Ui	UI001
▼	07/29/21 3:44:40 PM	26,567	1 - Login	mz_Ui	UI001
▼	07/29/21 3:47:38 PM	25,174	1 - Login	mz_Ui	UI001

Now we're looking at just the snapshots for Login (**Business Transactions > Login**). Here we could filter to only show those that are slow, very slow, or error transactions.

Or, we could click the **Slow and Error Transactions** subtab to start with those snapshots with issues. Let's do that.

Snapshot Details – Opening the Snapshot

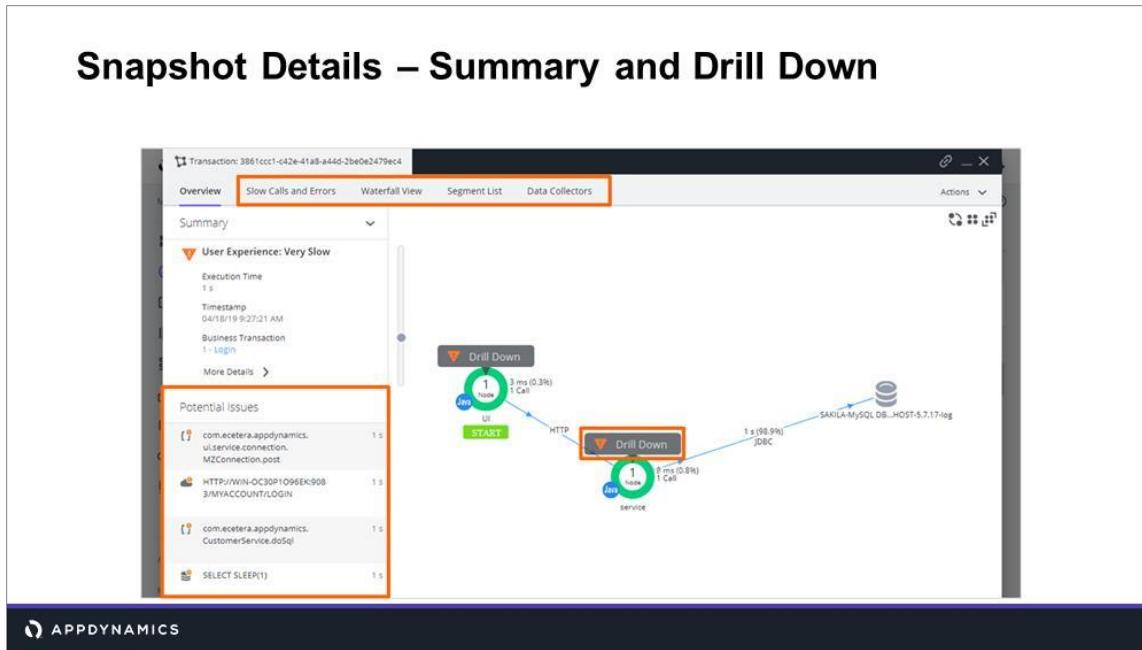


The screenshot shows the AppDynamics interface for 'My Bank'. The left sidebar includes 'Application Dashboard', 'Business Transactions' (selected), 'Service Endpoints', 'Tiers & Nodes', 'Servers', 'Containers', 'Database Calls', 'Remote Services', 'Troubleshoot' (with a dropdown arrow), and 'More'. The main content area is titled '1 - Login' and shows tabs for 'Dashboard', 'Events', 'Slow Response Times', 'Errors', 'Transaction Snapshots' (selected), 'Transaction Score', and 'DB Queries'. The 'Slow and Error Transactions' subtab is active. A table lists transactions with columns: Time, Exec Time (ms) (sorted by desc), User, Business Transaction, Tier, and Node. One row, '04/18/19 9:27:21 AM' with '1.012 ms' and '1 - Login' under 'Business Transaction', is highlighted with a red box. The table shows 113 of 113 rows.

On the **Slow and Error Transactions** subtab, you can use search and filter to further narrow down to the specific Business Transaction.

We've got a very slow login transaction right here which has a full call graph, so let's look at that one.

Double-clicking the snapshot opens the snapshot details.



The transaction snapshot details page offers you additional information on troubleshooting. Potential issues are listed on the left.

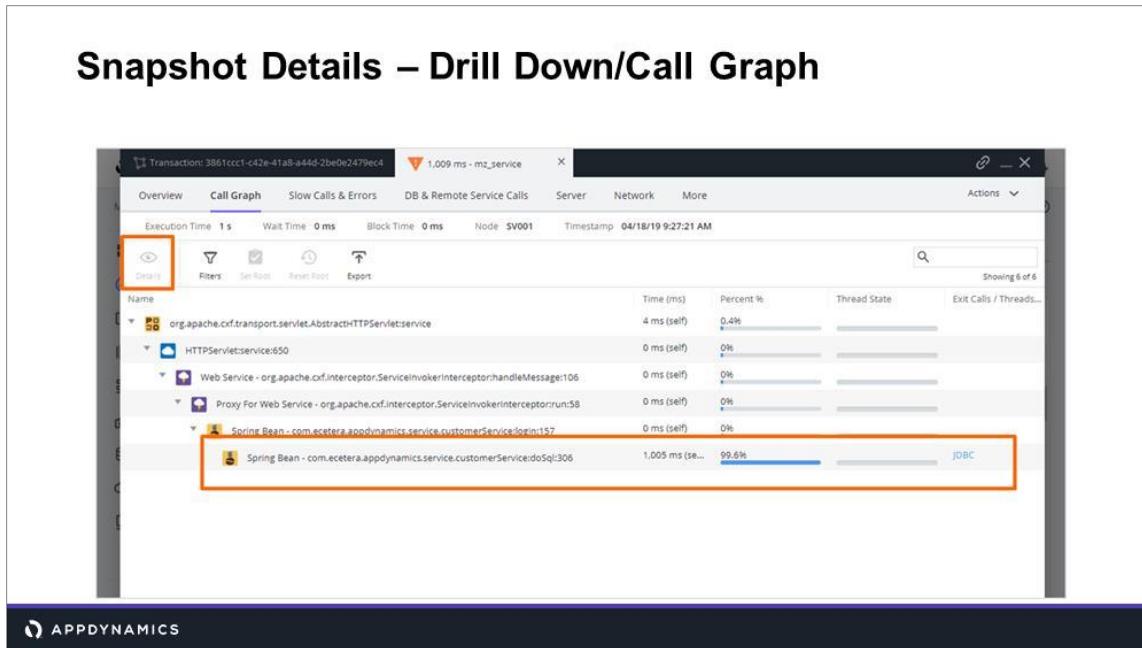
The **Slow Calls and Errors** tab allows you to focus only on slow and very slow calls, and errors.

The **Waterfall View** tab shows the execution of an individual **Business Transaction** broken into execution segments.

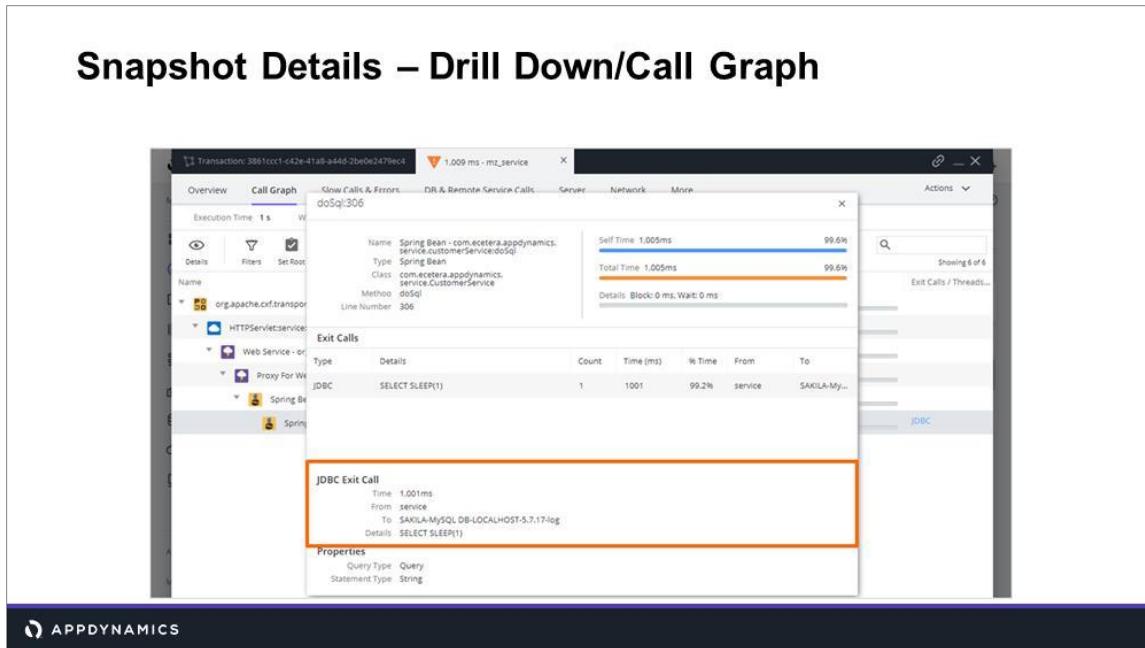
Segment List shows the various legs of the transaction in descending order of duration and give access to their snapshots and allows you to drill down into their details.

And the **Drill Down** links over the tiers in the flow map allow you to drill down into the details of what was going on with that transaction in the tier.

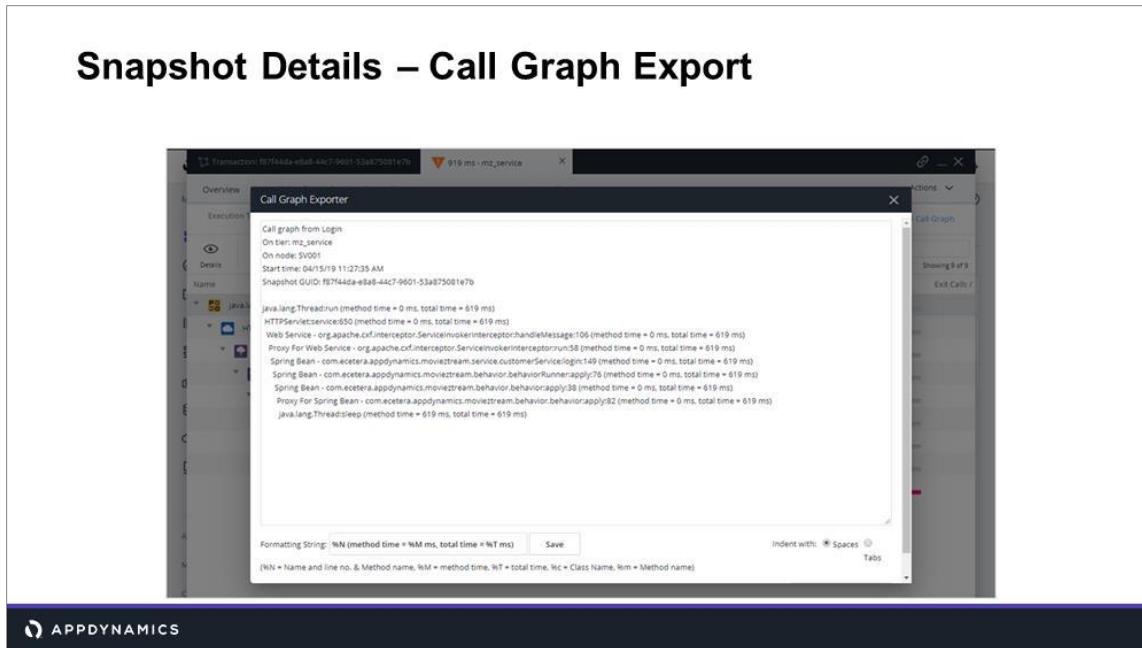
As the communication between the service tier and the database seems to be very slow, let's drill down there.



The call graph screen lets you see where the most time was spent in the transaction. You can see that the last call is taking up most of the time. So let's look at the details of that call. You can view the details by selecting the call and clicking the details button on the toolbar.



In this case, it looks like a sleep call was causing the slowdown on the login transaction.



To get the information out of AppDynamics so that you can send it to the appropriate audience, you can either export the call graph information using the **Export** button on the toolbar, or you can right-click the call and select **Copy to Clipboard**. You can then paste the info into an email or document.

Snapshot Details – Export

Transactions: c7b01a84-a9cd-47ff 92a8-a15c27fe50cf 1,093 ms - mz_service

Actions

Slowest Calls and Errors

Slowest Method: 48 ms
java.lang.Throwable#run

Slowest SQL Call: 44 ms

```
SELECT THIS.RENTAL_ID AS RENTAL1_23_A, THIS.CUSTOMER_ID AS CUSTOMER2_23_A,
THIS.INVENTORY_ID AS INVENTORY3_23_A, THIS.LAST_UPDATE AS LAST4_23_A,
THIS.RENTAL_DATE AS RENTALS_23_A, THIS.RETURN_DATE AS RETURNs_23_A,
THIS.STAFF_ID AS STAFF7_23_A, CUSTOMER2.CUSTOMER_ID AS CUSTOMER1_5_0,
CUSTOMER2.ACTIVE AS ACTIVES_0, CUSTOMER2.ADDRESS_ID AS ADDRESS5_5_0,
CUSTOMER2.CREATE_DATE AS CREATE4_5_0, CUSTOMER2.EMAIL AS EMAIL5_0,
CUSTOMER2.FIRST_NAME AS FIRST6_5_0, CUSTOMER2.LAST_NAME AS LAST7_5_0,
CUSTOMER2.LAST_UPDATE AS LAST8_5_0, CUSTOMER2.STORE_ID AS STORE9_5_0,
INVENTORY3.INVENTORY_ID AS INVENTORY1_22_1, INVENTORY3.FILM_ID AS FILM2_22_1,
INVENTORY3.LAST_UPDATE AS LAST3_22_1, INVENTORY3.STORE_ID AS STORE4_22_1,
FILM4.FILM_ID AS FILM1_20_2, FILM4.DESCRIPTION AS DESCRIPT2_20_2,
FILM4.LANGUAGE_ID AS LANGUAGES_20_2, FILM4.LAST_UPDATE AS LAST4_20_2,
FILM4.LENGTH AS LENGTH20_2, FILM4.ORIGINAL_LANGUAGE_ID AS ORIGINAL6_20_2,
FILM4.RATING AS RATING20_2, FILM4.RELEASE_YEAR AS RELEASEB_20_2,
FILM4.RENTAL_DURATION AS RENTAL9_20_2, FILM4.RENTAL_RATE AS RENTAL10_20_2,
FILM4.REPLACEMENT_COST AS REPLACE11_20_2, FILM4.SPECIAL_FEATURES AS
SPECIAL12_20_2, FILM4.TITLE AS TITLE20_2, PAYMENTRENTS.RENTAL_ID AS RENTALS_23_6,
PAYMENTRENTs.PAYMENT_ID AS PAYMENT1_6_6, PAYMENTRENTs.PAYMENT_ID AS
PAYMENT11_6_3, PAYMENTRENTs.AMOUNT AS AMOUNT10_3,
PAYMENTRENTs.CUSTOMER_ID AS CUSTOMERS3_10_3, PAYMENTRENTs.LAST_UPDATE AS
```

View Node Dashboard during this Snapshot

Search Logs By Host or IP Address

Export

APPDYNAMICS

You can export the snapshot details at the overview level, or on any of the other tabs here by clicking the **Actions** drop down and selecting **Export**.

The chain let's you send a link to the snapshot details. If you want the link to be available for longer than the standard two weeks, archive the snapshot.

Review Question - Answer

In this snapshots screen:

- A) How many snapshots are full snapshots with a complete call graph? 3
- B) How many snapshots have errors? 4
- C) Which Business Transaction is showing the most error snapshots? [billPay/Payments](#)

Review Question - Answer

You can access slow transactions by:

- A) Clicking **Troubleshoot** on the left navigation then **Slow Response Times**.
- B) Clicking **Application Dashboard** then **Transaction Snapshots** then **Slow and Error Transactions**.
- C) Clicking **Application Dashboard** then **Slow Transactions** from the **Transaction Scorecard**.
- D) Clicking **Containers** in the left menu.

Troubleshooting Slow Transactions

Troubleshooting Slow and Very Slow Transactions

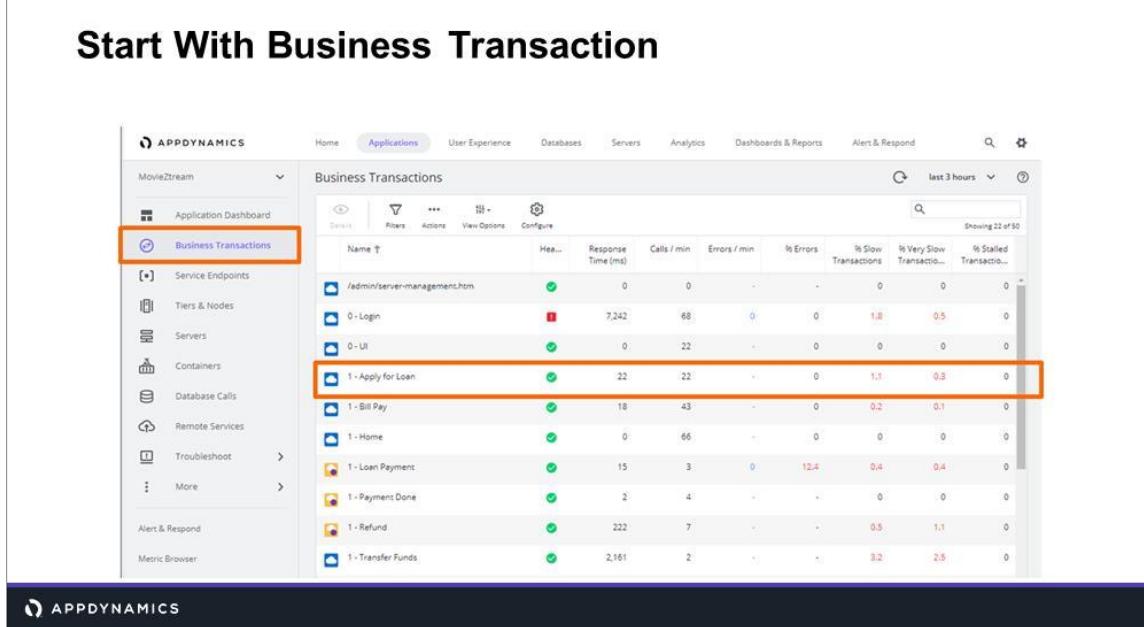


**Submitting my online
loan application is
taking forever!!**

 APPDYNAMICS

It looks like loan applications are starting to be very slow. Loan applications make up a large portion of our bank's business so we need to figure out what's going on.

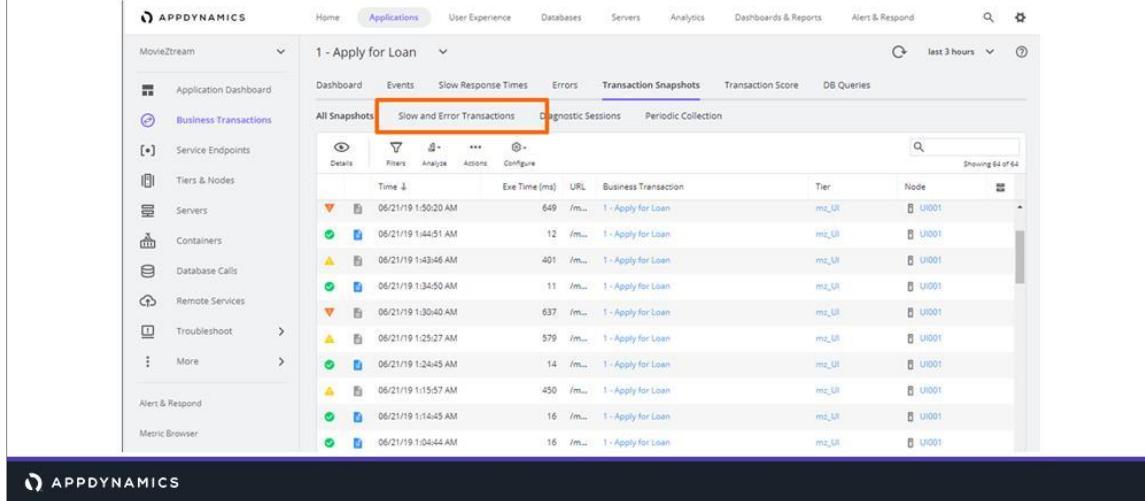
Start With Business Transaction



Name	Health	Response Time (ms)	Calls / min	Errors / min	% Errors	% Slow Transactions	% Very Slow Transactions	% Stalled Transactions
/admin/server-management.htm	Green	0	0	-	-	0	0	0
0 - Login	Red	7,242	68	0	0	1.6	0.5	0
0 - UI	Green	0	22	-	0	0	0	0
1 - Apply for Loan	Green	22	22	-	0	1.1	0.3	0
1 - Bill Pay	Green	18	43	-	0	0.2	0.1	0
1 - Home	Green	0	66	-	0	0	0	0
1 - Loan Payment	Green	15	3	0	12.4	0.4	0.4	0
1 - Payment Done	Green	2	4	-	-	0	0	0
1 - Refund	Green	222	7	-	-	0.5	1.1	0
1 - Transfer Funds	Green	2,161	2	-	-	3.2	2.5	0

When troubleshooting slow and very slow transactions, start by clicking **Business Transactions** in the left menu. Even though there's a green check-mark, more calls are starting to come in. So let's open that transaction by double-clicking it.

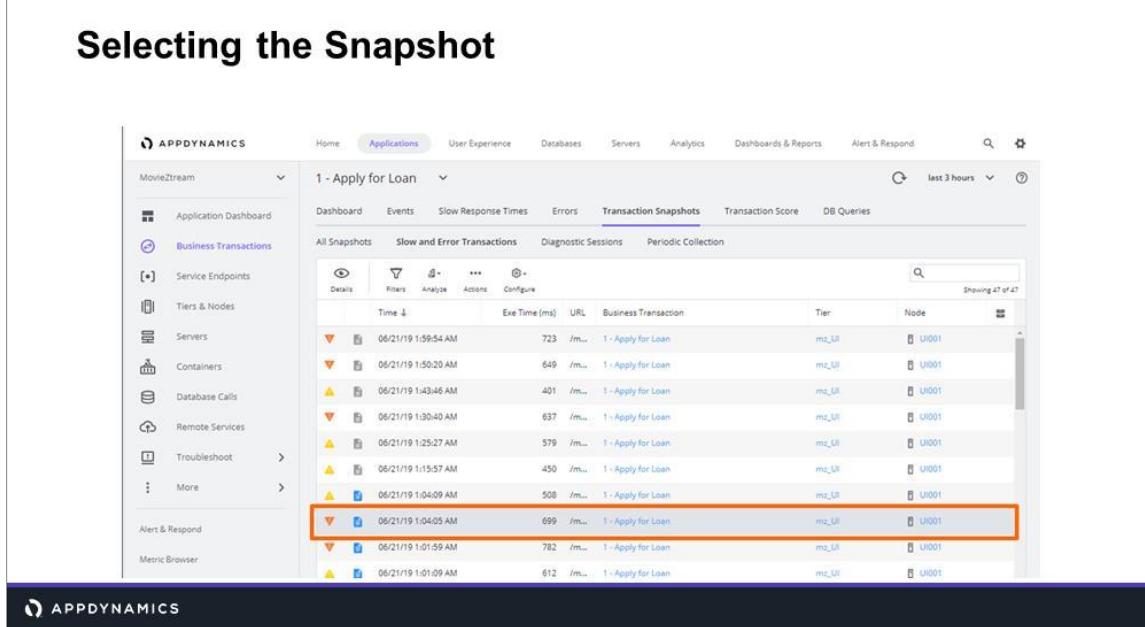
Slow and Error Transactions



Time	Event Time (ms)	URL	Business Transaction	Tier	Node
06/21/19 1:50:20 AM	649	/m...	1 - Apply for Loan	mtz_U1	UI001
06/21/19 1:45:51 AM	12	/m...	1 - Apply for Loan	mtz_U1	UI001
06/21/19 1:45:49 AM	401	/m...	1 - Apply for Loan	mtz_U1	UI001
06/21/19 1:34:50 AM	11	/m...	1 - Apply for Loan	mtz_U1	UI001
06/21/19 1:30:40 AM	637	/m...	1 - Apply for Loan	mtz_U1	UI001
06/21/19 1:25:27 AM	579	/m...	1 - Apply for Loan	mtz_U1	UI001
06/21/19 1:24:45 AM	14	/m...	1 - Apply for Loan	mtz_U1	UI001
06/21/19 1:15:57 AM	450	/m...	1 - Apply for Loan	mtz_U1	UI001
06/21/19 1:14:49 AM	16	/m...	1 - Apply for Loan	mtz_U1	UI001
06/21/19 1:04:44 AM	16	/m...	1 - Apply for Loan	mtz_U1	UI001

Now we can look specifically at the slow and error transactions for the Apply for Loan Business Transaction.

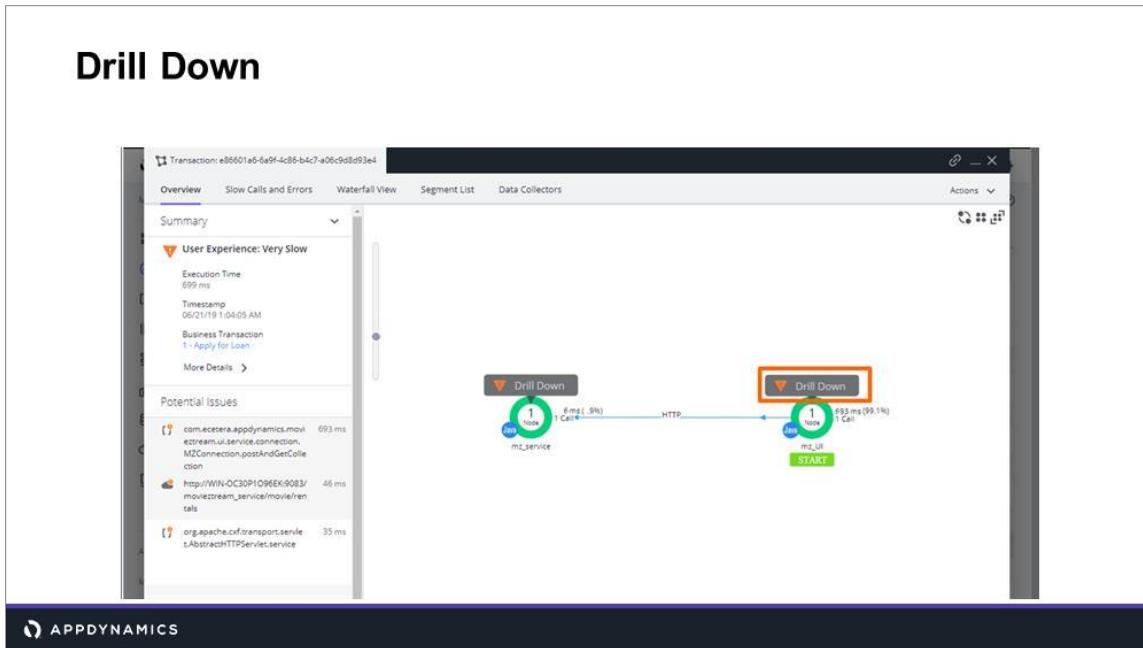
Selecting the Snapshot



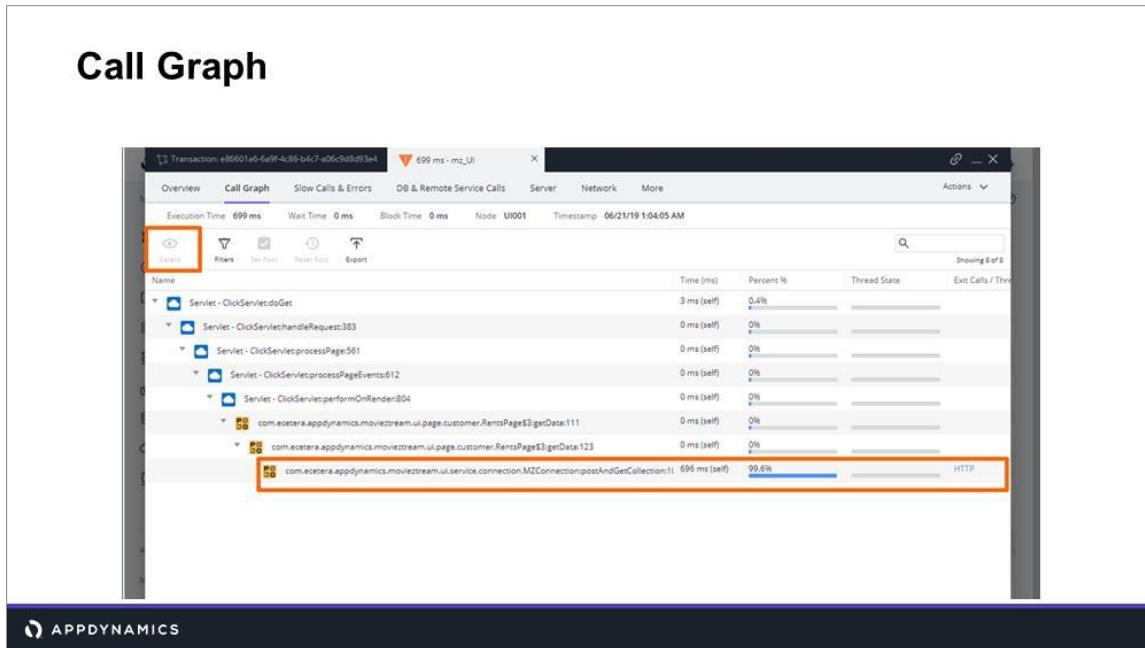
The screenshot shows the AppDynamics interface for the 'MovieZstream' application. The 'Applications' tab is selected. On the left, the navigation menu includes 'MovieZstream' (selected), 'Application Dashboard', 'Business Transactions' (selected), 'Service Endpoints', 'Tiers & Nodes', 'Servers', 'Containers', 'Database Calls', 'Remote Services', 'Troubleshoot' (selected), and 'More'. Under 'Alert & Respond', there is an 'Alerts' section. The 'Metric Browser' is also listed. The main content area shows a table of 'Transaction Snapshots' for the '1 - Apply for Loan' transaction. The table has columns: Time, Eve Time (ms), URL, Business Transaction, Tier, and Node. The table shows 12 rows of data. The 6/21/19 1:04:05 AM row is highlighted with a red box. The URL for this row is /m... and the Business Transaction is '1 - Apply for Loan'. The Tier is mz_U1 and the Node is U001.

Time	Eve Time (ms)	URL	Business Transaction	Tier	Node
06/21/19 1:59:54 AM	723	/m...	1 - Apply for Loan	mz_U1	U001
06/21/19 1:50:20 AM	649	/m...	1 - Apply for Loan	mz_U1	U001
06/21/19 1:43:46 AM	401	/m...	1 - Apply for Loan	mz_U1	U001
06/21/19 1:30:40 AM	637	/m...	1 - Apply for Loan	mz_U1	U001
06/21/19 1:25:27 AM	579	/m...	1 - Apply for Loan	mz_U1	U001
06/21/19 1:15:57 AM	450	/m...	1 - Apply for Loan	mz_U1	U001
06/21/19 1:04:09 AM	508	/m...	1 - Apply for Loan	mz_U1	U001
06/21/19 1:04:05 AM	699	/m...	1 - Apply for Loan	mz_U1	U001
06/21/19 1:01:59 AM	782	/m...	1 - Apply for Loan	mz_U1	U001
06/21/19 1:01:09 AM	612	/m...	1 - Apply for Loan	mz_U1	U001

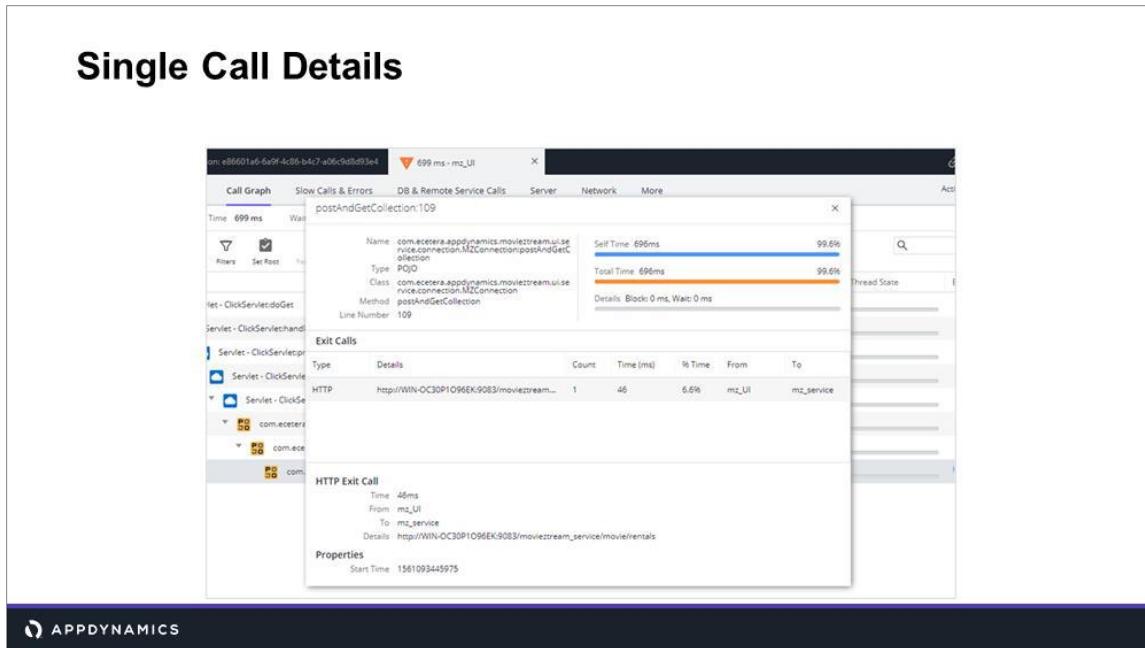
And here's a very slow transaction with a full snapshot. So let's open this one.



The transaction details opens to the **Overview** tab, and we can see the details of this transaction. On the flow map, it looks like both tiers have issues, but let's start with the drill down link on the UI tier.



Here we can see the entire call graph, and easily see where the time was spent. To see more details about the slow call, click to select it, and click the **Details** button.



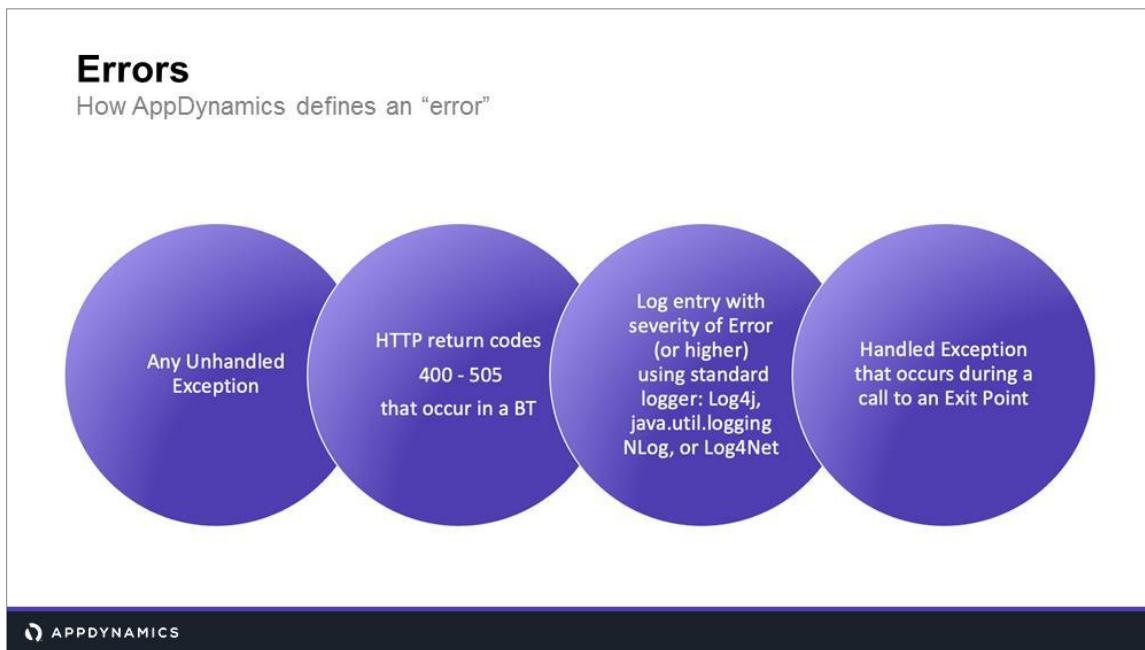
And here are the details for that call.

Review Question - Answer

If you're looking for slow snapshots **for a specific BT**, you could do which of the following:

- A) Click Troubleshoot on the left navigation then Slow Response Times and use search and filter to narrow the results.
- B) Click Business Transactions, then the specific BT, then the Slow and Error Transactions subtab.
- C) Click Application Dashboard then Slow from the Transaction Scorecard and then use search and filter to narrow the results.
- D) Click Application Dashboard and then Transaction Snapshots. Then click Slow and Error transactions subtab and filter for that BT and user experience.

Troubleshooting Error Transactions



An error transaction can be:

- An unhandled exception
- HTTP error codes from 400 to 505
- An exception that is handled but logged with a severity of Error or Fatal using Log4j or java.util.logging NLog or Log4Net
- A handled exception that occurs during a call to an exit point

They can be accessed using the **Transaction Scorecard**, and via the **Exceptions** summary section underneath the **Transaction Scorecard** on the Application Dashboard. You can also open errors from the left navigation pane under **Troubleshoot > Errors**.

If a transaction experiences an error, it is counted as an error transaction and not as a slow, very slow or stalled transaction even if the transaction was also slow or stalled.

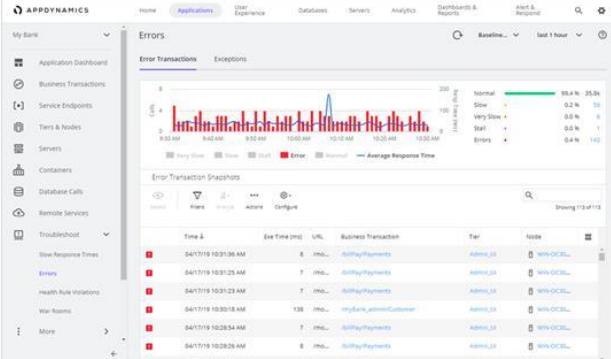
An application server exception is a code-logged message outside the context of a Business Transaction.

There is not a one-to-one correspondence between the number of errors and the number of exceptions. For example, a Business Transaction may experience a single code 500 error in which several exceptions were logged as the transaction passed through multiple tiers.

Troubleshooting Error Transactions

The **Troubleshoot > Errors** page shows all error transactions. The page contains two tabs, one for transaction errors and one for exceptions.

The tabs show information on the rate of errors or exceptions, and lets you drill down to the error or exception for more information, as shown.



The screenshot shows the AppDynamics interface with the 'Errors' tab selected. The top section displays a histogram of error transactions over time, with a legend for 'Slow', 'Very Slow', 'Stall', and 'Errors'. Below this is a table titled 'Error Transaction Snapshots' showing a list of errors with columns for Time, Err Type (err), URL, Business Transaction, Tier, and Node. Each row in the table represents an error snapshot with a red error icon.

Time	Err Type (err)	URL	Business Transaction	Tier	Node
6/4/19 10:31:09 AM	8	http://...	APIPayPayments	Admin_1d	WIN-DCB...
6/4/19 10:31:25 AM	7	http://...	APIPayPayments	Admin_1d	WIN-DCB...
6/4/19 10:31:29 AM	7	http://...	APIPayPayments	Admin_1d	WIN-DCB...
6/4/19 10:30:18 AM	138	http://...	myBank_adminCustomer	Admin_1d	WIN-DCB...
6/4/19 10:28:34 AM	7	http://...	APIPayPayments	Admin_1d	WIN-DCB...
6/4/19 10:28:26 AM	8	http://...	APIPayPayments	Admin_1d	WIN-DCB...

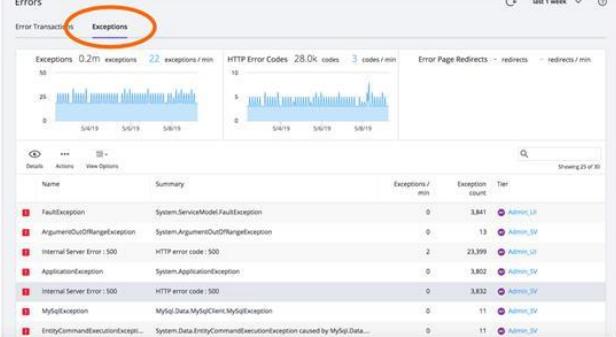
You can troubleshoot transaction errors from this screen. The error transaction snapshots are listed in the lower part of the viewer. Select the snapshot you want to investigate, and click the **View Transaction Snapshot** button to view the snapshot details including where the error occurred and information about the error.

Exceptions

Troubleshoot > Errors page > Exceptions

Exceptions may be outside the scope of a Business Transaction.

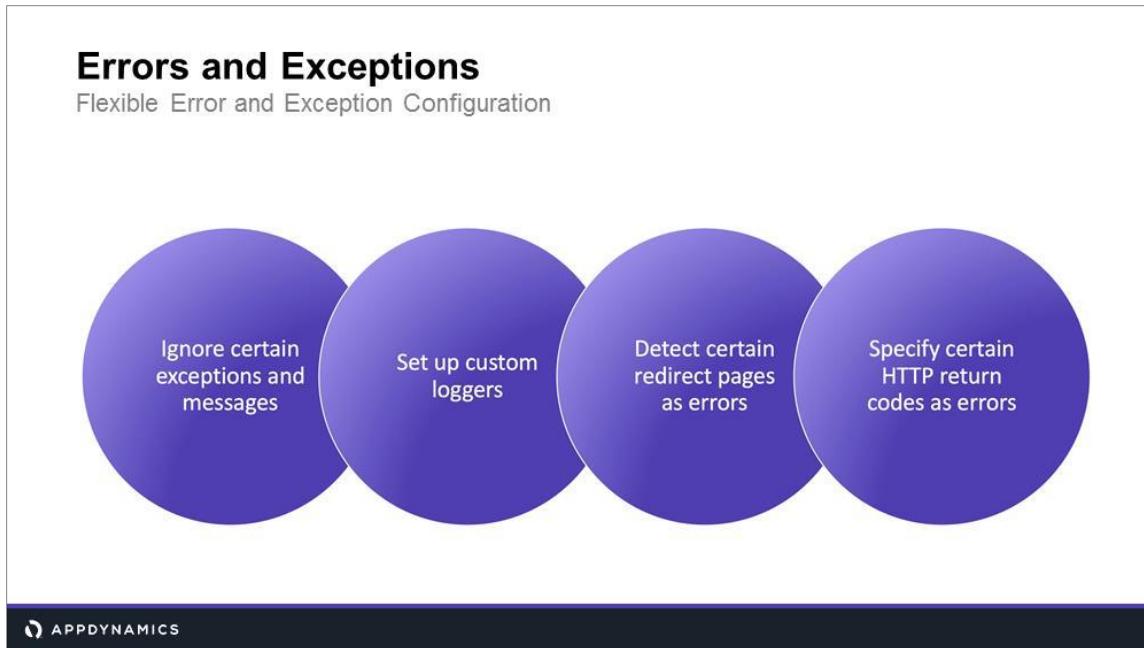
Exceptions may not impact the user experience.



APPDYNAMICS

Exceptions that are thrown and handled within a Business Transaction are not captured by AppDynamics, and do not appear in the **Exceptions** tab.

If a stack trace for the exception is available, you can access it from the **Exceptions** tab in the Controller UI. A stack trace is available for a given exception if the exception was passed in the log call. For example a logging call in the form of `logger.log(Level.ERROR, String msg, Throwable e)` would include a stack trace, whereas a call in the form of `logger.log(Level.ERROR, String msg)` would not.

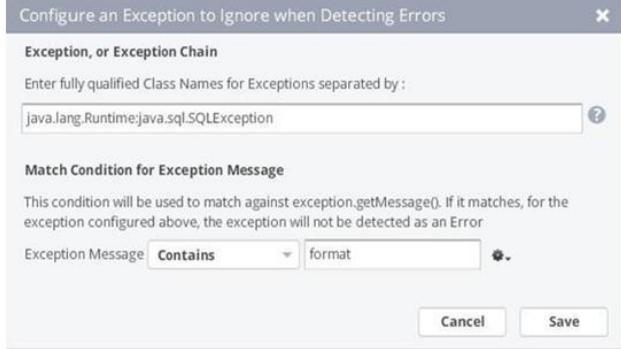


AppDynamics error detection settings are flexible, and the configuration changes you make will be applied to all tiers in the application. Access the error configuration screen by navigating to **Configuration > Instrumentation > Error Detection**.

Configuring AppDynamics to Ignore Certain Errors

You can configure AppDynamics to ignore exceptions and log messages as error indicators by adding an exception.

This is covered in the Core APM - Advanced course.



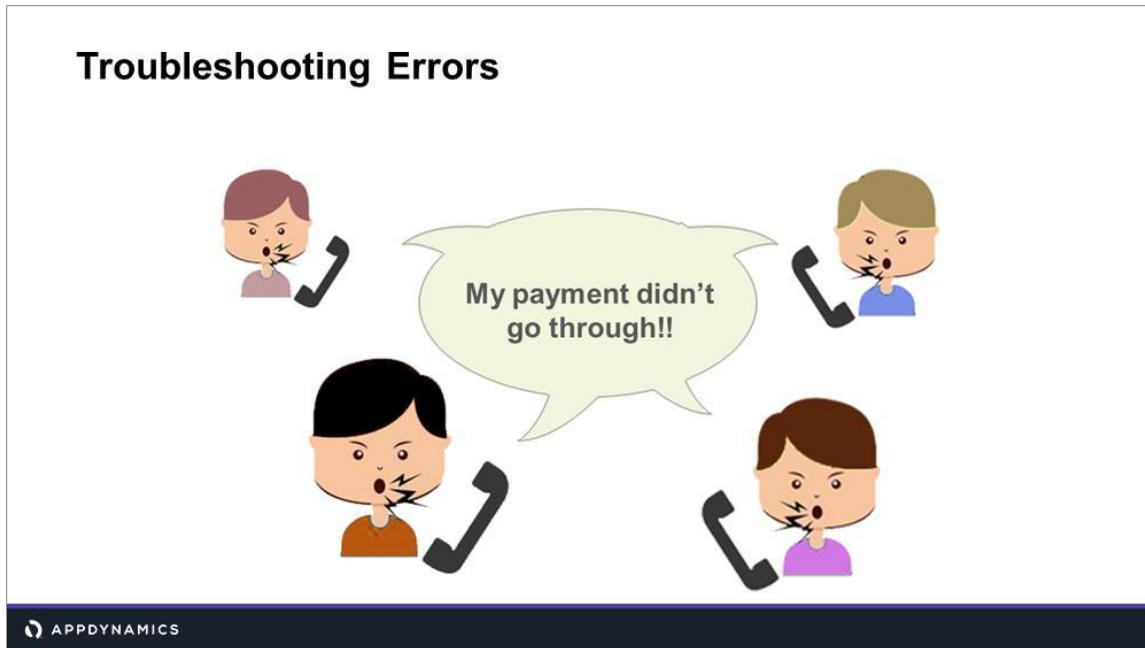
The dialog box is titled "Configure an Exception to Ignore when Detecting Errors". It has a section for "Exception, or Exception Chain" where "java.lang.RuntimeException;java.sql.SQLException" is listed. Below that is a "Match Condition for Exception Message" section with a dropdown "Exception Message" set to "Contains" and a text input "format". At the bottom are "Cancel" and "Save" buttons.

Certain types of exceptions, loggers or log messages as transaction error indicators may not reflect events that should be counted as transaction errors in your environment. They may include exceptions raised by application framework code or by user login failures.

When you configure an exception to be ignored, the agent detects the exception, increments the exception count, and displays the exception in Exceptions lists in the UI, but the Business Transaction in which the error is thrown is not considered an "error transaction" for monitoring purposes. The transaction snapshot would not show the exception in the Summary or Error Details section and the user experience for the transaction instance would be unaffected by the exception.

To configure exceptions for the agent to ignore, click the Add New Exception to Ignore button in the error configuration window.

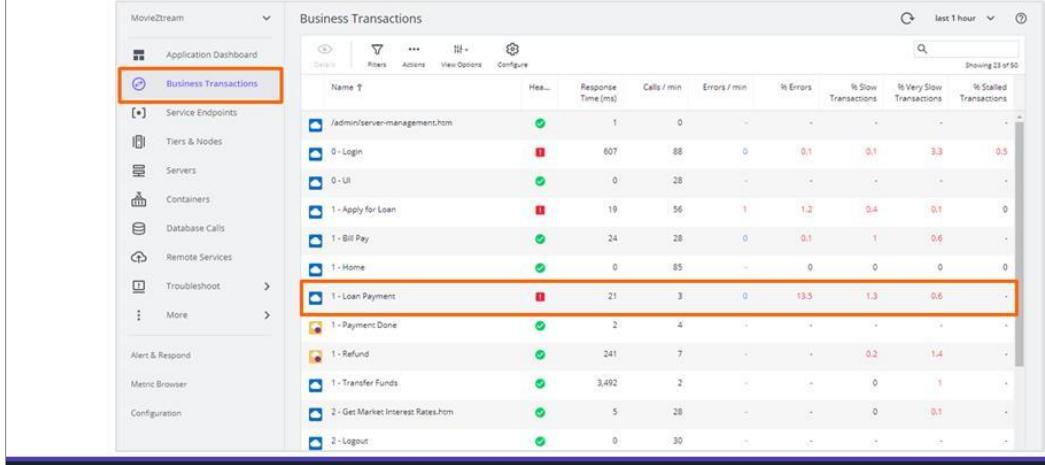
Enter the class name of the exception to be ignored and the match condition for the exception message. For the match condition, if you do not need to filter for specific messages in the exception, select "Is Not Empty". If you want to filter for Null, select "Is Not Empty" and use the gear icon to select NOT, which, in effect, tests for "is empty".



Support starts to get calls about payments not going through correctly, and users are getting error messages.

Troubleshooting errors is similar to troubleshooting slow transactions.

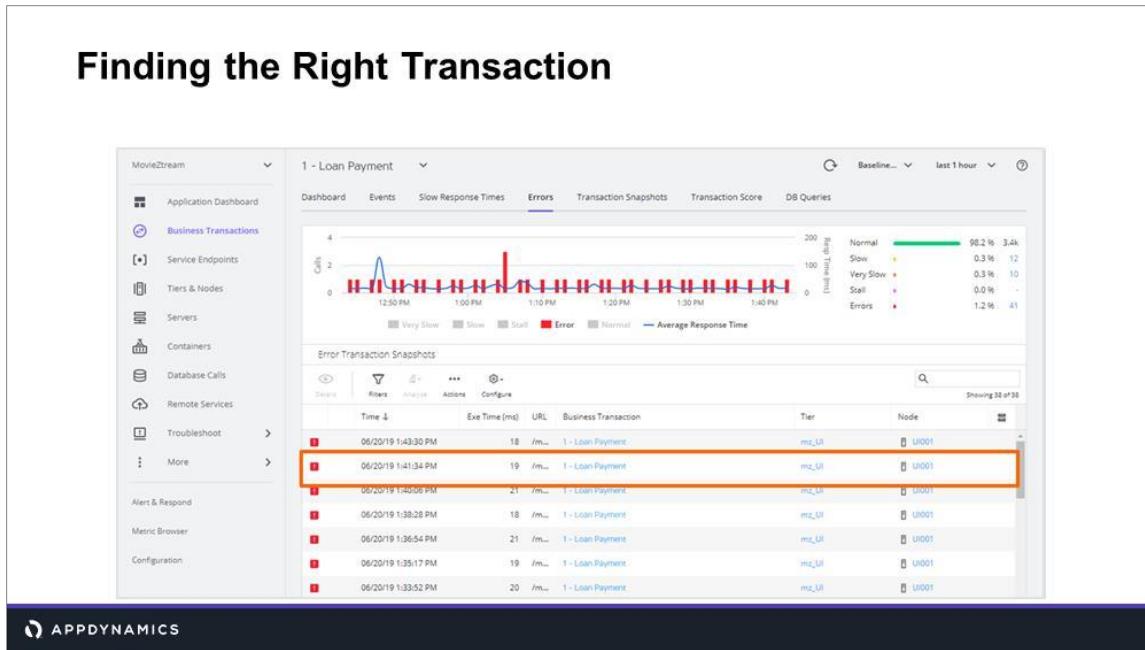
Accessing Errors



The screenshot shows the AppDynamics interface for the Movie2stream application. The left sidebar has a 'Business Transactions' link highlighted with a red box. The main content area is titled 'Business Transactions' and shows a table of transactions. The '1 - Loan Payment' transaction is highlighted with a red box. The table includes columns for Name, Health, Response Time (ms), Calls / min, Errors / min, % Errors, % Slow Transactions, % Very Slow Transactions, and % Stalled Transactions. The '1 - Loan Payment' row shows values: Health (red), Response Time (21 ms), Calls / min (3), Errors / min (0), % Errors (13.5), % Slow Transactions (1.3), % Very Slow Transactions (0.6), and % Stalled Transactions (0).

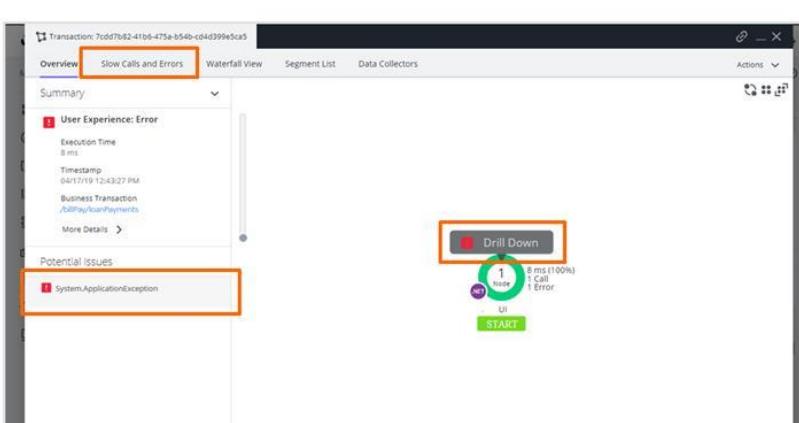
Name	Health	Response Time (ms)	Calls / min	Errors / min	% Errors	% Slow Transactions	% Very Slow Transactions	% Stalled Transactions
/admin/server-management.htm	green	1	0	-	-	-	-	-
0 - Login	red	607	88	0	0.1	0.1	3.3	0.5
0 - UI	green	0	28	-	-	-	-	-
1 - Apply for Loan	red	19	56	1	1.2	0.4	0.1	0
1 - Bill Pay	green	24	28	0	0.1	1	0.6	-
1 - Home	green	0	85	-	0	0	0	0
1 - Loan Payment	red	21	3	0	13.5	1.3	0.6	-
1 - Payment Done	green	2	4	-	-	-	-	-
1 - Refund	green	241	7	-	-	0.2	1.4	-
1 - Transfer Funds	green	3,492	2	-	-	0	1	-
2 - Get Market Interest Rates.htm	green	5	28	-	-	0	0.1	-
2 - Logout	green	0	30	-	-	-	-	-

You can either get to errors by click the **Errors** link in the Transaction Scorecard, or by expanding the **Troubleshoot** menu item on the left nav, and selecting **Errors**. In this case, we know the problem is with the online Loan Application transaction, so we can go directly to that transaction on the Business Transactions page.



With error transactions, you don't have to worry about full or partial snapshots. Just double-click the error to open it.

Error Details – Summary



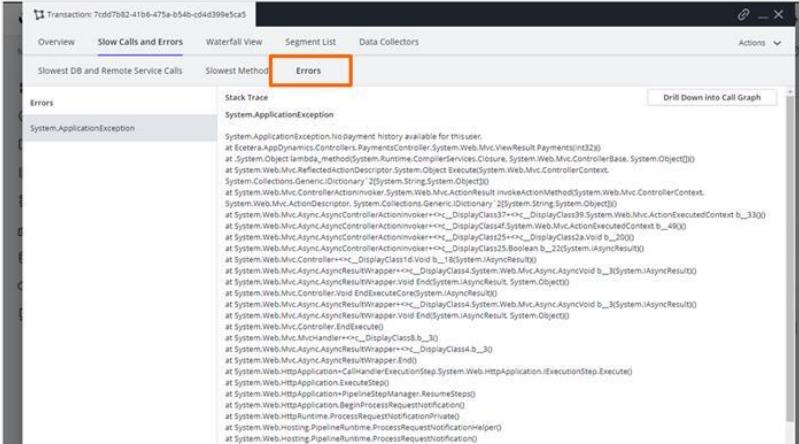
The screenshot shows the AppDynamics interface for troubleshooting. The 'Slow Calls and Errors' tab is active. In the 'Potential Issues' section, there is a single entry: 'System.ApplicationException'. On the right, a flow map displays a single node labeled 'UI START' with 8 ms (100%) execution time, 1 call, and 1 error. A 'Drill Down' button is highlighted with an orange box.

The error details screen gives you more information. The potential issues area is populated with an error icon.

You can see on the flow map that the error occurred on the Admin_UI tier, and you can drill down to find out more.

You can also click the **Slow Calls and Errors** tab to find out more. Let's go that route and see what's there.

Snapshot Details – Slow Calls and Errors



The **Slow Calls and Errors** tab can give you more details on specific slow calls, methods, and errors. In this case, we need to see the **Errors** tab, which shows us the specific error.

The **Slow Calls and Errors** tab can give you more details on specific slow calls, methods, and errors. In this case, we need to see the **Errors** tab, which shows us the specific error.

Review Question - Answer

Which of the following are considered errors by default in AppDynamics?

- A) An unhandled exception.
- B) HTTP error codes from 150 to 350.
- C) HTTP error codes from 400 to 505.
- D) An exception that is handled and logged with a severity of Error or Fatal using Log4j or java.util.logging NLog or Log4Net.
- E) A handled exception that takes longer than 10 ms.
- F) A handled exception that occurs during a call to an exit point.

Review Question - Answer

What are some options you can configure for error detection?

- A) Specify certain HTTP return codes to be errors.
- B) Ignore certain exceptions and messages.
- C) Detect certain redirect pages as errors.
- D) Set up custom loggers.

What You Learned

- What a diagnostic session is and the different ways you can run a diagnostic session
- How to troubleshoot different application issues using transaction snapshots
- How AppDynamics defines an error

Differentiate yourself and your company with AppDynamics Certification

APPDYNAMICS | Certification Program

General Certification Information

<https://learn.appdynamics.com/certifications>

AppDynamics Certified Associate Performance Analyst

<https://learn.appdynamics.com/certifications/performance-analyst>

AppDynamics Certified Associate Administrator

<https://learn.appdynamics.com/certifications/administrator>

AppDynamics Certified Implementation Professional

<https://learn.appdynamics.com/certifications/implmenter>

APPDYNAMICS

To differentiate yourself and your company in an increasingly competitive market, please consider becoming AppDynamics certified.

For more information, please copy and paste this URL into a separate tab in your browser: <https://learn.appdynamics.com/certifications>

For information on specific certification tracks, please copy and paste the appropriate URLs below:

AppDynamics Certified Associate Performance Analyst:

<https://learn.appdynamics.com/certifications/performance-analyst>

AppDynamics Certified Associate Administrator:

<https://learn.appdynamics.com/certifications/administrator>

AppDynamics Certified Implementation Professional:

<https://learn.appdynamics.com/certifications/implmenter>

Advanced Troubleshooting and Tools (APM213)



University

APM213 - Advanced Troubleshooting and Tools

Core APM I: Essentials - Module 3

Objectives

Course

After completing this course, you will be able to:

- Describe how to troubleshoot common database, bottlenecks, and thread contention issues
- Explain what a data collector is and the kind of data they collect
- Use the node level dashboards to troubleshoot node issues
- Explain how to use the Memory Dashboard to monitor and identify troubled segments

Labs

In this course's labs, you will:

- Troubleshoot slow database calls
- Troubleshoot with Data Collectors
- Troubleshoot node-level issues

Too Many / Slow Database Calls

Too Many DB / Slow Database Calls

More data is requested than needed

EXAMPLE: Pulling all the account information and not the specified data.

Same data gets requested multiple times

EXAMPLE: Accidentally placing a call to the database inside a loop.

Multiple queries required to form the dataset

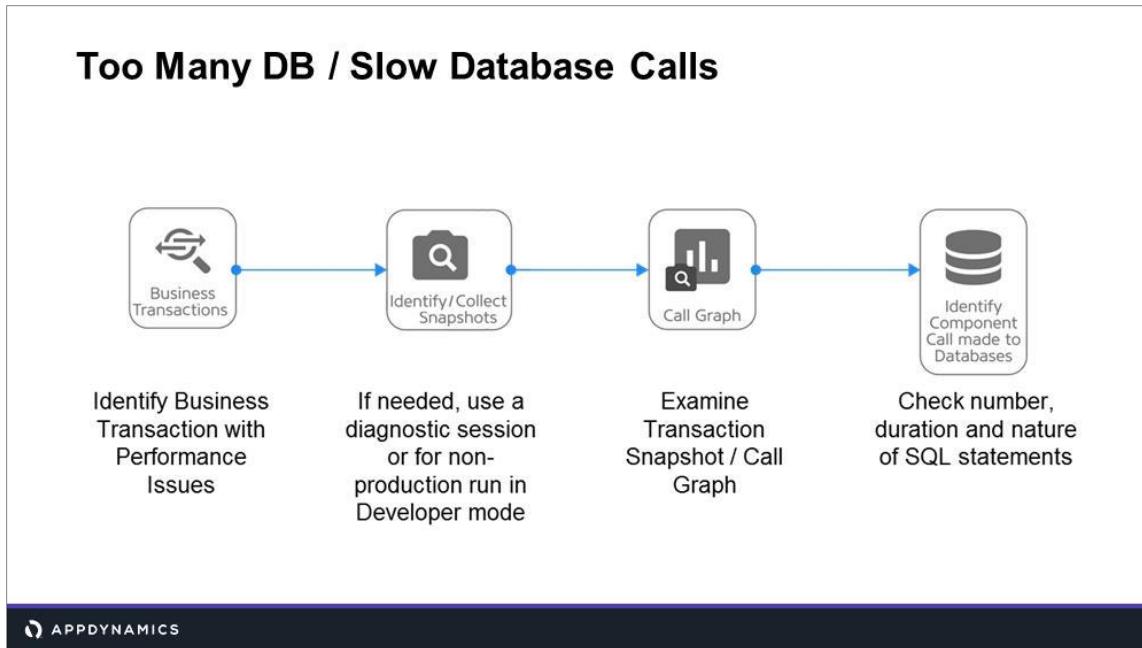
EXAMPLE: Not taking advantage of complex SQL statements or stored procedures to retrieve data in one batch.



One problem we see frequently is too many database queries per request/transaction. In this context, “too many” means more than necessary.

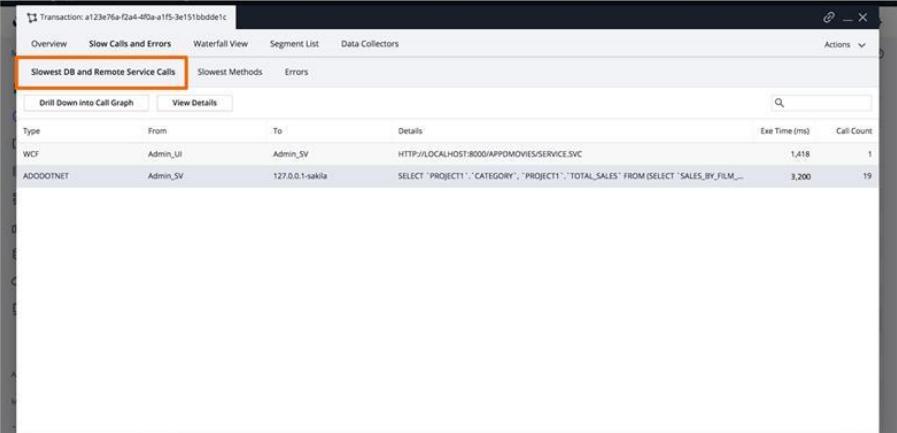
There could be a number of database issues. In this topic, we’re just going to discuss the following three issues.

- More data is requested than actually required in the context of the current transaction, e.g.: requesting all account information instead of only requesting the pieces we need to display on the current screen.
- The same data is requested multiple times in the same transaction. This usually happens as a result of misplaced code.
- Multiple queries are executed to retrieve a certain set of data. This is often a result of not taking full advantage of complex SQL statements or stored procedures and pagination to retrieve the data in one batch.



For troubleshooting transactions that have too many database calls, step through this process – looking at performance for each query as you go, and evaluate whether you’re using “too many” database queries. Be aware that database connections as well as database processing time tend to be precious application resources, which should be conserved.

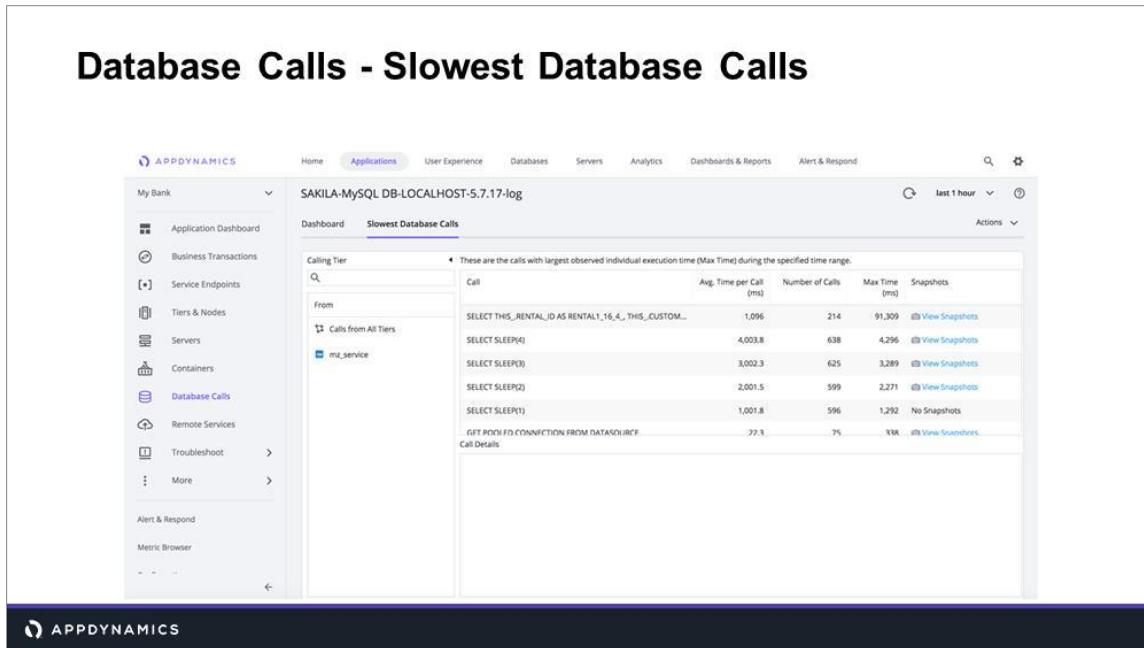
Transaction Snapshot Details - Slowest DB Calls



Type	From	To	Details	Exe Time (ms)	Call Count
WCF	Admin_UI	Admin_SV	HTTP/LOCALHOST:8000/APPMOVIES/SERVICE.SVC	1,418	1
ADO.NET	Admin_SV	127.0.0.1-sakila	SELECT 'PROJECT1'.'CATEGORY', 'PROJECT1'.'TOTAL_SALES' FROM (SELECT 'SALES_BY_FILM,...	3,200	19

The **Slow Calls and Errors** tab on the **Transaction Snapshot** details screen has three subtabs for the slowest calls to DB and remote services, the slowest methods, and errors.

Database Calls - Slowest Database Calls



Call	Avg. Time per Call (ms)	Number of Calls	Max Time (ms)	Snapshots
SELECT THIS_RENTAL_ID AS RENTAL1_16_4_, THIS_CUSTOM...	1,096	214	91,309	View Snapshots
SELECT SLEEP(4)	4,003.8	638	4,296	View Snapshots
SELECT SLEEP(3)	3,002.3	625	3,289	View Snapshots
SELECT SLEEP(2)	2,001.5	599	2,271	View Snapshots
SELECT SLEEP(1)	1,001.8	596	1,292	No Snapshots
GET POOLID CONNECTION FROM DATASOURCE	22.1	75	318	View Snapshots

You can take a broader look at the slowest calls by tier by clicking Database Calls on the left menu. This opens the database dashboard which displays communication, load, and error health. Click the **Slowest Database Calls** sub-tab to see a list of calls with long execution times. You can then drill into snapshots.

Review Question - Answer

How would you diagnose too many or slow database calls?

- A) Snapshots > Bottlenecks
- B) Snapshots > Drill Down > Call Graph > Backend Calls**
- C) Snapshots > Slow Calls and Errors > Slowest DB and Remote Calls sub-tab**
- D) Database Calls > Slowest Database Calls sub-tab

Thread State Analysis

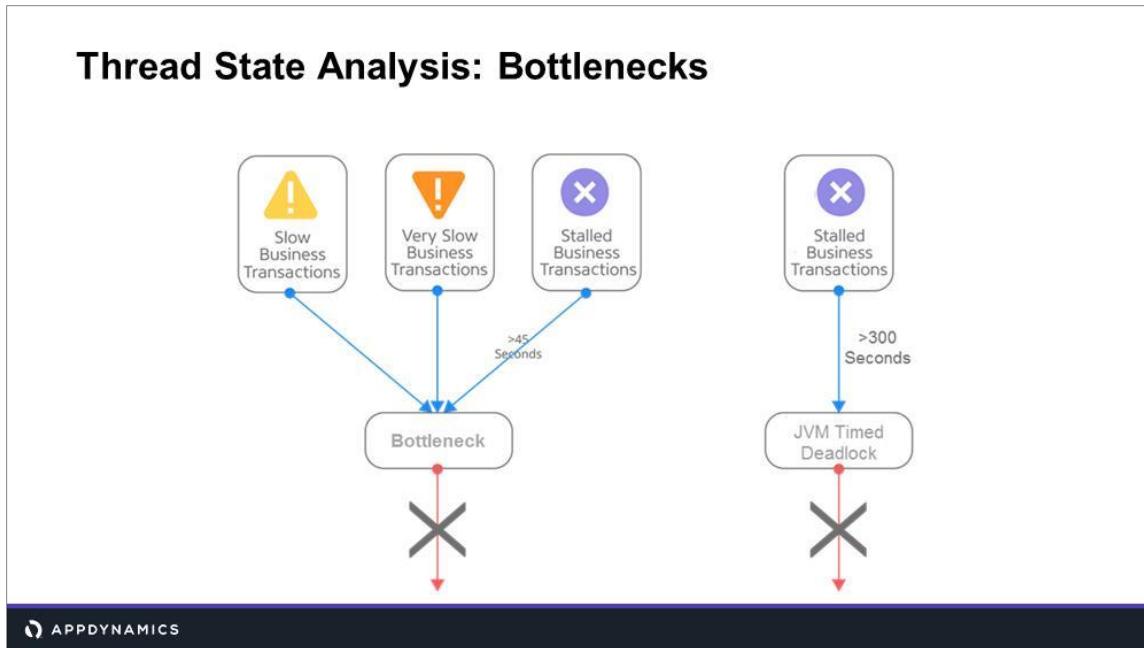
Thread State Analysis: Bottlenecks

Defining characteristics

- Excessively large code sequences are synchronized
 - Developers adopt defensive coding practices
 - Lengthy operation to refactor code to ensure only the necessary minimum code is synchronized
- Under low load (on the local developers workstation) performance will not be significantly affected
- In a high load environment over-synchronization results in severe performance and scalability problems



There is no question that synchronization is necessary to protect shared data in an application. Too often, application developers make the mistake of over-synchronization, e.g.: excessively-large code sequences are synchronized. Under low load—such as on the local developers workstation—performance will not suffer significantly. However, in a high load environment, such as a production enterprise application, over-synchronization results in severe performance and scalability problems.



Since the stall and the deadlock are monitored from 2 independent sources it is possible that a stall in a Business Transaction can appear as a deadlock along the timeline. However not every deadlock will appear as a stall only those that are within a Business Transaction.

You may have a number of different Business Transactions that are slowing down. When you look at the snapshots you may see a variety of slow, very slow, and stalled transactions. All of these could be attributed to a single bottleneck.

Thread State Analysis: Thread Contention - JAVA

Thread Contention refers to a situation where one or more threads are blocked/waiting for a resource that is currently being held by another thread.

The blocked/waiting threads are queued, waiting for their turn to access the resource.

The Controller can display details about thread contention for business transactions

- Select a transaction snapshot > Transaction Flow Map > Potential Issues
- The issues labelled as Thread Contentions display the blocked and or waiting time for the thread



Waiting threads cannot use the resource until the other thread has released the specific object. This results in slow or blocked performance.

Multithreading is a type of concurrent processing that uses resources more efficiently. However, multi-threading can create conditions that result in slow or blocked performance.

For example: two threads attempt to access the same resource (or related resource) this could result in performance suffering.

Synchronising methods or statements can prevent thread interference, memory consistency and deadlock patterns.

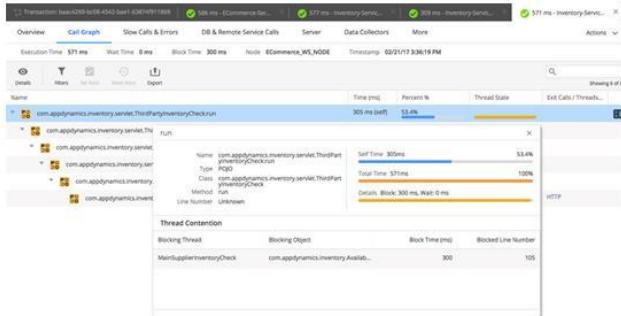
However synchronisation could introduce thread starvation and livelock.

Thread State Analysis: Thread Contention - JAVA

Details surfaced in AppDynamics call graphs

For the target method the following details are displayed:

- Blocking Thread
- Blocking Object
- Block Time
- Source code line number



Blocking Thread: com.appdynamics.inventory.service.ThirdPartyInventoryCheck

Business Transaction Thread Contention

To view details about thread contention for a business transaction:

Select a transaction snapshot > **Transaction Flow Map** > **Potential Issues**.

The issues labelled as Thread Contentions display the blocked time for the thread.

Double-click one of thread contention issues to display further information about the method that is blocked. Select **Drill Down into Call Graph**.

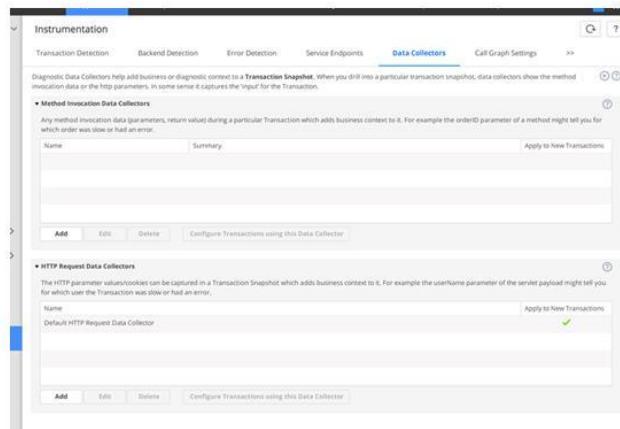
The actual wait state could be as a result of any of the following:

- Thread.sleep
- Object.wait
- Thread.join
- LockSupport.parkNanos
- LockSupport.parkUntil
- LockSupport.park

Data Collectors

Data Collectors

- Troubleshoot specific transactions causing problems
- Collect data displayed as a string
- Can include application code arguments, return values, and getter chain
- Displays the information in a tab in transaction snapshot details
- Can be a data security risk



Data collectors can collect any data passing through an application as long as it can be displayed as a string. They can help determine whether data that a transaction passed into an application is causing problems.

When you configure data collectors, AppDynamics accesses information in application code arguments, return values, and getter chain and displays the information in the transaction details drill down screen.

Data Collectors are also used to collect data for the analytics engine.

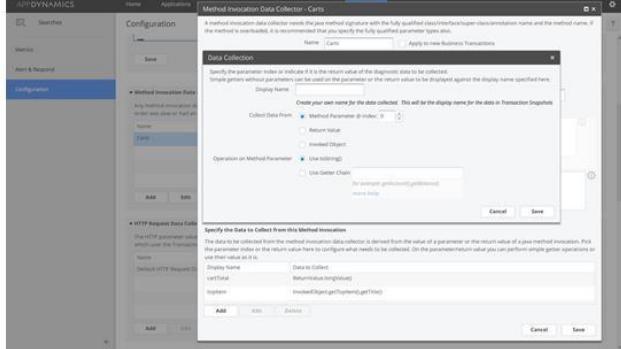
Two Types of Data Collectors

Method Invocation Data Collectors

- Method arguments
- Variables
- Return values

HTTP Requests Collectors

- URL
- Parameter values
- Headers
- Cookies
- Session objects



Method Invocation Data Collectors

- Capture code data such as method arguments, variables, and return values
- Use to create custom fields to collect business related data

HTTP Requests

- Capture the URLs, parameter values, headers, and cookies of HTTP messages exchanged in a business transaction
- When HTTP data collectors are configured, the following information can be collected (see list on slide)

Data Collector Use Case

- The **/billPay/Payments** business transaction is experiencing sporadic errors
- You could set up a method invocation data collector to extract business information (type of card, payment amount) for specific requests
- You will now know the context in which errors are occurring, and can use this knowledge to diagnose the root cause

Notes From the Field

Data Collectors

Business Problem

- A customer was getting random errors in an API call.

Solution

- Further investigation revealed that a required field was not being validated to ensure that an entry had been made.
- We set up a data collector and noticed that for one field, there was a null value.

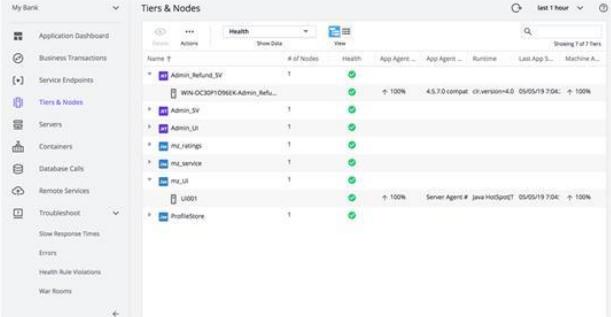


Troubleshooting Node Level Issues

Tiers & Nodes

Quickly view node status for:

- General health
- Hardware
- Memory (Java and .NET)
- Network

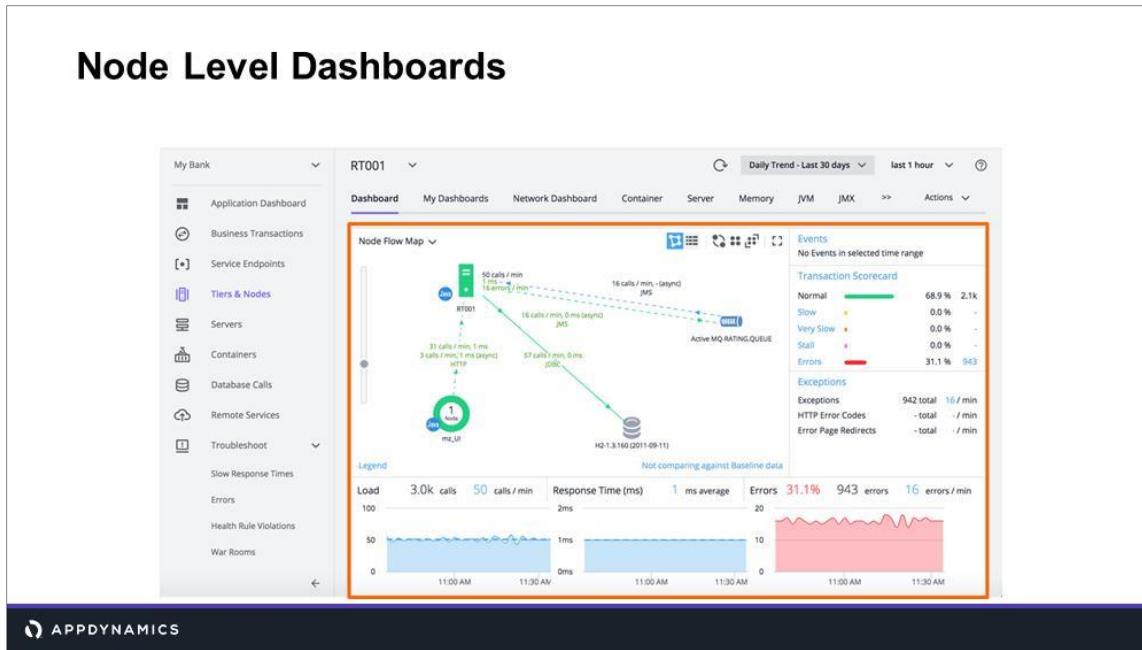


APPDYNAMICS

Clicking **Tiers & Nodes** in the left menu opens a list of the tiers and nodes in your application. From the main screen, you can select a node and click Actions to view the health rule violations that have occurred on that node.

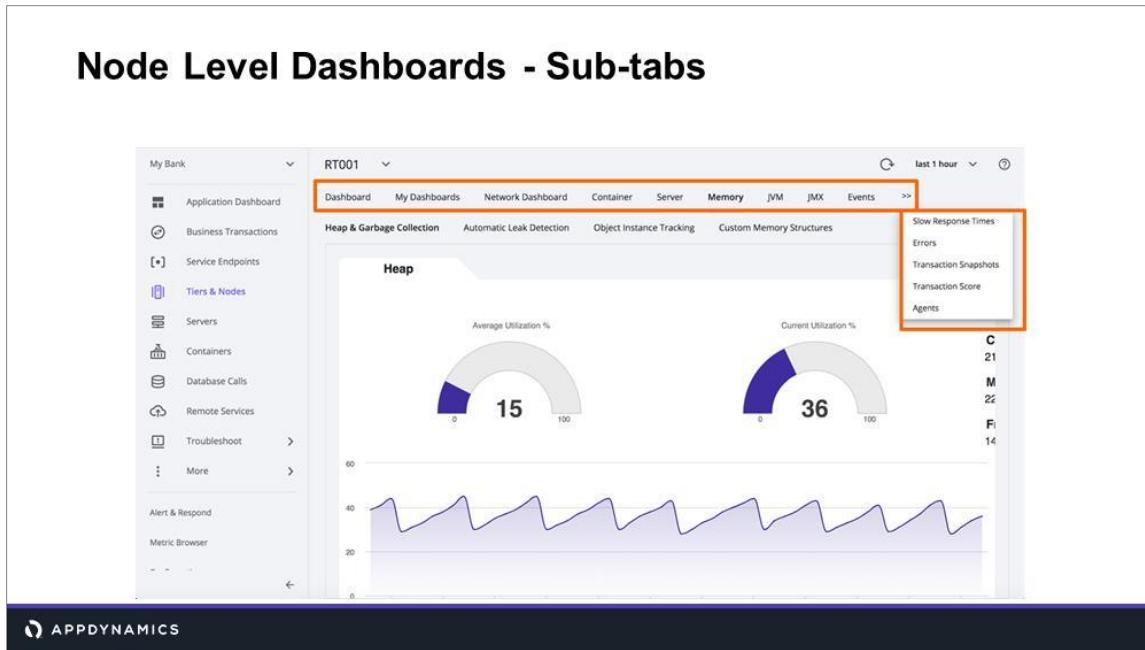
You can also use the **Show Data** drop down and select from Health, Hardware, Memory, and Network to see data for each of these.

You can also drill down into each node by either selecting the node and clicking **Details**, or by double clicking a node.



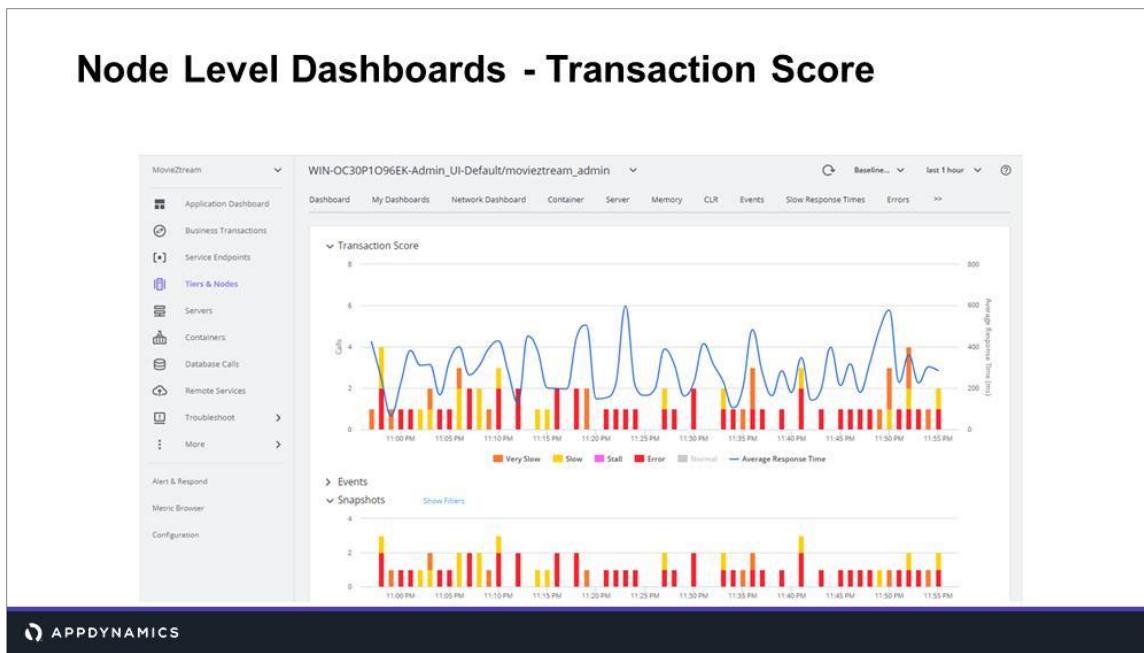
As with the Application and Business Transactions dashboards, each node has a dashboard where you can view the node flow map, load, error and response time information, and a transaction scorecard. You can also view exceptions and errors, and any events that occurred during the selected time range.

Node Level Dashboards - Sub-tabs



The screenshot shows the AppDynamics Node Level Dashboard for node RT001. The left sidebar lists various monitoring categories: Application Dashboard, Business Transactions, Service Endpoints, Tiers & Nodes, Servers, Containers, Database Calls, Remote Services, Troubleshoot, and More. The Troubleshoot section is expanded, showing Alert & Respond and Metric Browser. The main dashboard area is titled 'Heap' and displays two gauge charts: 'Average Utilization %' (15) and 'Current Utilization %' (36). Below these are two line charts showing utilization over time. A sub-tab menu on the right is highlighted with an orange box, containing options: Slow Response Times, Errors, Transaction Snapshots, Transaction Score, and Agents. The bottom right corner shows performance metrics: C 21, M 22, and F 14. The AppDynamics logo is at the bottom left.

In addition to the dashboard information, there are sub-tabs across the top that give you access to additional information for that node - from server and memory metrics, to transaction snapshots and score. Let's take a closer look at Transaction Score.



You can click the **Transactions** tab on the **Tier** dashboard screen to see all transactions.

For a more detailed view of the performance of transactions on that tier over time, click the **Transaction Score** tab. The graph in the tab shows the user experience for that tier, showing its performance relative to the performance thresholds as a bar chart.

For each time slot, the bar reflects the total calls, while the color segments—green, yellow and so on—indicate the relative number of those calls that were normal, slow, very slow, stall or errors.

The solid blue line in the graph indicates the average response time for the transaction over time, while the dotted line shows the baseline, if available.

Memory Management

Memory Management

Manage Garbage Collection

Major Collections / Large Object Heap

- Consume a lot of time, and increase Application Latency
- If frequent, then Heap may need to be tuned

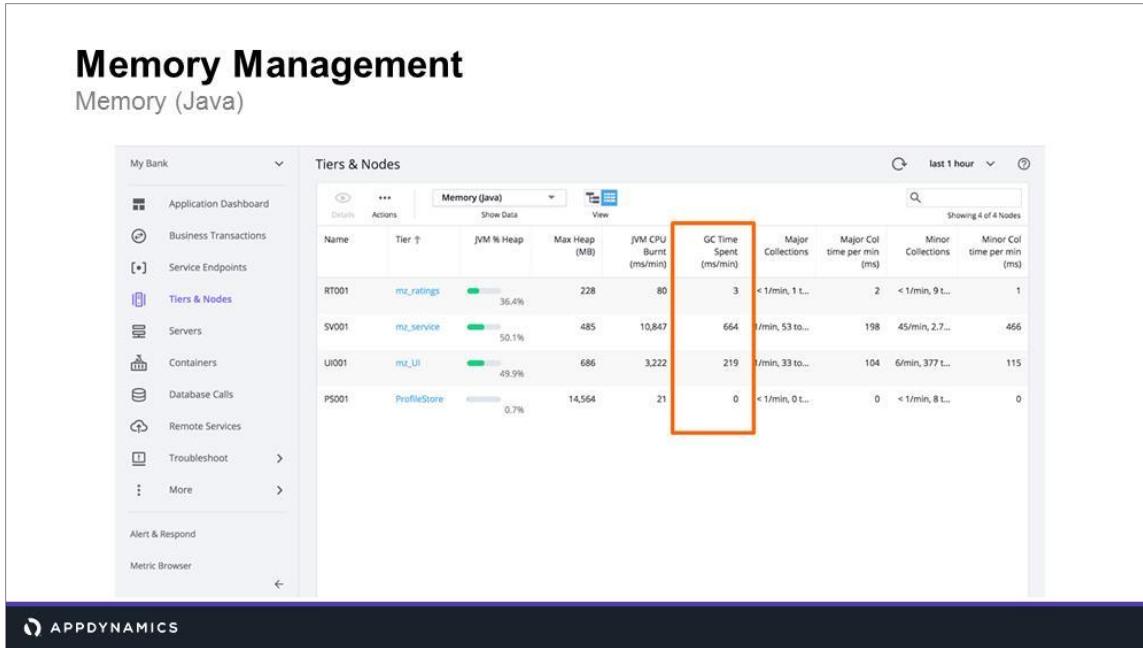
Minor Collections / Gen 0, 1, 2

- Should collect short lived objects before promotion to 'old gen'
- If Young Gen pools are too small, Objects can be promoted prematurely
- If frequent, then Heap may need to be tuned



If major garbage collections occur frequently, or consume a lot of time, application latency might increase, as many garbage collection algorithms suspend other application tasks as they compact the memory pools. A large amount of time spent garbage collecting also affects application throughput; CPU cycles spent on garbage collection reduce the amount of processor time available to process user requests.

Frequent major garbage collections are also an indication that the heap may be improperly tuned. Short-lived objects should be collected during minor garbage collections, before they get promoted to long-lived ("old generation") memory pools. If the memory pools reserved for short-lived ("young generation") objects are too small, objects may be promoted prematurely, and as a result both minor and major garbage collections occur more frequently than needed

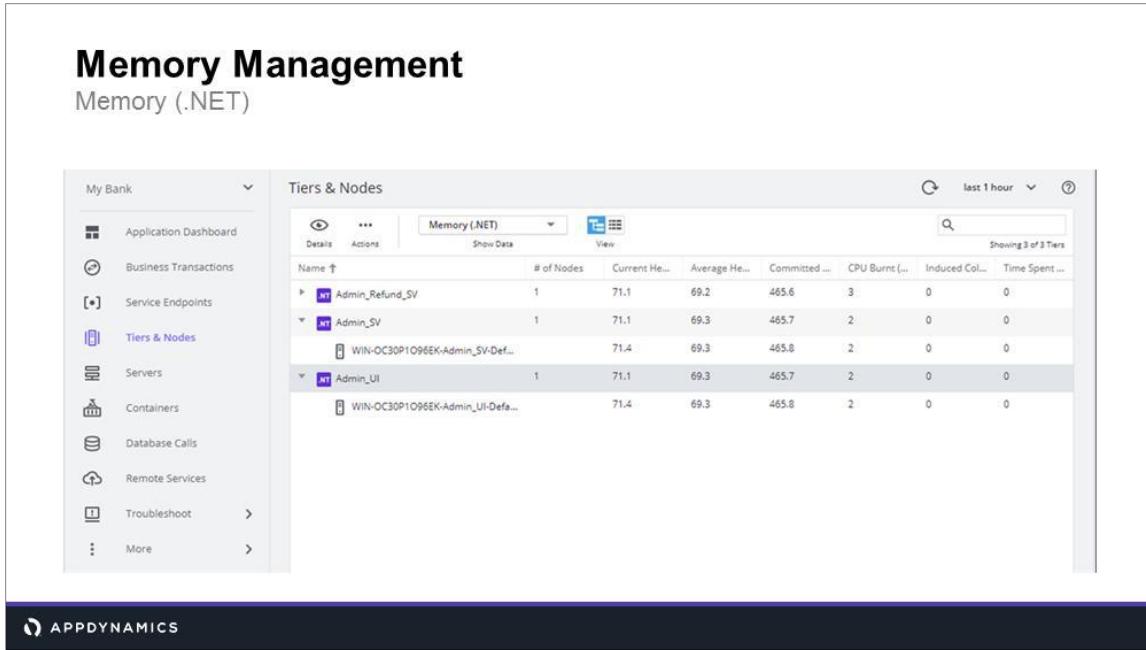


The screenshot shows the AppDynamics Memory Management dashboard for the 'My Bank' application. The left sidebar includes links for Application Dashboard, Business Transactions, Service Endpoints, Tiers & Nodes (which is selected and highlighted in blue), Servers, Containers, Database Calls, Remote Services, Troubleshoot, and More. The main content area is titled 'Memory (Java)' and displays a grid of JVM nodes. The columns in the grid are: Name, Tier, JVM % Heap, Max Heap (MB), JVM CPU Burnt (ms/min), GC Time Spent (ms/min), Major Collections, Major Col time per min (ms), Minor Collections, and Minor Col time per min (ms). Four nodes are listed: RT001 (mz_ratings, 36.4%, 228, 80, 3, <1/min, 1 t...), SV001 (mz_service, 50.1%, 485, 10,847, 664, 1/min, 53 t...), UI001 (mz_UI, 49.9%, 686, 3,222, 219, 1/min, 33 t...), and PS001 (ProfileStore, 0.7%, 14,564, 21, 0, <1/min, 0 t...). The 'GC Time Spent (ms/min)' column for PS001 is highlighted with an orange box. The bottom of the dashboard features the AppDynamics logo.

Name	Tier	JVM % Heap	Max Heap (MB)	JVM CPU Burnt (ms/min)	GC Time Spent (ms/min)	Major Collections	Major Col time per min (ms)	Minor Collections	Minor Col time per min (ms)
RT001	mz_ratings	36.4%	228	80	3	<1/min, 1 t...	2	<1/min, 9 t...	1
SV001	mz_service	50.1%	485	10,847	664	1/min, 53 t...	198	45/min, 2.7...	466
UI001	mz_UI	49.9%	686	3,222	219	1/min, 33 t...	104	6/min, 377 t...	115
PS001	ProfileStore	0.7%	14,564	21	0	<1/min, 0 t...	0	<1/min, 8 t...	0

In grid view, in the **Tiers & Nodes List > Memory (Java) dropdown**. The running Java JVMs in your environment are displayed.

Sort the nodes by GC Time Spent in descending order. You can then identify where garbage collection is taking too long and possibly hindering app performance.



The screenshot shows the AppDynamics interface for Memory Management. The left sidebar is titled 'My Bank' and includes links for Application Dashboard, Business Transactions, Service Endpoints, Tiers & Nodes (which is selected and highlighted in blue), Servers, Containers, Database Calls, Remote Services, Troubleshoot, and More. The main content area is titled 'Tiers & Nodes' and shows a table of data. The table has columns for Name, # of Nodes, Current He..., Average He..., Committed ..., CPU Burnt ..., Induced Col..., and Time Spent There are three tiers listed: Admin_Refund_SV, Admin_SV, and Admin_UI. Each tier has one node listed under it. The data for Admin_Refund_SV shows 1 node, 71.1 current health, 69.2 average health, 465.6 committed, 3 CPU burnt, 0 induced, and 0 time spent. The data for Admin_SV and Admin_UI show similar values: 1 node, 71.1 current health, 69.3 average health, 465.7 committed, 2 CPU burnt, 0 induced, and 0 time spent. A search bar and a time filter for 'last 1 hour' are at the top of the table. The AppDynamics logo is at the bottom left.

Name	# of Nodes	Current He...	Average He...	Committed ...	CPU Burnt ...	Induced Col...	Time Spent ...
Admin_Refund_SV	1	71.1	69.2	465.6	3	0	0
Admin_SV	1	71.1	69.3	465.7	2	0	0
WIN-OC30P1O96EK-Admin_SV-Def...		71.4	69.3	465.8	2	0	0
Admin_UI	1	71.1	69.3	465.7	2	0	0
WIN-OC30P1O96EK-Admin_UI-Def...		71.4	69.3	465.8	2	0	0

You can view memory health for each tier/node by selecting Memory (Java) or Memory (.NET) on the Tiers & Nodes page. Memory health should be fairly consistent across nodes of a given tier.

Memory Dashboard

Heap and Garbage Collection Tab

Metrics	Graphs	Generational Heap Analysis
Current and Average Heap Utilization percent	Heap Utilization Percent	Code Cache
Current Used, Committed, Maximum Available and Free memory in MB	Max Heap vs. Used Heap	Eden Space
	Minor and Major Garbage Collections: Garbage Collection Time Spent, Number of Garbage Collections per Minute	Old Gen / Tenured Gen
		Survivor Space

 APPDYNAMICS

Memory metrics vary between Java and .NET.

Memory Management

In-Depth JVM GC Metrics (Java Only)

These metrics are available in the Metric Browser, and support both the CMS Actually 'Parallel/Throughput' collector is default) and the new G1 (Garbage First) collector that is the default as of Java 8.

- Object Allocation Rate
- Object Promotion Rate
- Live Data Size
- Object Free Rate



***CMS = 'Concurrent-Mark-Sweep'

Object Allocation Rate - tracks the number of objects created per minute.

Object creation almost always happens in the Eden space.

- This metric is important for tuning Young Gen size, as a very high allocation rate means Eden space fills up quicker, which leads to more frequent minor garbage collection (a Stop-The-World event).
- By understanding the relationship between the allocation rate and two other metrics: free rate and promotion rate, the young heap can be sized to fill up less frequently.

Object Promotion Rate - tracks the number of objects promoted from young to old gen per minute.

- This metric is important for tuning both young and old size, as a very high promotion rate means Eden (or Survivor) space may need to be increased, and leads to quicker filling of the old gen, which leads to more frequent major garbage collection (a Stop-The-World event if using default Parallel/Throughput collector memory collection and might for the others).

Live Data Size - tracks the memory footprint of the application.

- This metric is useful for understanding the effect of application changes or changes in load characteristics on the amount of memory the application needs to work efficiently.
- Any change that results in a higher memory footprint needs to be analyzed – having this metric allows analysis of memory footprint regression.

Object Free Rate - tracks the number of objects freed from young gen per minute.

- This metric is important for tuning the size of the young generation and can give some clues to possible coding issues.
- It can be used to analyze the effect of young gen tuning, and can be compared to Object Allocation Rate to anticipate and detect memory leaks.

Memory Management

Memory (Java)

- Memory is typically divided up into multiple segments, and not all of them have problems all at once.
- Keeping an eye on the condition of the individual memory pools is a good idea in quickly identifying the troubled segments.
- Accessed via **Tiers & Nodes** **[tier] > [node] > Memory > Heap & Garbage Collection > Memory Pools**

Dashboard My Dashboards Network Dashboard Container Server Memory JVM >>

Heap & Garbage Collection Automatic Leak Detection Object Instance Tracking >>

Memory Pool

Name	Used ...	Committed...	Max Availa...	Current Utili...	Current Usage Trend
Compr...	7	7	1024	0%	
PS Old...	402	512	512	78%	
PS Sur...	51	80	80	63%	
Metas...	68	69	0	0%	
PS Ed...	40	93	94	42%	
Code ...	55	56	240	22%	

 APPDYNAMICS

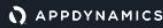
Memory Management

Object Instance Tracking

Use Object Instance Tracking > Identify Business Transactions and code paths responsible for excessive object allocations.

Tracking Tab

- Shows the allocation trend for:
 - Top Application Classes
 - Top System Classes
 - Custom Classes
- Allows a drill-down into how objects are being instantiated and which part of your application code is creating each object type
- Useful for diagnosing memory leak / memory thrash situations



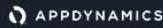
OIT tracks objects that are created, but never released. Thus helps pinpoint a type of memory leak.

Once the object types and memory allocation patterns have been identified, you can then start an allocation tracking session to identify the Business Transactions and code paths that are responsible for the excessive object allocations.

Memory Management

Automatic Leak Detection - Java only

- Monitors common collection classes in your application
- Shows monitored collections. If this grows over time beyond normal fluctuations, a memory leak may exist
- Useful for detecting collection-based memory leaks
- Can cause significant overhead:
 - Best used in dev | test
 - If used in production, enable on a single node rather than the whole tier

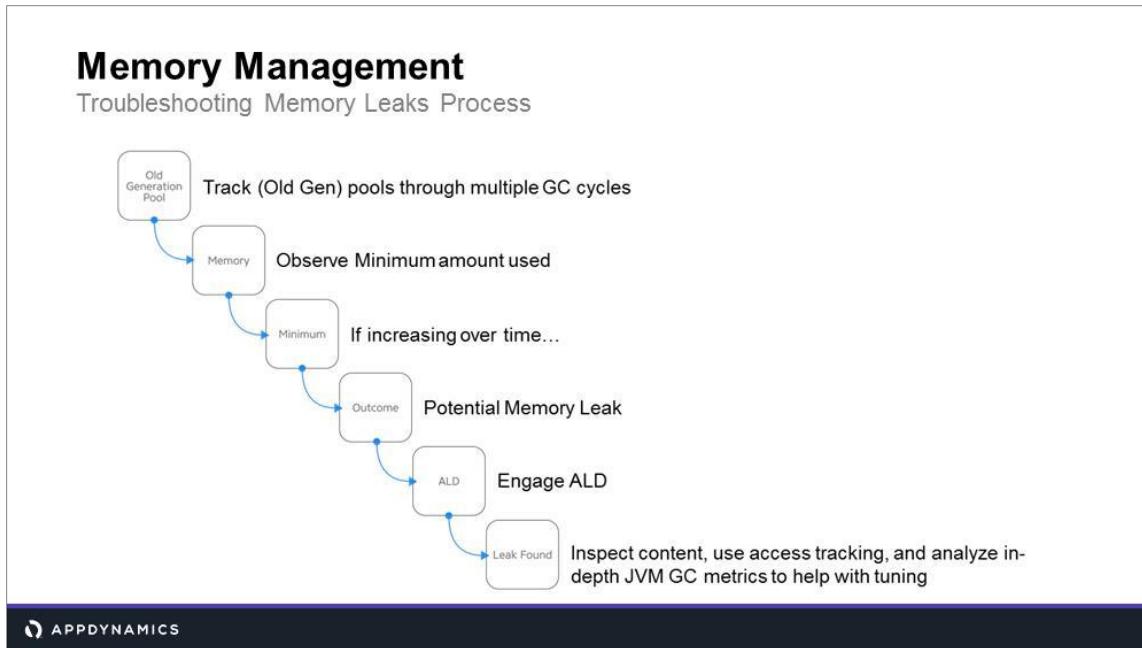


A collection is also called a 'container.'

Common collection types are any class that implements JDK 'Map' or 'Collection' interface, for example:

- Hash maps
- Vectors
- Linked lists

To be a candidate, by default the collection must be at least 5mb in size and have 1000 or more elements.



If you track the long-lived (old generation) memory pools over multiple major garbage collection cycles, and observe that the minimum amount of memory used (the valleys in the graph) is steadily increasing over time, the application may be suffering from a memory leak.

The Java Agent offers an automatic leak detection feature to help identify the source of the memory leak. This feature tracks collections over time to identify growing collections. The minimum collection age and size is configurable. If a potential memory leak is found, you can inspect the collection content or use access tracking to determine which code paths are accessing the growing collection.

Troubleshooting Scenarios

Problem Diagnosis

Too many | Slow database calls → Snapshot details > Slow Calls and Errors

Bottlenecks → Snapshots

Memory Leaks | Memory Thrashing → Node-level memory dashboards > Automatic Leak Detection (Java) and Object Instance Tracking

Node Problems → Compare performance of different nodes

Thread Contention → Snapshots



A frequent problem is too many database queries per request/transaction and/or queries that are too slow, but what is “too many” or “too slow” in this context? There are 3 specific instances:

- More data is requested than is actually required in the context of the current transaction.
- The same data is requested multiple times by in a single transaction.
- Multiple queries are executed to retrieve a certain set of data. This is often a result of not taking full advantage of complex SQL statements or stored procedures to retrieve the data in one batch.

Too often people make the mistake of over-synchronization (e.g. excessively-large code sequences are synchronized). In a high-load or production environment over-synchronization results in severe performance and Memory Leaks.

It is important to release object references as soon as they are no longer needed. It is important to understand that Java’s Automatic Garbage Collection does not prevent memory leaks. Additionally, web application reloading can affect parts of the memory heap and cause unsuspected memory leaks.

What You Learned

- How to troubleshoot common database, and thread contention issues including over-synchronization
- The types of data collectors and their uses in troubleshooting
- How to use memory management to troubleshoot segments
- How to use the node level dashboards to troubleshoot node issues

Differentiate yourself and your company with AppDynamics Certification

APPDYNAMICS | Certification Program

General Certification Information

<https://learn.appdynamics.com/certifications>

AppDynamics Certified Associate Performance Analyst

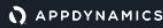
<https://learn.appdynamics.com/certifications/performance-analyst>

AppDynamics Certified Associate Administrator

<https://learn.appdynamics.com/certifications/administrator>

AppDynamics Certified Implementation Professional

<https://learn.appdynamics.com/certifications/implmenter>



To differentiate yourself and your company in an increasingly competitive market, please consider becoming AppDynamics certified.

For more information, please copy and paste this URL into a separate tab in your browser: <https://learn.appdynamics.com/certifications>

For information on specific certification tracks, please copy and paste the appropriate URLs below:

AppDynamics Certified Associate Performance Analyst:

<https://learn.appdynamics.com/certifications/performance-analyst>

AppDynamics Certified Associate Administrator:

<https://learn.appdynamics.com/certifications/administrator>

AppDynamics Certified Implementation Professional:

<https://learn.appdynamics.com/certifications/implmenter>

Monitoring the Health of Your Application (APM214)



The logo for AppDynamics University is displayed. It features a white square containing the AppDynamics logo (a stylized 'A' icon) and the word 'APPDYNAMICS' in a sans-serif font. Below this, a thin vertical line separates the logo from the word 'University' in a larger, bold, sans-serif font.

APM214 - Monitoring the Health of Your Application

Core APM 1: Essentials - Module 4

Objectives

Course

After completing this course, you will be able to:

- Describe the difference between service endpoints, data collectors and information points
- Explain how the alert and respond model works in AppDynamics
- Define metrics and baselines, and describe where to view each in the Controller
- Describe how a health rule works and where to find violations
- Explain how AppDynamics policies work
- View dashboards

Labs

In this course's labs, you will:

- Use Baselines to Understand Application Performance
- Identify Issues using Health Rules
- Explore Custom Dashboards

Service Endpoints and Information Points

Two Additional Ways of Capturing Information

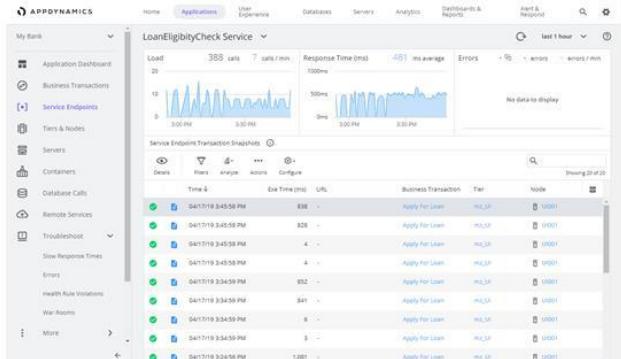
Service
Endpoints

Information
Points

Service Endpoints

Service endpoints give you key performance indicators, metrics, and associated snapshots.

They provide that information exclusively in the context of that service, omitting business transaction context or downstream performance data.



Time	Exe Time (ms)	URL	Business Transaction	Tier	Node
04/17/19 3:45:58 PM	838	-	Apply For Loan	mt_M	54601
04/17/19 3:45:58 PM	928	-	Apply For Loan	mt_M	54601
04/17/19 3:45:58 PM	4	-	Apply For Loan	mt_M	54601
04/17/19 3:45:58 PM	4	-	Apply For Loan	mt_M	54601
04/17/19 3:45:58 PM	832	-	Apply For Loan	mt_M	54601
04/17/19 3:45:58 PM	841	-	Apply For Loan	mt_M	54601
04/17/19 3:45:59 PM	8	-	Apply For Loan	mt_M	54601
04/17/19 3:45:59 PM	3	-	Apply For Loan	mt_M	54601
04/17/19 3:45:58 PM	1,081	-	Apply For Loan	mt_M	54601

A business transaction offers a view of application performance that cuts across the environment, reflecting all services that participate in fulfilling the transaction. Another way of viewing application performance, however, focuses on the performance of a particular service independently of the business transactions that use it. For example, imagine that you're in charge of the eligibility check that users have to go through to get a loan. That check involves multiple transactions, but neither individual transaction gives you the information you need. A Service Endpoint solves this by focusing on your specific service.

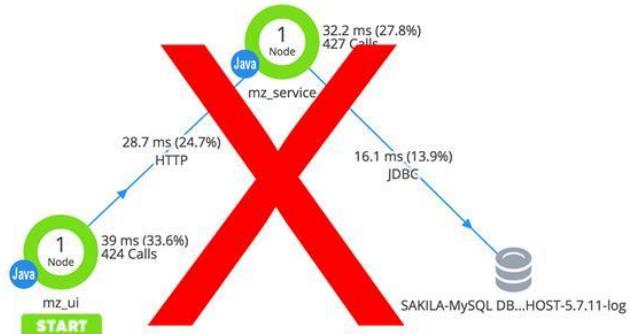
Like business transactions, service endpoints give you key performance indicators, metrics, and snapshots. However, they provide that information exclusively in the context of that service, omitting business transaction context or downstream performance data.

Service endpoints are similar to business transactions except that they report metrics only at the entry point and do not track metrics for any downstream segments. Service endpoints support the same entry point types as business transactions and you can configure them in a similar way.

Service Endpoints

Service Endpoints vs. Business Transactions

- Service Endpoints do not track downstream activity
- So you get NO FLOW MAPS
- You get no Snapshots, though the Service Endpoint may feature in a BT Snapshot



Like Business Transactions, Service Endpoints give you key performance indicators, metrics, and snapshots. However, they provide that information exclusively in the context of that service, omitting BT context or downstream performance data.

Information Points

Used to instrument a method and capture numeric information:

- System Metrics
- Custom (Business) Metrics

They have no dependence on Business Transactions, but can be invoked by one or more BTs, and capture data from those contexts.

Name	Response Time (ms)	Calls	Calls / min	Errors	Errors / min	Error %	# of Custom Metrics
Total Revenue (USD)	33	360	28	8	1	2.2	1

APPDYNAMICS

An Information Point is used to instrument a method. Information Points capture numeric data that fall into two categories: system metrics and custom (or business) metrics. Code metrics evaluate how a method is performing across the application. Business Metrics, also called Custom metrics, evaluate data from a method's parameters or return values, or anything that could be called with a getter chain. Custom metrics are aggregated – either averaged or summed over time.

For example, if you set up an Information Point to get metrics about a particular method invocation, such as credit card authentication, any time that authentication method is invoked the Information Point is triggered.

Information Point Examples

1. How many times was the method executed?
2. How long did it take to execute on average?
3. Are there errors occurring when the method is being executed? If so, how often?
4. What is the average value of the credit card total?
5. How many credit cards did my application process in a certain time period, regardless of the Business Transaction?
6. What was the average time spent processing a credit card transaction?
7. What is the current rate of rejected credit cards?



Information points can capture how a method is performing, as well as data from a method's parameters or return values to report on the performance of the business.

Notes From the Field

Information Points

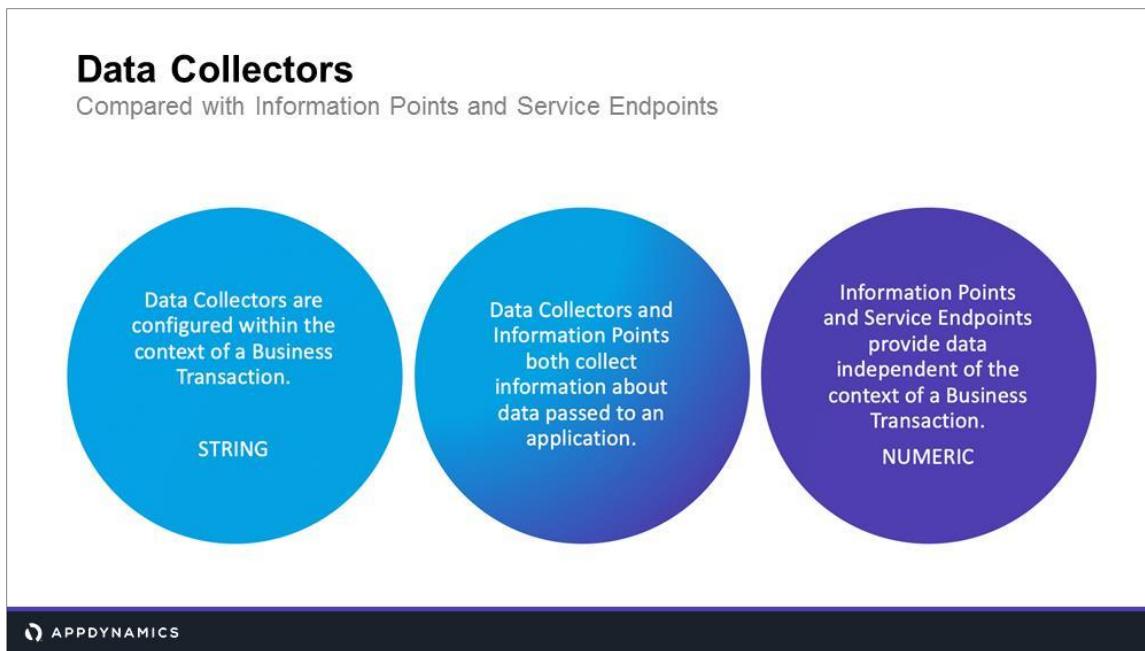
Business Problem

- A company had two separate credit card validation business transactions. They wanted to track the total number of credit card validations across the organization.

Solution

- Set up an information point to collect and aggregate information beyond the context of a single Business Transaction.





You may be wondering how the Data Collectors differ from Information Points. They are very similar as they collect information about data passed to an application, and both must be explicitly configured.

Data Collectors add STRING data to a Business Transaction Snapshot. Data from Data Collectors appear in the panels of the Transaction Snapshot call drill down - HTTP DATA, COOKIES, or USER DATA - depending on the type of the Data Collector. Data Collectors do not appear in the Metric Browser and cannot be used in Health Rules. You can filter Transaction Snapshots based on the value of a Data Collector in the Transaction Snapshot list or using the AppDynamics REST API.

Information Points aggregate numeric data about invocations of a method outside the context of any Business Transaction. Use Information Points to get metrics about method invocations across zero, one or multiple Business Transactions. Any time the method configured for the Information Point is invoked, no matter from where, the Information Point is triggered.

Like Business Transactions, Service Endpoints give you key performance indicators, metrics, and snapshots. However, like Information Points, they provide that information independent of the context of a Business Transaction or downstream performance data.

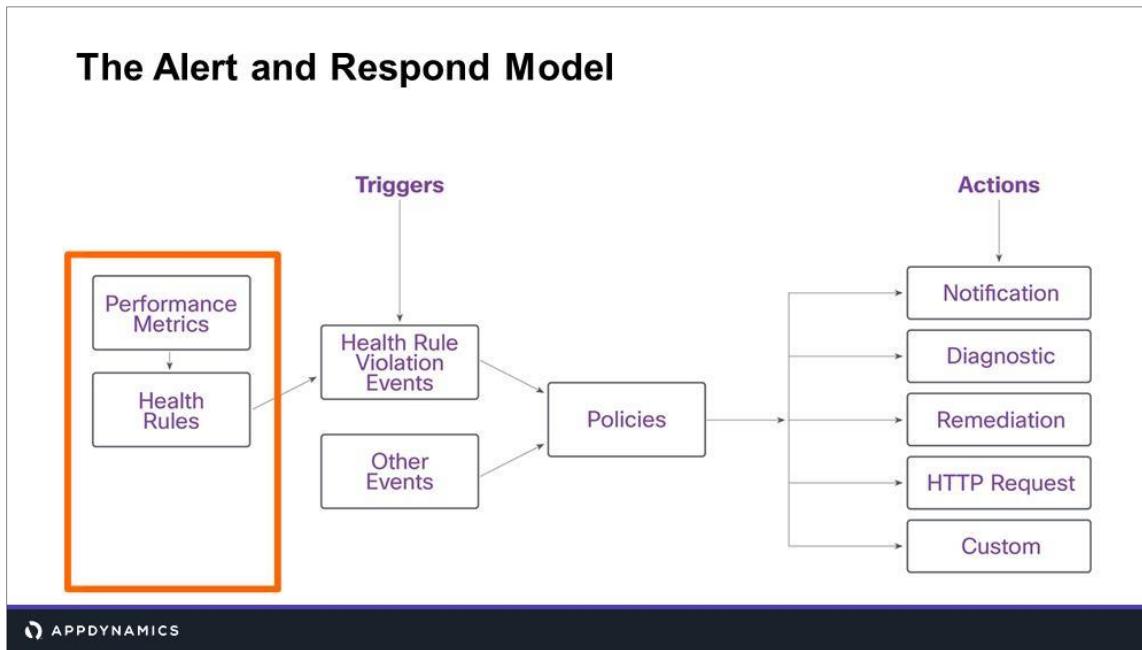
Review Question - Answer

Information Points can collect information outside the context of business transactions. They can also capture data across several business transactions.

True

False

Introduction to the Alert and Respond Model



In this module we are going to take a look at the proactive alerting mechanism, and the individual pieces that make up this mechanism.

The AppDynamics alert and respond model starts with metrics. Specifically those metrics that are critical. Once those critical metrics have been defined, health rules can be created to monitor those metrics.

A policy connects a trigger and an action. The trigger could be a health rule violation, an error, or other event. The action could be a notification, or a remediation script. When you create a policy, you're specifying both the trigger and the action.

Three pieces work together to enable this feature, so before we discuss each piece let's take a quick look at the whole to understand the relationship.

In this model, you've set up your performance thresholds and health rules. An event occurs. This could be a health rule violation or other AppDynamics event. These events serve as the trigger.

Notes From the Field

Alert and Respond

Business Problem

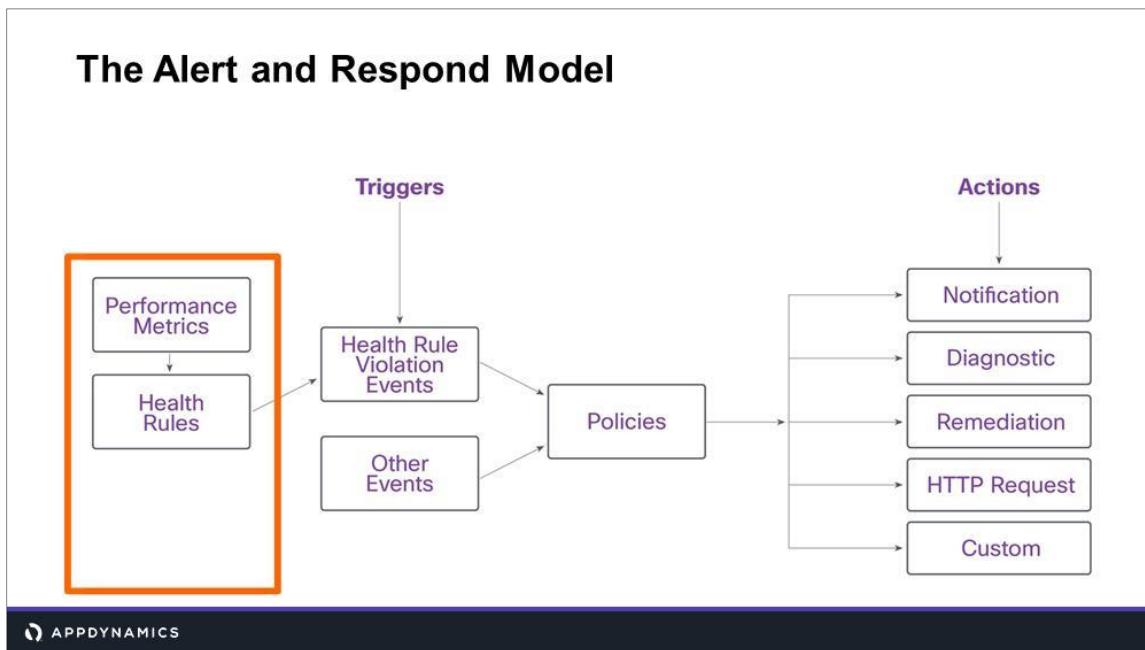
- A company currently learns about service downtime or slow response time via customer support tickets. Downtime typically lasts from 30 minutes to 2 hours

Solution

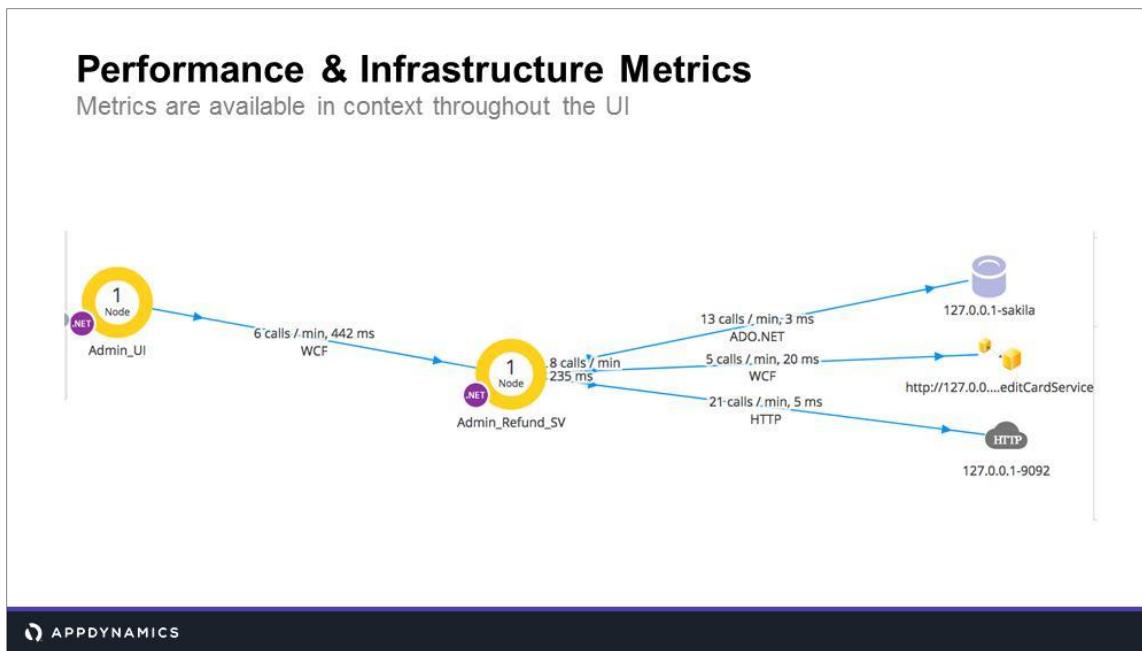
- Create a Health Rule to monitor the service. If the service experiences slow or down time, the alert kicks off a script to restart the service. If that script works, an email is sent that everything is ok. If the restart does not fix the issue, a text message is sent to the appropriate associate.



Metrics and Baselines

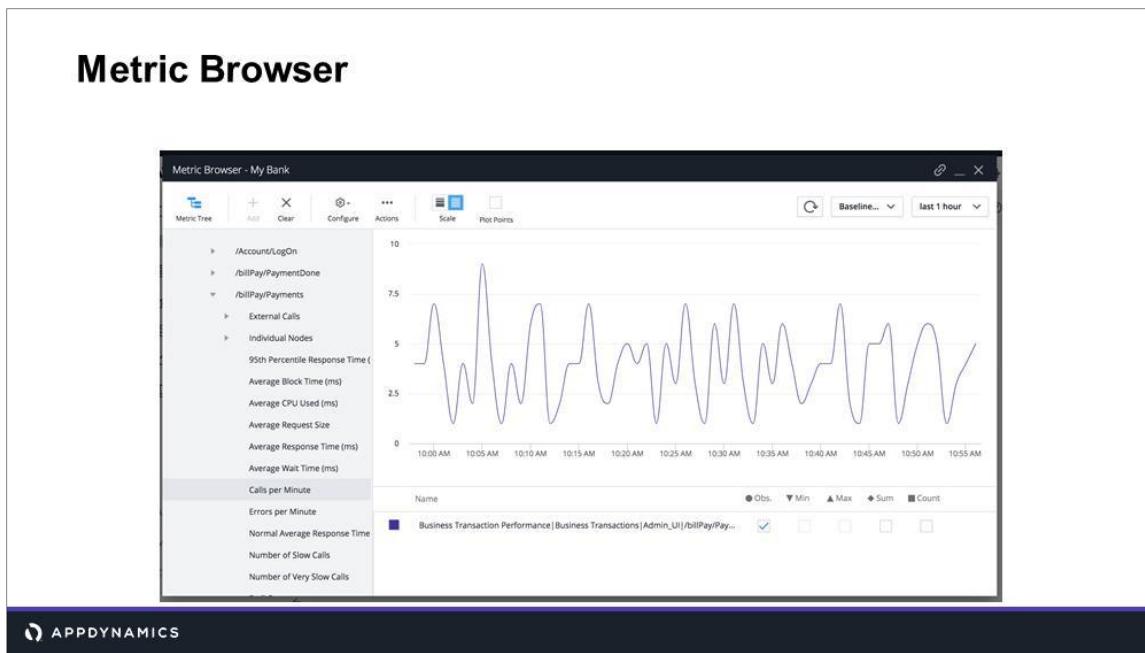


Let's start our look at this model by focusing on Performance Metrics and Health Rules.



AppDynamics reports metrics for monitored systems over time. The AppDynamics UI provides a variety of graphs and tools to help you visualize metric data so you can use it effectively for analyzing your application's performance.

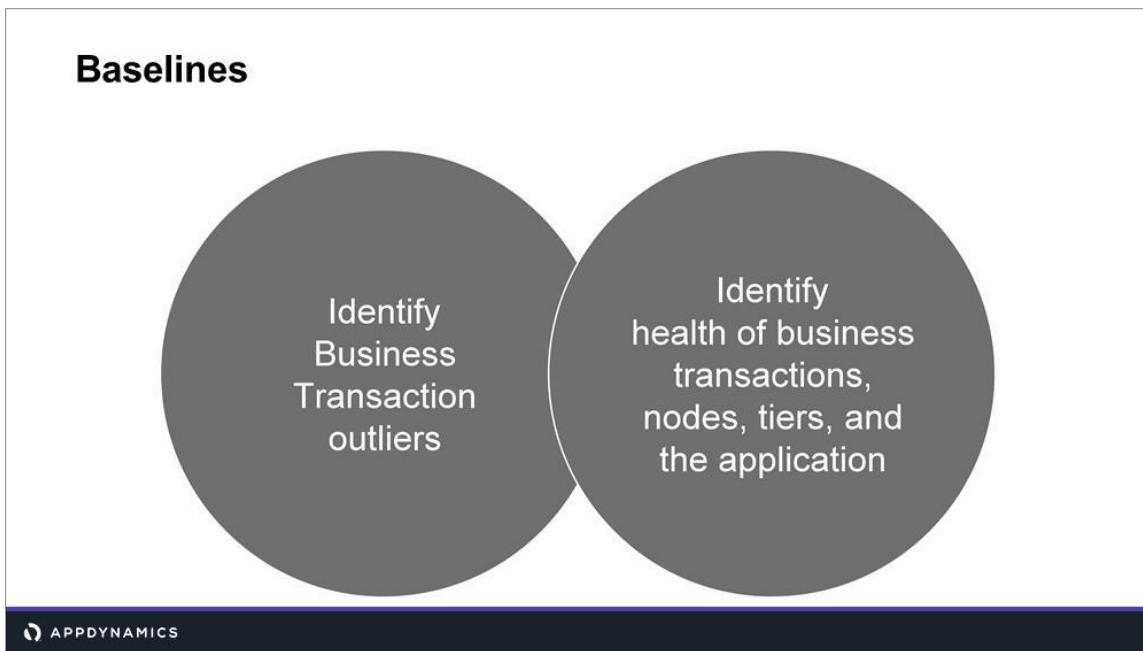
The AppDynamics user interface organizes the infrastructure metrics for your application by tiers. In addition to performance metrics for the tiers, for each application, you can see metrics gathered by the machine agent (when installed). In general, agents upload metrics to the controller once a minute at staggered intervals. The controller then rolls up the data and makes it available in the UI.



The Metric Browser is the main interface through which you inspect metrics. The Metric Browser displays all the metric values collected over the selected time period as an expandable tree.

In the browser, you can drag and drop metrics from the tree onto graphs to compare metrics and analyze patterns.

To see the metrics in each category, access the Metric Browser and expand the categories in the left pane of the browser until you get the metric you want to display. They are organized first by category, then tier, then sub-categories.



Baselines are a set of values calculated from metrics within a given time range that show historic performance patterns and averages.

AppDynamics automatically calculates dynamic baselines for your metrics, and defines what is 'normal' for each metric based on actual usage. The platform uses these baselines to identify subsequent metrics whose values fall out of this normal range.

Static thresholds that are tedious to set up and, in rapidly changing application environments can be prone to errors, are no longer needed.



You can view metrics in the metric browser against their baselines by selecting the check mark in the Base column for that metric.

In this example, notice the Call per Minute value this morning and how it relates to the calculated baseline. You can specify the time range used to calculate the baseline, as well as take the daily, weekly, and monthly trends into consideration.

Dynamic Baselines

Understanding Trends

Daily	Weekly	Monthly	None
			
Calculated for the same hour of each day for up to 31 previous days	Calculated for the same hour for the same day of the week for up to 3 previous months	Calculated for the same hour for the same day of the month for up to 12 previous months	Calculated as the unweighted average across the entire time range

 APPDYNAMICS

Dynamic baselines can be calculated on a daily, weekly, or monthly basis. Which one you use will depend on what you're trying to achieve. For example, if your application is busier in the morning than it is at night, you would want the daily trend. With the daily trend, AppDynamics would calculate the baseline by the hour. So at 9 a.m., your current application performance is compared to the average value for all data from 9 to 10 a.m. for the past 31 days. If you have more traffic on Monday rather than Sunday, you don't want to be comparing 9 am Monday with 9 am Sunday. Here, you would use weekly.

If you had an employee timesheet application that was busy on the 30th of each month, you would select the monthly trend. With this option, AppDynamics will compare metrics from the past year so that you're looking at the same day from each previous month.

There are times when the dynamic baseline isn't helpful. Let's say there was a significant downtime event two months ago. In this case, you might want to create a custom time range and exclude that week where the downtime occurred so that it doesn't affect your baseline.

For more information on Dynamic Baselines, see:

<https://docs.appdynamics.com/display/PRO45/Dynamic+Baselines>

Notes From the Field

Fixed Baselines

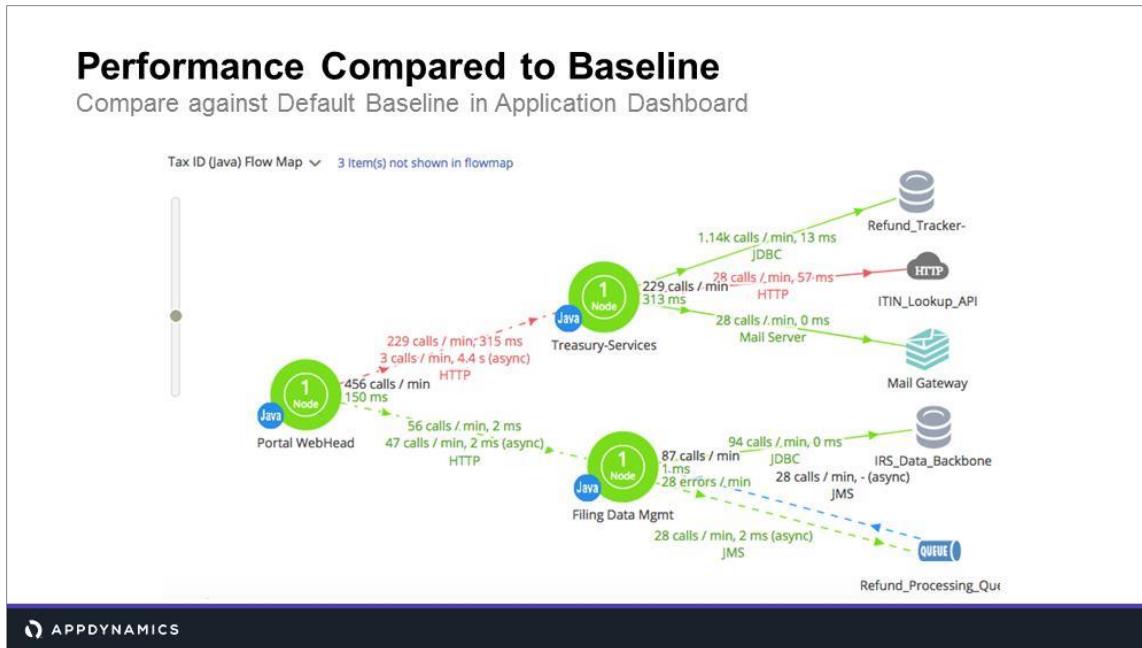
Business Problem

- A customer would frequently launch large scale email campaigns. During these campaigns, traffic would go up considerably.

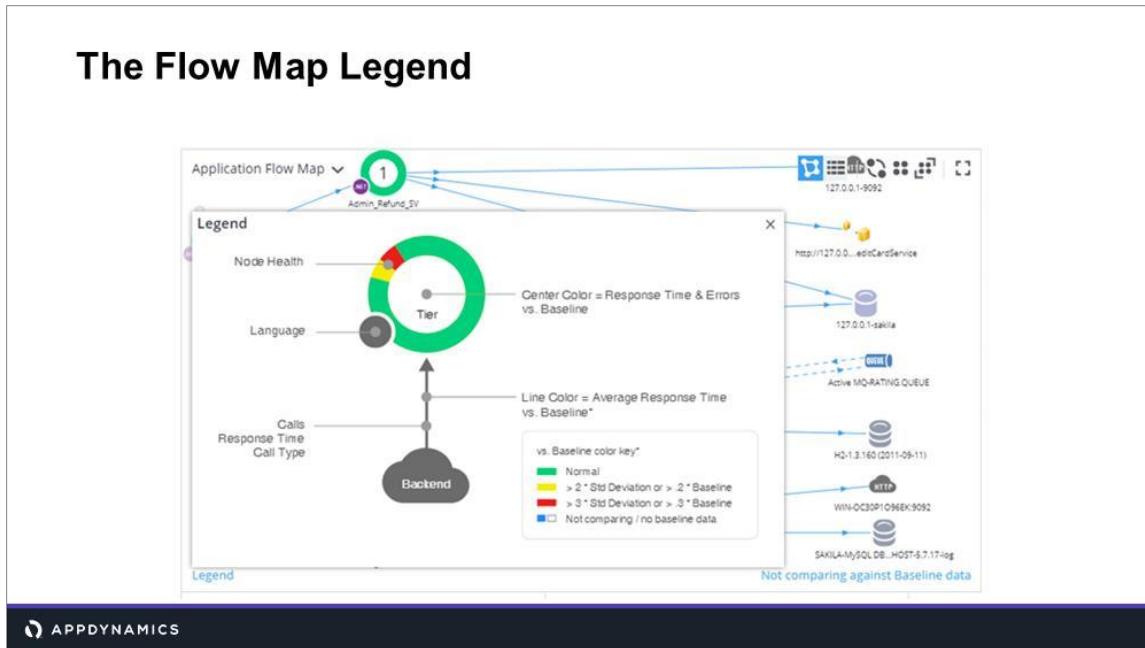
Solution

- They created a baseline based on one email campaign, and then selected that baseline during subsequent campaigns. This reduced the number of traffic-based alerts.





You can also include baseline comparison in the flow maps throughout the controller from the tier level all the way down to transaction flow maps. The outer circle continues to reflect the overall health of the tier, while the inner circle represents the health of the tier relative to the baseline.



Click the Legend link in the flow map to see what flow map elements are indicating.

Review Question - Answer

Baseline data is available in the Metric Browser.

True

False

Health Rules

- Health Rules monitor metrics and generate Events when the metrics deviate from defined thresholds.
- Health Rules have two thresholds:
Warning | **Critical**
- Default Health Rules are included, and you can create Custom Health Rules.

The screenshot shows the 'Health Rules' section of the AppDynamics interface. On the left, a list of pre-defined health rules is displayed, each with a status indicator (green checkmark). On the right, a detailed view of a specific rule is shown, titled 'Business Transaction error rate is much higher than normal'. This view includes a table showing the rule's impact across different application tiers and its execution status. The table has columns for Tier, Policy Executed, Health, Evaluation Status, and Status By Node.

Tier	Policy Executed	Health	Evaluation Status	Status By Node
Admin_UI	/Account/LogOff	Green	Green	Green
Admin_UI	/Account/LogOn	Green	Green	Green
Admin_UI	/Loans/Payments	Green	Green	Green
Admin_UI	/Payments/Fees	Green	Green	Green
mz_UI	/admin/behaviors...	Grey	Grey	Grey
mz_UI	/admin/server/ma...	Grey	Grey	Grey
Admin_UI	/bill/payment/...	Green	Green	Green

A health rule defines a condition or set of conditions in terms of metrics. The condition compares the performance metrics that AppDynamics collects with some static or dynamic threshold that you define. If performance exceeds the threshold, a health rule violation event is triggered.

AppDynamics comes pre-loaded with a set of default health rules. You can view the details of the health rules used in your application at **Alert and Respond > Health Rules**.

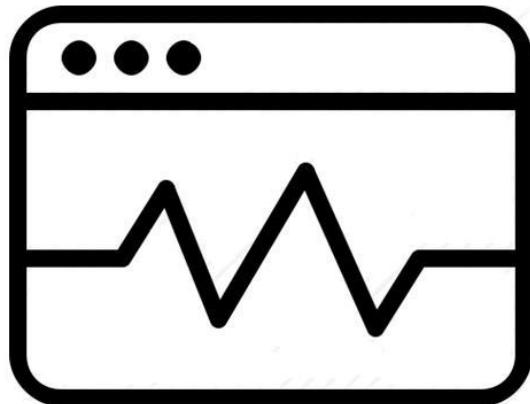
Custom health rules can be created with thresholds based on dynamic or static parameters, and different critical or warning conditions. You can set different actions for each condition.

Health rules can be applied to the whole application, or only apply to specific entities in your application. Also, health rules can be set up to run only at specific times, for a set time period, and you can specify how much recent data to use to evaluate a health rule.

Health Rule Types

Health rules can be applied to:

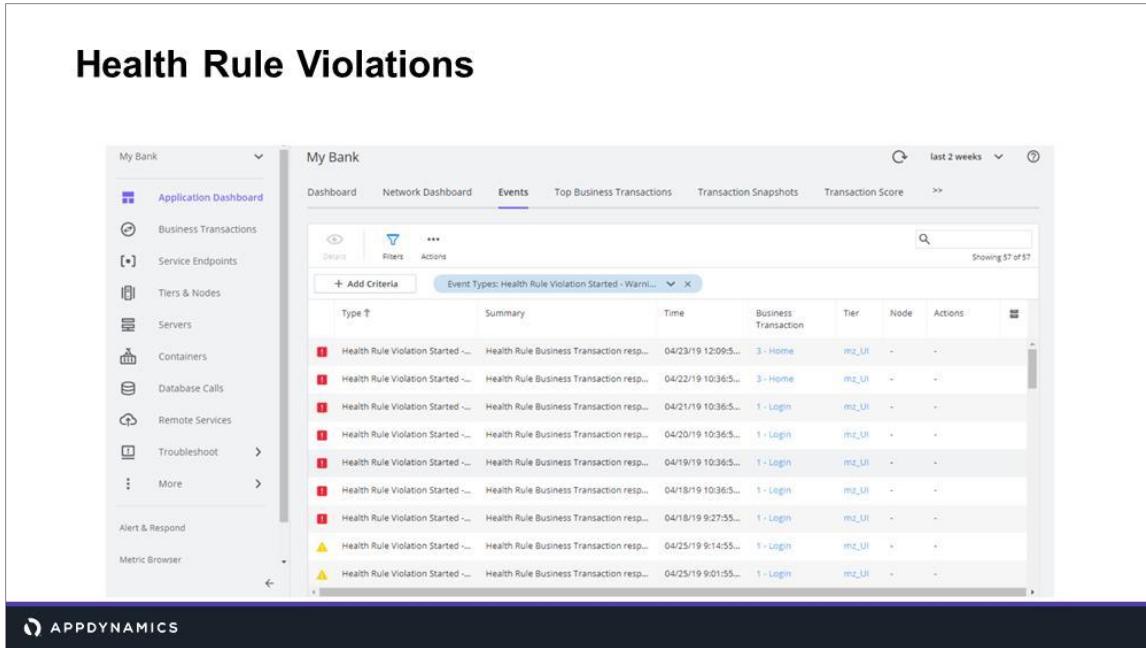
- Business Transactions
- Service Endpoints
- Information Points
- Tiers / Nodes / JMX Objects
- Pages / AJAX Requests
- Servers



 APPDYNAMICS

In AppDynamics, Health Rules can be set for any number of items, from overall application performance down to individual Business Transactions.

Health Rules can also be used in other AppDynamics monitoring products which monitor databases, browser performance and end user experience, and business analytics.

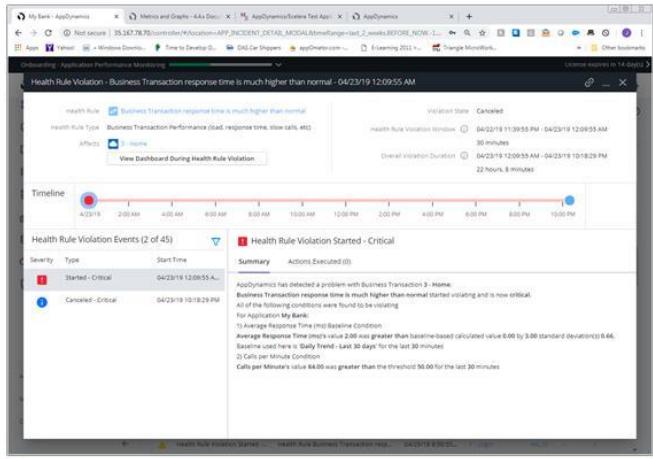


The screenshot shows the AppDynamics Application Dashboard for the 'My Bank' application. The left sidebar contains navigation links for Application Dashboard, Business Transactions, Service Endpoints, Tiers & Nodes, Servers, Containers, Database Calls, Remote Services, Troubleshoot, More, Alert & Respond, and Metric Browser. The main content area is titled 'Health Rule Violations' and shows the 'Events' tab selected. The table displays 57 events, with the first 10 rows visible. The columns include Type (with icons for error, warning, and info), Summary, Time, Business Transaction, Tier, Node, and Actions. The first 10 events are all of the 'Health Rule Violation Started - Warning' type, occurring between April 18 and 25, 2019, on various tiers and nodes.

Type	Summary	Time	Business Transaction	Tier	Node	Actions
Health Rule Violation Started - Warning	Health Rule Business Transaction resp...	04/23/19 12:09:5...	3 - Home	mz_Ui	-	-
Health Rule Violation Started - Warning	Health Rule Business Transaction resp...	04/22/19 10:36:5...	3 - Home	mz_Ui	-	-
Health Rule Violation Started - Warning	Health Rule Business Transaction resp...	04/21/19 10:36:5...	1 - Login	mz_Ui	-	-
Health Rule Violation Started - Warning	Health Rule Business Transaction resp...	04/20/19 10:36:5...	1 - Login	mz_Ui	-	-
Health Rule Violation Started - Warning	Health Rule Business Transaction resp...	04/19/19 10:36:5...	1 - Login	mz_Ui	-	-
Health Rule Violation Started - Warning	Health Rule Business Transaction resp...	04/18/19 10:36:5...	1 - Login	mz_Ui	-	-
Health Rule Violation Started - Warning	Health Rule Business Transaction resp...	04/18/19 9:27:55...	1 - Login	mz_Ui	-	-
Health Rule Violation Started - Warning	Health Rule Business Transaction resp...	04/25/19 9:14:55...	1 - Login	mz_Ui	-	-
Health Rule Violation Started - Warning	Health Rule Business Transaction resp...	04/25/19 9:01:55...	1 - Login	mz_Ui	-	-

When the conditions set in those rules are violated, the Health Rule Violation alert appears on the Events tab of the Application Dashboard.

Health Rule Violation Details



The screenshot shows the AppDynamics Application Performance Monitoring interface. A specific window titled "Health Rule Violation - Business Transaction response time is much higher than normal - 04/23/19 12:09:55 AM" is open. The window displays the following information:

- Health Rule Type:** Business Transaction Performance (avg.response time, slow calls, 404)
- Affects:** 3 items
- Violator State:** Canceled
- Health Rule Violation Window:** 04/23/19 11:39:55 PM - 04/23/19 12:09:55 AM (30 minutes)
- Overall Violation Duration:** 04/23/19 12:09:55 AM - 04/23/19 10:18:29 PM (22 hours 8 minutes)

The timeline shows two events:

Severity	Type	Start Time
Critical	Started - Critical	04/23/19 12:09:55 AM
Critical	Canceled - Critical	04/23/19 10:18:29 PM

The summary section details the violation:

Health Rule Violation Started - Critical

AppDynamics has detected a problem with Business Transaction 3 - Home. Business Transaction response time is much higher than normal started violating and is now critical. All of the following conditions were found to be violating:

- 1) Average Response Time (ms) value 2.00 was greater than baseline-based calculated value 0.00 by 3.00 (standard deviation) 0.44.
- 2) Baseline used here is Daily Trend - Last 30 days for the last 30 minutes.
- 3) Calls per Minute Condition - Calls per Minute value 84.00 was greater than the threshold 50.00 for the last 30 minutes.

You can open the violation to see more details about the nature and duration of the violation, the timeline, and any actions executed.

From this screen, you can also open the Health Rule to view its contents, or click View Dashboard During Health Rule Violation to open the dashboard for the affected business transaction at the time the violation occurred.

Review Question - Answer

What are the two thresholds of a health rule?

- A) Alert and Critical.
- B) Warning and Critical.**
- C) Alert and Warning.
- D) Baseline and Critical.

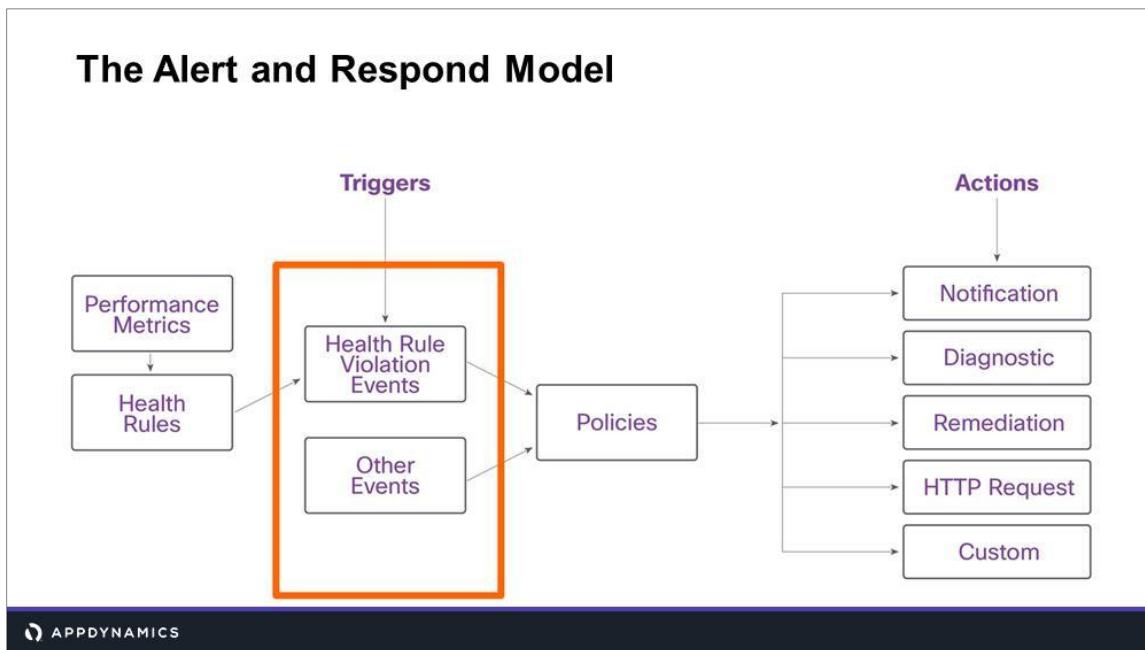
Review Question - Answer

You can only base a health rule threshold on one condition.

True

False

Events

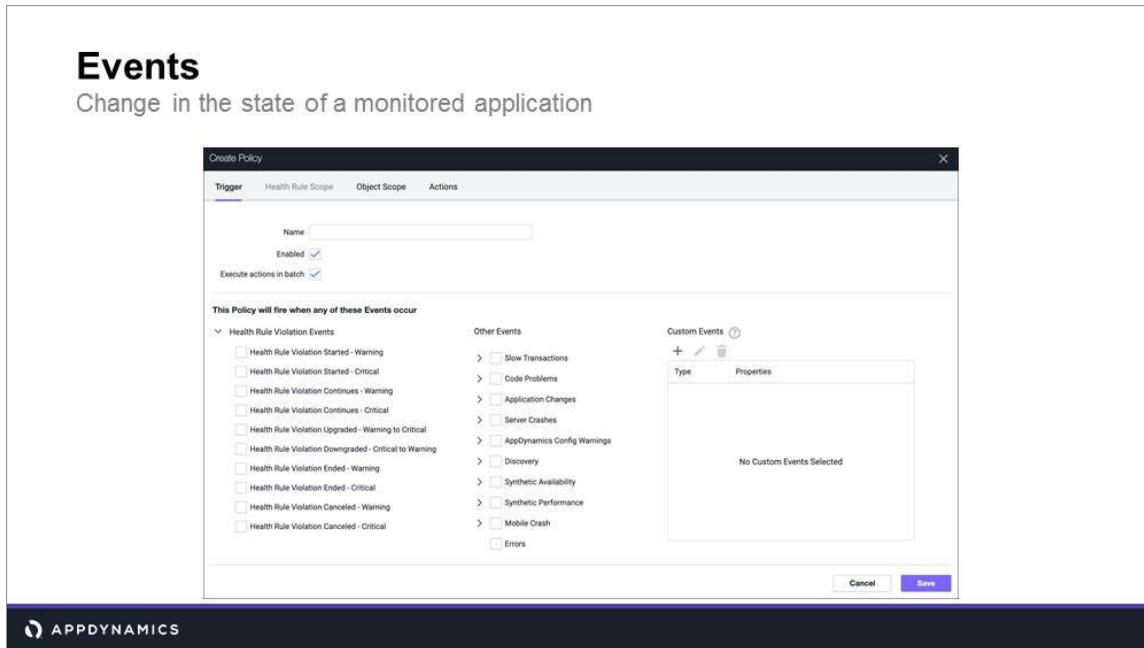


An event is a change in application state that can potentially be interesting to you. AppDynamics detects 9 different types of events automatically, and lists them in the Events tab on the application dashboard.

You can drill down into events of any type for more information. In the case of some events, such as health rule violations, you can view the state of the dashboard at the moment when the particular event occurred.

A summary of the event is available on the application dashboard, and you can access the Event List page from the Summary header, or by clicking on Events on the left hand navigation pane. You can apply filters to narrow down the list, and drill down into any of the specific events from here also.

Let's look at a few common events, and the kind of information captured for each.



The screenshot shows the 'Create Policy' dialog box with the 'Trigger' tab selected. In the 'Health Rule Scope' section, 'Health Rule Violation Events' is selected. The 'Custom Events' section is empty, showing 'No Custom Events Selected'. The 'Save' button is visible at the bottom right.

An event is a change in application state that can potentially be interesting to you. AppDynamics detects 9 different types of events automatically, and lists them in the Events tab on the application dashboard.

You can drill down into events of any type for more information. In the case of some events, such as health rule violations, you can view the state of the dashboard at the moment when the particular event occurred.

A summary of the event is available on the application dashboard, and you can access the Event List page from the Summary header, or by clicking on Events on the left hand navigation pane. You can apply filters to narrow down the list, and drill down into any of the specific events from here also.

Let's look at a few common events, and the kind of information captured for each.

Event Details - Code Problems

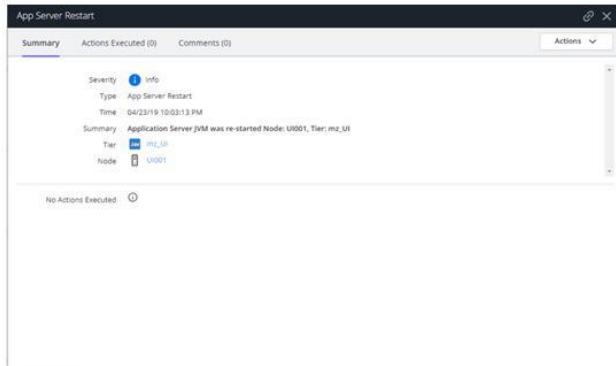
There are various types of code problem events, a typical one is a JVM Deadlock.

Affected threads are shown, with detailed troubleshooting information.

When AppDynamics detect problems with your application code, it logs the finding as a code problem event. It could be a code deadlock, or resource pool limit has been reached. Either way, you can drill down to identify the specific segment of code to be examined.

Event Details - Application Changes

- Emitted by application changes configured via Events > More Actions > Register Application Change Event
- Emitted automatically for app server restarts
- Can be used to correlate application changes with other monitoring data



© APPDYNAMICS

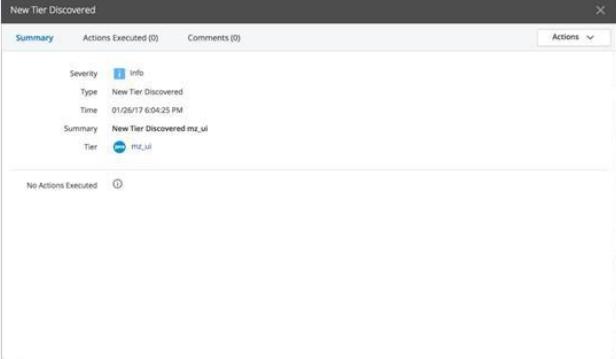
Changes in an application can provide an insight to the change in performance. The Application Changes events include the following types.

- Application Deployment - this event is manually registered. See **Monitor Application Change Events** in the AppDynamics documentation. Alternatively you can Create a Custom Event.
- Application Configuration Change - this event is manually registered.
- App Server Restarts - this event is generated when an app server is restarted.

Event Details - Discovery

Event Discovery Types:

- New Application Discovered
- New Tier Discovered
- New Node Discovered
- New Machine Discovered
- New Business Transaction Discovered



The screenshot shows a modal window titled 'New Tier Discovered'. It has tabs for 'Summary', 'Actions Executed (0)', and 'Comments (0)'. The 'Actions' dropdown is set to 'Actions'. The summary section displays the following details:

- Severity: Info
- Type: New Tier Discovered
- Time: 01/26/17 6:04:25 PM
- Summary: New Tier Discovered mz_zul
- Tier: mz_zul

The 'Actions Executed' section is empty.

Event Discovery Types:

- New Application Discovered
- New Tier Discovered
- New Node Discovered
- New Machine Discovered
- New Business Transaction Discovered

Notes From the Field

New Code Release Event

Business Problem

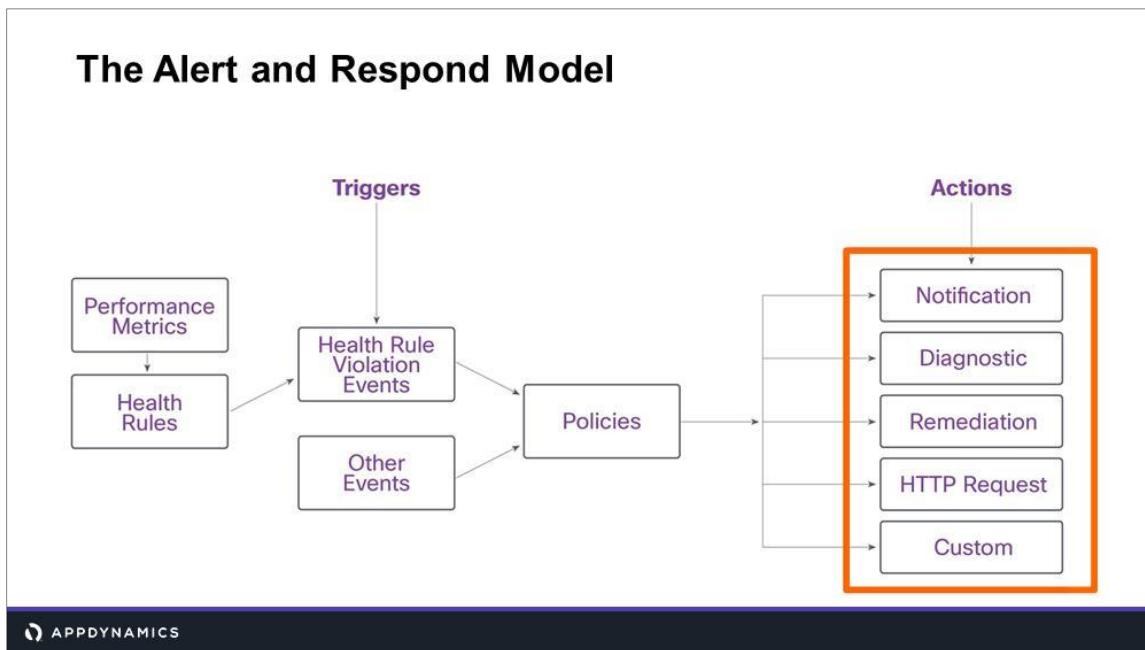
- AppDynamics users often do not have insight into when new code is pushed live, or what that code might affect.

Solution

- Whenever any new code is pushed into production, you can include an AppDynamics REST API call to create an application deployment event, detailing any bug fixes or code changes. That way, anyone looking at post-code release issues will be able to see that new code was pushed live, and what it affected



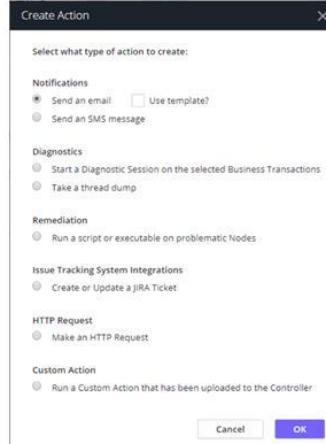
Actions and Policies



We're going to come back to policies. Let's look at the actions you can set up to respond to triggering events.

Actions

- Predefined, reusable, automated response to an event
- Send a notification
- Start a diagnostic session or execute a thread dump
- Run a local script for remediation
- Create an HTTP request
- Set a custom action

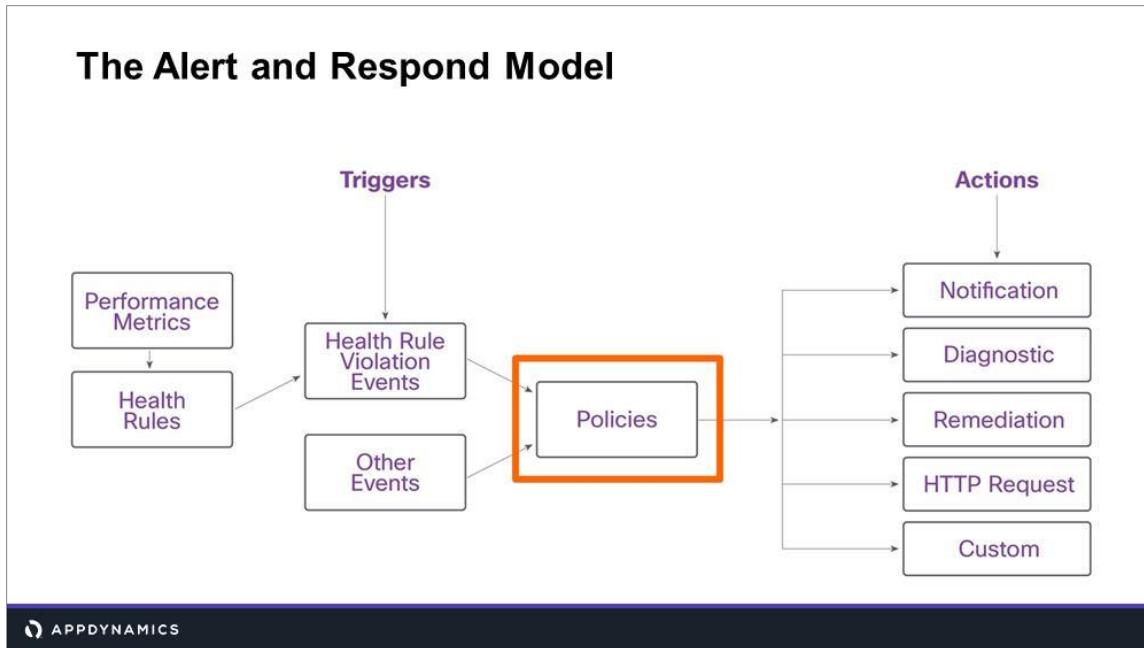


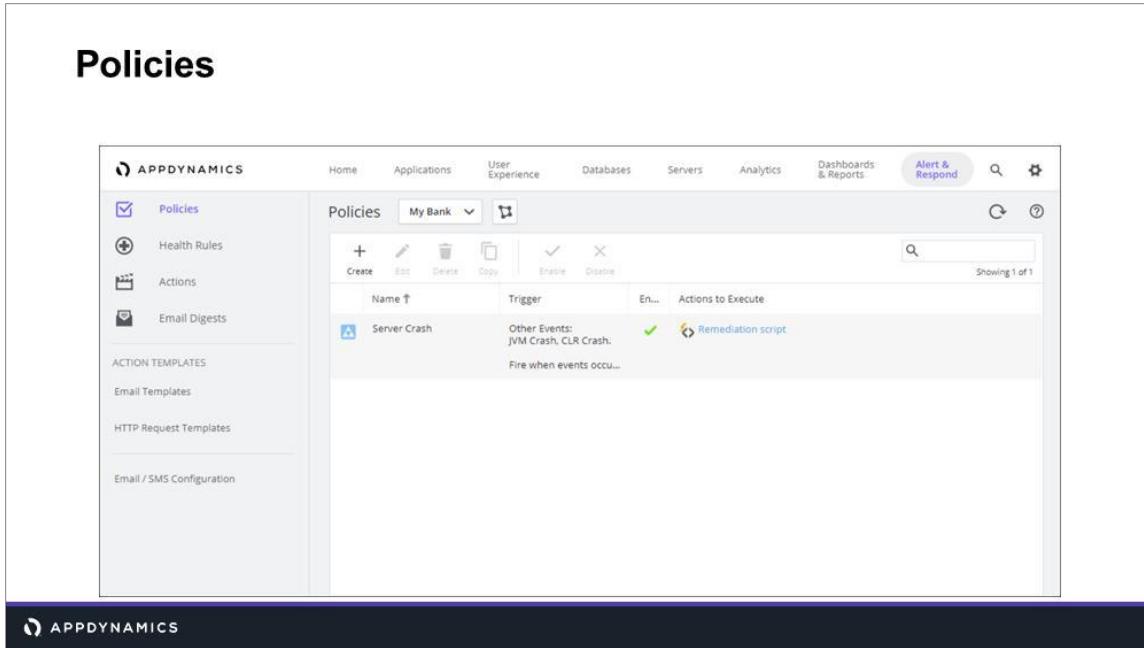
APPDYNAMICS

An action is a predefined, reusable, automated response to an event. AppDynamics allows for a number of different actions. A common action is to send a notification by email or SMS to a recipient list. The text of the notification is automatically generated by the event that triggered the action.

Another common action is to start a Diagnostic Session to collect snapshots or direct the agent to take a thread dump for a specified number of samples (maximum of 50) with each sample lasting for a specified number of milliseconds (maximum of 500 ms).

Beyond these typical responses, you could also use an action to run local scripts in a node. The script executes on the machine from which it was invoked or on the machine specified by the remediation action configuration.



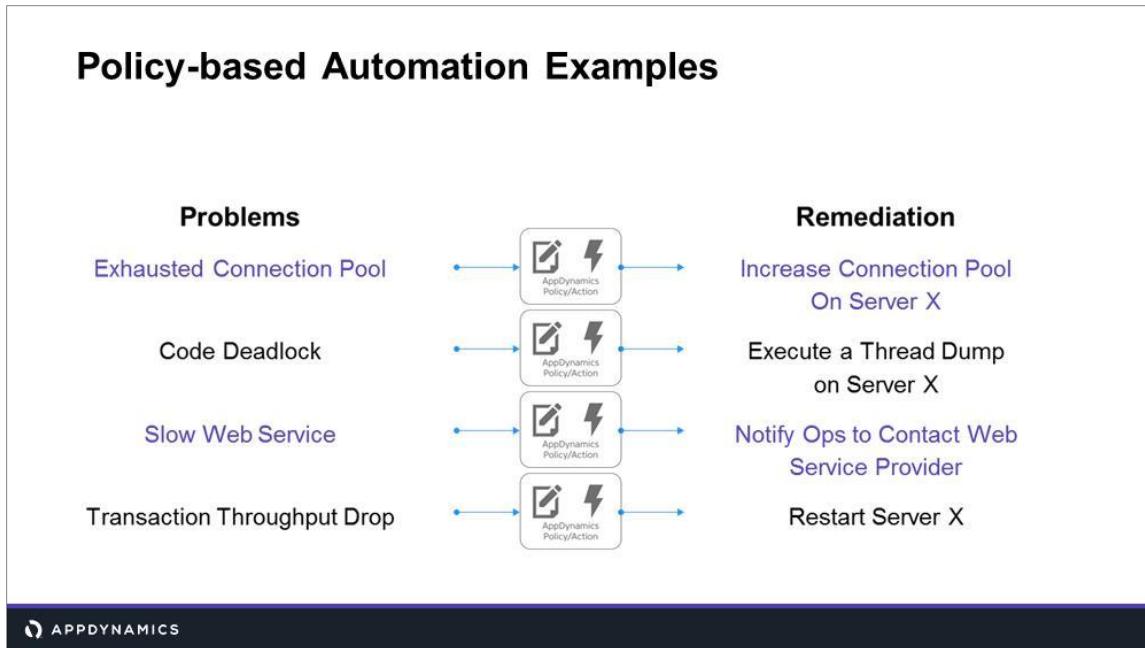


The screenshot shows the AppDynamics Policies interface. The left sidebar has a checked 'Policies' option. The main area shows a table with one row for 'Server Crash'. The columns are 'Name' (Server Crash), 'Trigger' (Other Events: JVM Crash, CLR Crash), 'En...' (status), and 'Actions to Execute' (Remediation script). The table has a footer 'Showing 1 of 1'.

Ultimately, policies let you connect triggers/events to actions. Policies give you the ability to anticipate problems and take actions to address those problems before they cause a severe slowdown or outage.

The action is triggered automatically as soon as the event occurs.

Because the definition of health rules is separate from the definition of actions, and both health rules and actions can be very precisely defined, you can take different actions for breaching the same thresholds based on context, for example, which tier or node the violation occurred in.



Policies can be used to respond to a variety of situations with appropriate actions. Here are some of the conditions you can set policies for, and typical remediation you can use.

Review Question - Answer

A policy has three main parts: a trigger, an action, and a log.

True

False



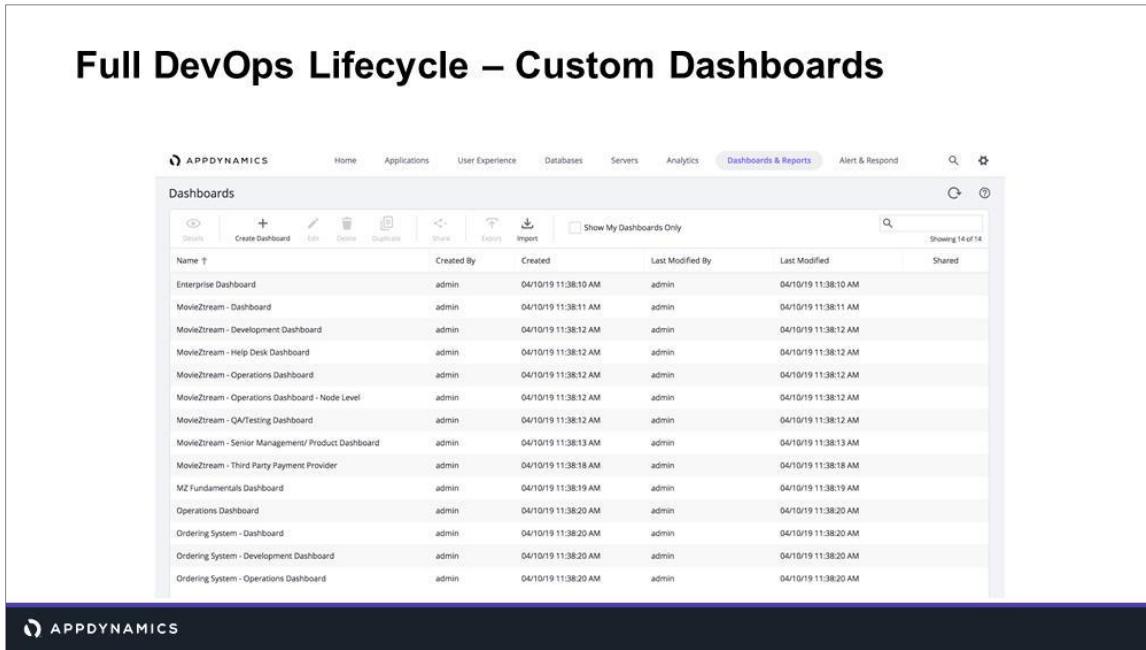
No log.

Review Question - Answer

Which of the following are available actions for a policy?

- A) Send a notification.
- B) Start a diagnostic session.
- C) Run a report.
- D) Execute a thread dump.
- E) Run a local script for remediation.
- F) Create an HTTP request.

Custom Dashboards

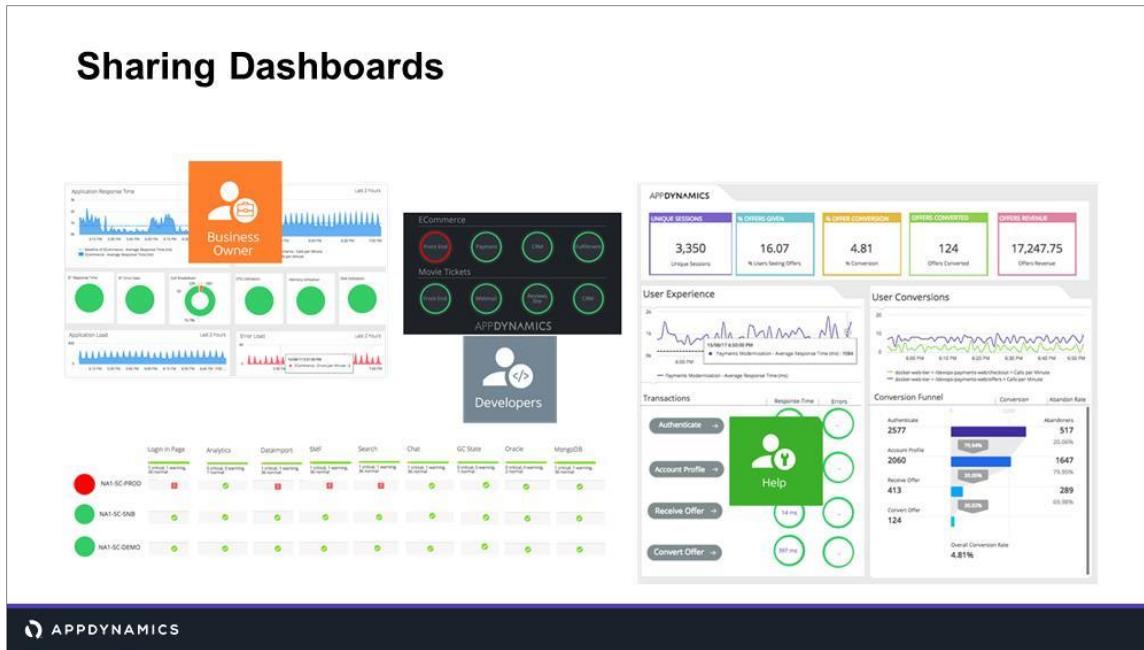


The screenshot shows the AppDynamics interface with the title "Full DevOps Lifecycle – Custom Dashboards". The top navigation bar includes Home, Applications, User Experience, Databases, Servers, Analytics, Dashboards & Reports (which is selected and highlighted in blue), and Alert & Respond. Below the navigation is a search bar and a "Show My Dashboards Only" checkbox. The main content is a table titled "Dashboards" with 14 rows, showing details for each dashboard. The columns are: Name, Created By, Created, Last Modified By, Last Modified, and Shared. The data includes various dashboards for Enterprise, MovieZstream, and Ordering System, all created and last modified by "admin" on 04/10/19 at 11:38:10 AM.

Name	Created By	Created	Last Modified By	Last Modified	Shared
Enterprise Dashboard	admin	04/10/19 11:38:10 AM	admin	04/10/19 11:38:10 AM	
MovieZstream - Dashboard	admin	04/10/19 11:38:11 AM	admin	04/10/19 11:38:11 AM	
MovieZstream - Development Dashboard	admin	04/10/19 11:38:12 AM	admin	04/10/19 11:38:12 AM	
MovieZstream - Help Desk Dashboard	admin	04/10/19 11:38:12 AM	admin	04/10/19 11:38:12 AM	
MovieZstream - Operations Dashboard	admin	04/10/19 11:38:12 AM	admin	04/10/19 11:38:12 AM	
MovieZstream - Operations Dashboard - Node Level	admin	04/10/19 11:38:12 AM	admin	04/10/19 11:38:12 AM	
MovieZstream - QA/Testing Dashboard	admin	04/10/19 11:38:12 AM	admin	04/10/19 11:38:12 AM	
MovieZstream - Senior Management/ Product Dashboard	admin	04/10/19 11:38:13 AM	admin	04/10/19 11:38:13 AM	
MovieZstream - Third Party Payment Provider	admin	04/10/19 11:38:18 AM	admin	04/10/19 11:38:18 AM	
M2 Fundamentals Dashboard	admin	04/10/19 11:38:19 AM	admin	04/10/19 11:38:19 AM	
Operations Dashboard	admin	04/10/19 11:38:20 AM	admin	04/10/19 11:38:20 AM	
Ordering System - Dashboard	admin	04/10/19 11:38:20 AM	admin	04/10/19 11:38:20 AM	
Ordering System - Development Dashboard	admin	04/10/19 11:38:20 AM	admin	04/10/19 11:38:20 AM	
Ordering System - Operations Dashboard	admin	04/10/19 11:38:20 AM	admin	04/10/19 11:38:20 AM	

The ability to create Custom Dashboards is one of the most powerful features of AppDynamics.

You can group monitoring metrics that are relevant to you in one consolidated dashboard.



You can share dashboard data within your organization, and even with vendors outside of your organization.

And dashboards are read only. All could be internal and external.

Custom Dashboards

Present relevant data for each role

 Development Interested in the data that helps improve the code	 Administration Interested in all high-level data across all applications	 Operations Interested in the data that identifies possible server issues
---	---	---

 APPDYNAMICS

Depending on your role, you may have a specific set of metrics you are interested in tracking. If you work in Operations, you are interested in metrics that deal with server performance; if your colleague is a developer, he or she may be more interested in comparing metrics that have to do with performance of the application code.

AppDynamics offers the ability to create custom dashboards, so everyone can have a dashboard that presents all the data that's relevant to their responsibilities.

Custom Dashboards Features

- Your branding
- All HTML 5 views
- Share with others via URL, including non-AppDynamics users
- Integrate with other monitoring tools via iFrame widget
- Create widgets from additional AppDynamics monitoring tools (Analytics, Database, and Server)
- Visual alerting with status lights



Custom dashboards have several features that enable you to do more with AppDynamics. You can use your company color and logos to custom-brand the dashboard. They are HTML5 so you can check your application's health from anywhere, at any time and with any device.

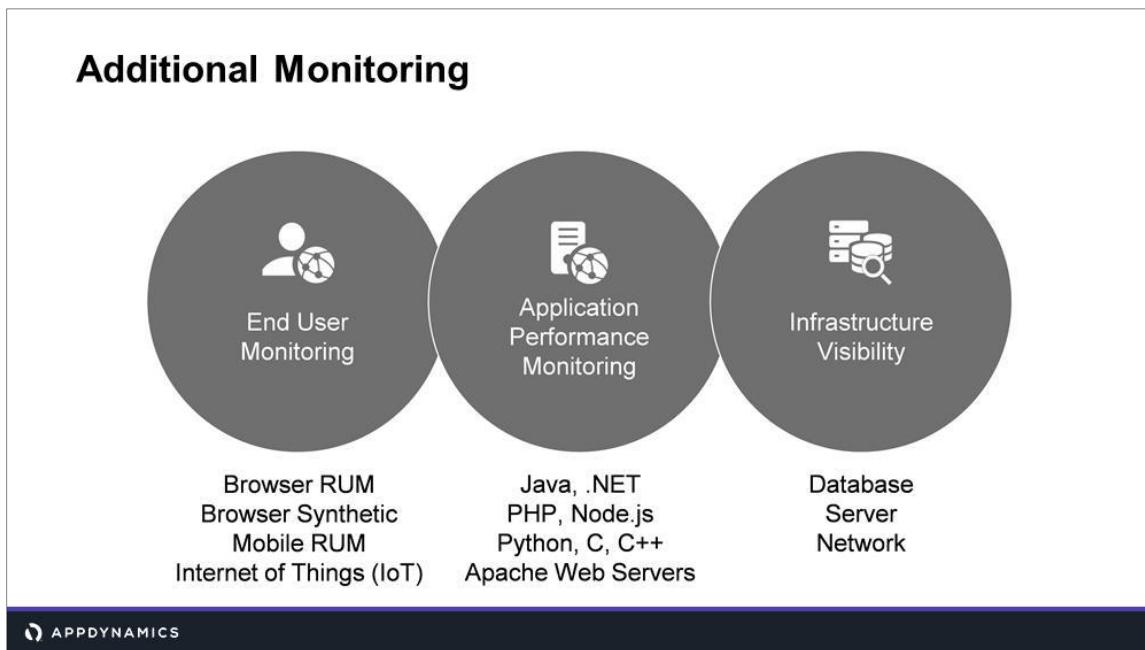
By default only users logged into the controller with the necessary permissions can view a Custom Dashboard. However, if you share a dashboard, it becomes available to anyone possessing the shared URL, regardless of whether they can log into AppDynamics or not. You can also stop sharing a dashboard if you don't wish to make data available to a wider audience indefinitely.

Further, with the use of the iFrame widget, you can embed HTML content or insert data from other monitoring applications and make this custom dashboard the single pane of glass to view all available application performance information.

Custom dashboards can present baseline information along with actual metrics, so it's easy to compare and detect anomalies, and the status light feature presents health rule status visually so you know the health status right away. And if you detect anomalies, you can drill down to the root cause of the problem right from the dashboard.

Additionally, you can import & export dashboards, so you can use the same dashboard in dev, QA, and production environments without creating each one separately.

Additional Monitoring



Additional monitoring tools exist for browser, mobile and IoT, and for database, server, and network monitoring.

End User Monitoring

Browser Real-User Monitoring

- Identify how your application is performing for real users across different pages, devices, and geographies
- Troubleshoot using snapshots and use correlated APM Business Transactions for additional analysis

Browser Synthetic Monitoring

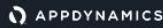
- Proactively test your key user journeys to make sure they work 24/7 using our global network of test nodes, or your own internal nodes

Mobile Real-User Monitoring

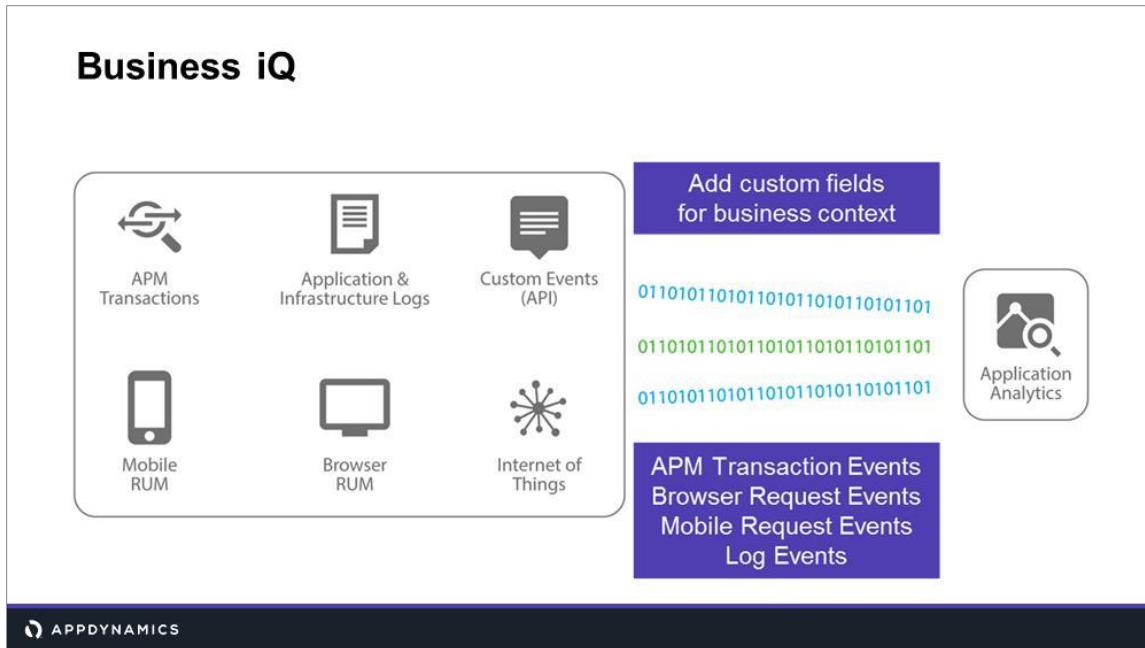
- Connect mobile app performance to your real users' behavior or experience
- Troubleshoot crashes with detailed analysis down to the stack trace
- Link directly to correlated APM snapshots, allowing for true end-to-end analysis

Internet of Things (IoT)

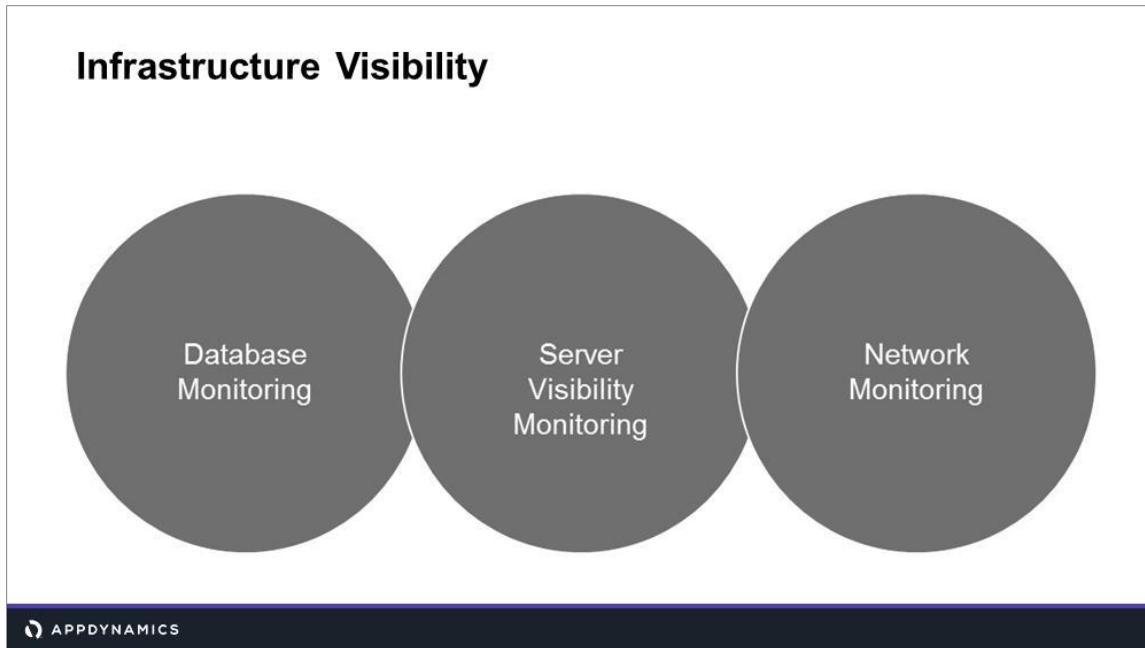
- Monitor device connectivity, network requests, as well as errors and exceptions
- Use predefined and custom widgets to visualize business value impact data



End user monitoring gives you insight into application performance from the perspective of the end user.



AppDynamics agents collect business data, enabling operational visibility.



There are three components that fall under Infrastructure Visibility: Database Monitoring, Server Visibility Monitoring, and Network Monitoring.

Database Monitoring

Monitor your databases and database servers for vital information

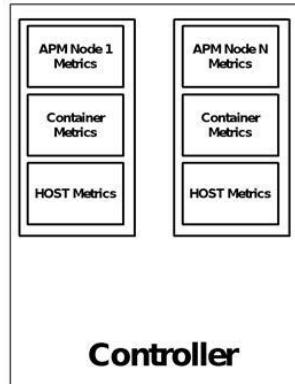
- DB Monitoring examines your databases 24/7 automatically and can alert you when unusual or critical situations arise
- View the current performance of connected sessions
- Many commonly used databases supported
- DB Monitoring Agent is a standalone java program that can report on typically 25 DB Instances
- DB Monitoring records baselines of database metrics allowing for review of performance outliers
- Discrete windows for Live View, Query Analysis, Execution Plans, connected Clients, active Sessions, active Users, Database Object Browser and more
- Ability to view current performance reports: Wait State, Top Activity, Time Comparison, and Query Wait State Reports
- DB Monitoring offers Policies, Health Rules and Actions just like in core APM

Server Monitoring

End-to-end server visibility

- Container Visibility – 3-way drill down – APM Metrics, Container Metrics, Host Metrics
- Tier Metric Correlator – identify load and performance anomalies across all nodes in a tier
- Service Availability Monitoring

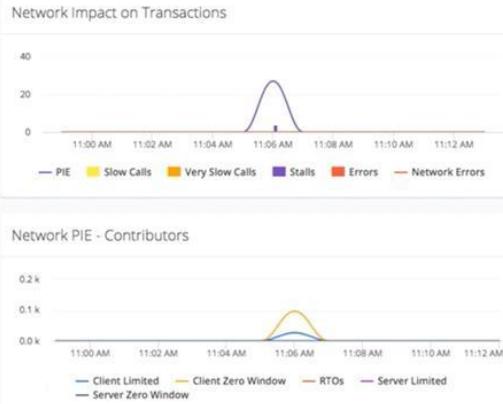
Docker Visibility



Network Monitoring

Is the network causing performance issues in my application?

- Pinpoint the traffic flows, individual nodes, and transport connections where network or application/network issues are occurring
- Collect detailed metrics that show how the network is performing and how well your application uses network connections and resources
- Collect and report targeted troubleshooting information to network, IT, DevOps, and other teams



APPDYNAMICS

With Network Monitoring, you can:

- Answer the key question quickly and easily: *Is the network to blame for any performance issues in my application?*
- Identify the root cause of performance issues in the application, the network, or in application/network interactions (i.e., how an application or server utilizes the network).
- Pinpoint the traffic flows, individual nodes, and transport connections where network or application/network issues are occurring.
- Collect detailed metrics that show how the network is performing and how well your application uses network connections and resources.
- Collect and report targeted troubleshooting information to network, IT, DevOps, and other teams.
- Drill down to contributing factors. For example: Retransmission Timeouts (RTOs) are a sign of network packet loss, which results in retransmission of data when the TCP retransmission timer expires (Timer to make sure data is ACK'ed). Typically this timer varies from 200ms-3 sec by default (different for each operating systems and their versions). It causes severe performance degradations as considerable amount of time gets wasted to resend the lost data. TCP falls back to the "Slow Start" phase impacting the performance even more.

What You Learned

- The difference between service endpoints, data collectors and information points
- How the alert and respond model works in AppDynamics
- The definition of a metric and a baseline, and where to view each in the Controller
- How a health rule works and where to find violations
- How AppDynamics policies work
- How to view dashboards

Differentiate yourself and your company with AppDynamics Certification

APPDYNAMICS | Certification Program

General Certification Information

<https://learn.appdynamics.com/certifications>

AppDynamics Certified Associate Performance Analyst

<https://learn.appdynamics.com/certifications/performance-analyst>

AppDynamics Certified Associate Administrator

<https://learn.appdynamics.com/certifications/administrator>

AppDynamics Certified Implementation Professional

<https://learn.appdynamics.com/certifications/implmenter>



To differentiate yourself and your company in an increasingly competitive market, please consider becoming AppDynamics certified.

For more information, please copy and paste this URL into a separate tab in your browser: <https://learn.appdynamics.com/certifications>

For information on specific certification tracks, please copy and paste the appropriate URLs below:

AppDynamics Certified Associate Performance Analyst:

<https://learn.appdynamics.com/certifications/performance-analyst>

AppDynamics Certified Associate Administrator:

<https://learn.appdynamics.com/certifications/administrator>

AppDynamics Certified Implementation Professional:

<https://learn.appdynamics.com/certifications/implmenter>

AppDynamics Strategy and Introduction to Business Transaction Discovery (APM221)



University

APM221 - AppDynamics Strategy and Introduction to Business Transaction Discovery

Core APM II: Advanced - Module 1

Objectives

Course

After completing this course, you will be able to:

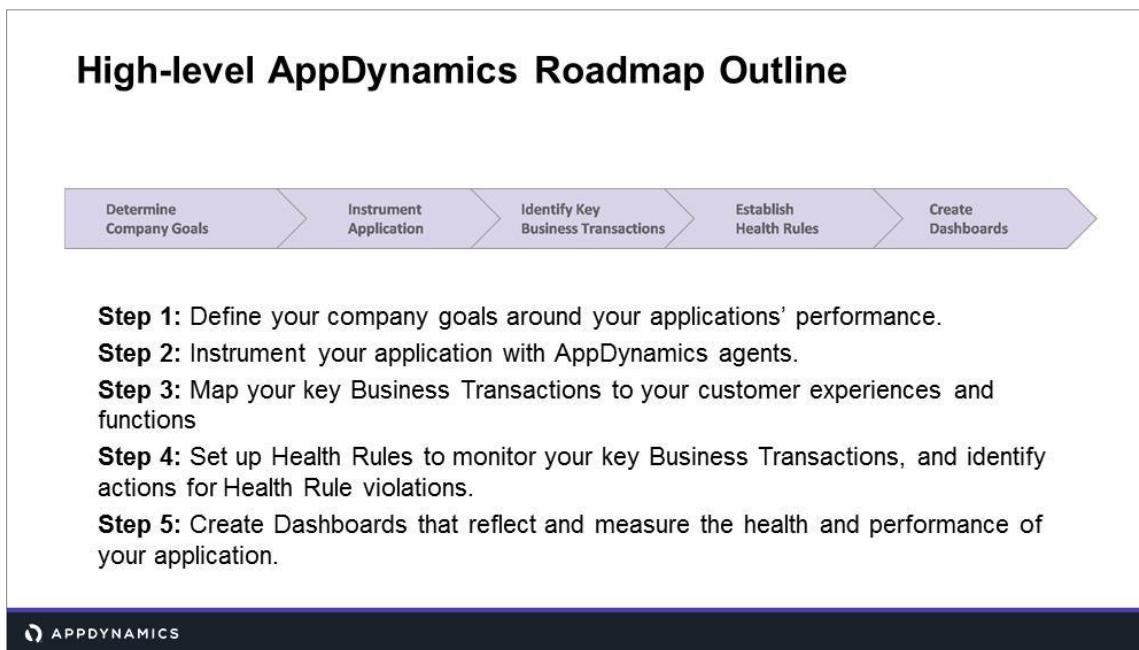
- Define your company goals and the key steps for implementing AppDynamics
- Demonstrate the significance of prioritizing Business Transactions
- Explain the principle of BT Entry and Exit Points
- Describe how Controller communication works and the technique of Tag and Follow
- Describe how Auto Discovery works and the range of technologies automatically discovered
- Change default URL-based discovery rules
- Demonstrate the process of selecting Business Transactions by hand
- Create Custom Match Rules to control BT discovery
- Use Scopes to control application of BT Rules

Labs

In this course's labs, you will:

- Identify your Key Transactions
- Configure Business Transactions using Automatic Discovery Rules
- Use BT Lockdown to select Business Transactions by Hand
- Register Key Business Transactions and Custom Match Rules

Steps to an AppDynamics Implementation



- **Step 1:** What is your APM Strategy? What do you want to accomplish with AppDynamics?
- **Step 2:** Instrument your application. Validate that the flow maps match your expected architecture and design.
- **Step 3:** Identify KEY Business Transactions and map them to your customer experiences and functions. It is critical to understand and identify the most important transactions that affect your customers' experience.
- **Step 4:** Set up SLA based Health Rules on key Business Transactions. Ensure you are fully aware of the customer experience at all times. For each Health Rule, identify actions to be taken. What do you do when you get a warning alert? A critical alert? What are your workflows? This can and should include the ability to potentially self-heal your applications if applicable.
- **Step 5:** Show people. Create dashboards that reflect and measure the health and performance of your application.

Notes From the Field

Preparing for Implementation

Business Problem

MovieZstream asks you to implement AppDynamics for them. How are you going to approach it?

Solution

- Determine budget (setting up complex software isn't free)
- Identify who in your org needs to be involved?
 - Which teams will benefit from using AppD?
- Looking at the system, define the Top transactions
 - Don't try to monitor everything
 - Prioritize what matters most to customers



© APPDYNAMICS

- Customer has purchased AppD. Wants to implement it now.
- Is there budget to get it set up properly? It's a sophisticated tool and requires configuration to get the most value out of it.
- Who needs to be in the loop on this project? Business, IT (Dev, Ops, DBA, Support etc).
- Now explore the interface of this system and figure out what the key functionality is from the CUSTOMER perspective that you will need to end up monitoring.
- Focus on your Top 20 transactions. When a customer phones up with a complaint that "it's not working", you want to know what "it" is, e.g., cannot login, cannot rent a movie.
- Don't try to monitor everything, you don't need to and it's too much work
- Work towards prioritizing the transactions you've identified, so you also ultimately name your BTs according to which matter most.

Discussion

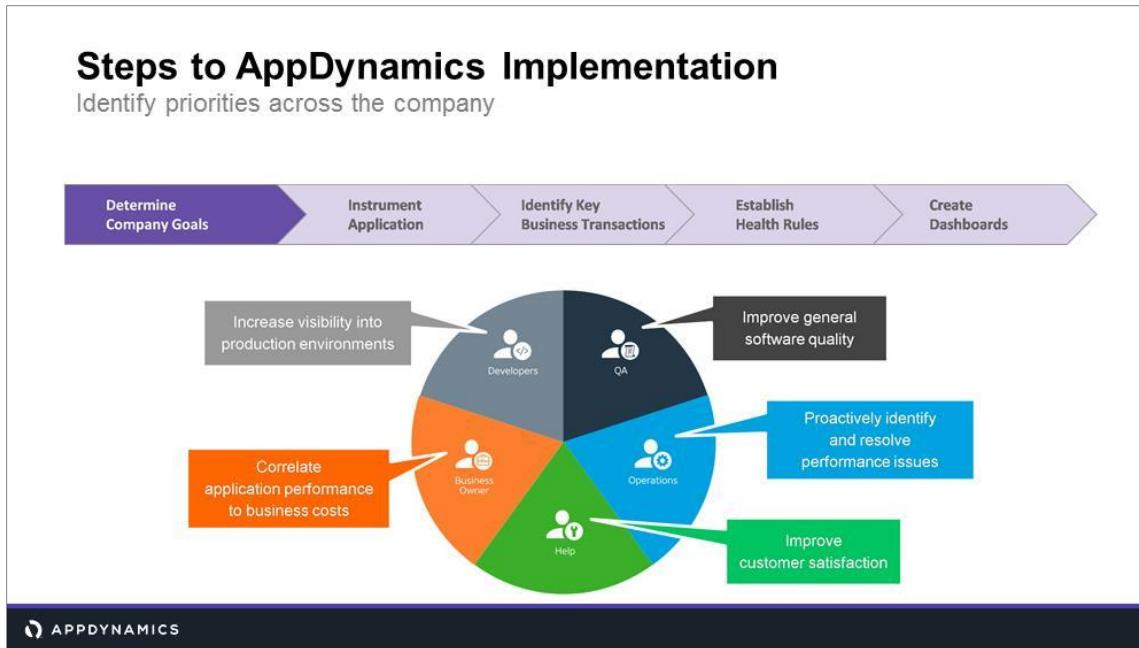
In Your Last Outage...

Key IT Metrics	How AppDynamics Can Help
What is your current Mean Time to Detect?	Focus on Health Rules and Alerts
What is your current Mean Time to Diagnose?	Focus on Snapshots
What is your current Mean Time to Repair?	Focus on Runbook Automation via Policies and Alerts
How much time do you spend on average in a War Room? What is the cost?	Reduce Number of people Number of hours/incident Number of incidents/month Cost/person

 APPDYNAMICS

It's far better to have a monitoring tool that can spot problems as they occur, then take steps to capture relevant detail and even potentially perform some kind of remediation activity to help ensure your customers aren't impacted.

- Do you know what your current Mean Time to Detect a problem is? How about the Mean Time To Diagnose a problem? And how long do you need to fix problems?
- It's usually pretty hard to give clear answers on this, but with AppDynamics you get measurements about what the problems are, when they occurred, along with the detail to help you work out what is causing them.
- Downtime is expensive, right? The quicker you can reduce your downtime, the less money you will lose - both in terms of short-term revenue impacted as well as longer term reputation damage.
- War rooms are typically expensive and time-consuming, as you often have to gather experts from various areas of your IT team to then figure out what has happened and why. That all costs you money, time and hassle.



Finding out what goals different teams in your organization have should be considered a driving force in how you approach setting up your AppDynamics monitoring solution.

The needs of Developers naturally differ from those of Helpdesk, as they do from those of Operations, and the overall company goals must also be used to steer the implementation and maximize the value you get from adopting AppDynamics.

Discussion

What are the priorities for YOUR company and YOUR role in using AppDynamics?



For this you will need to:

- Quantify application performance
- Gauge how well the application is running over time
- Establish “healthy” vs “unhealthy” performance
- Compare baselines and thresholds of performance
- Interpret how this impacts your company’s bottom line

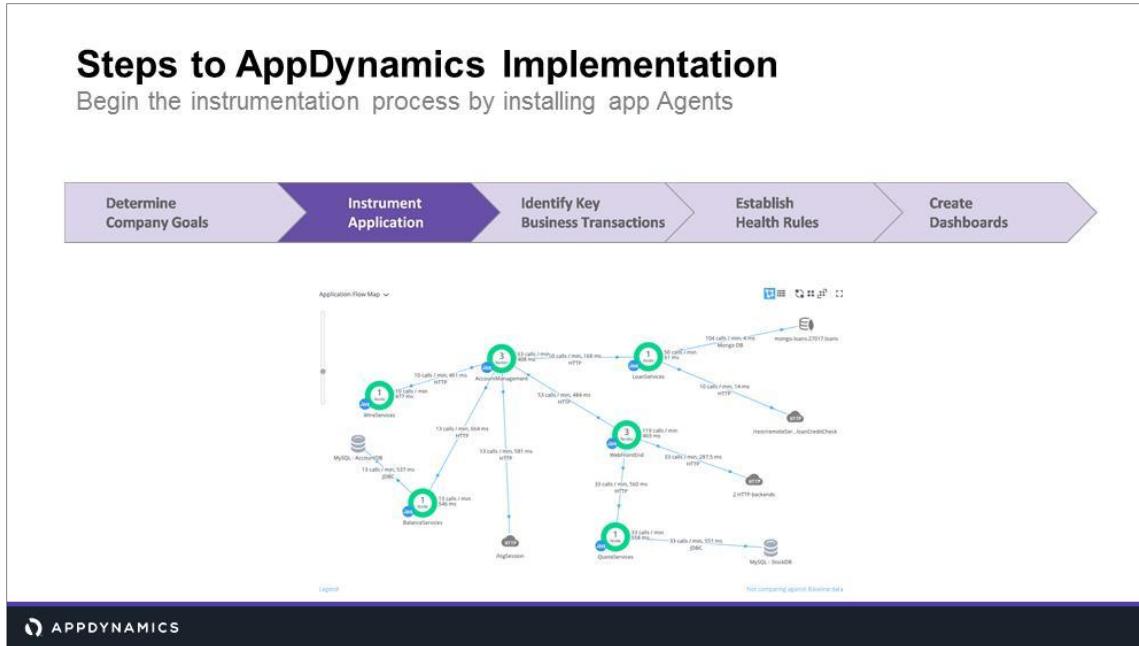


General answers may include (but not be limited to):

- Visibility into production
- Proactively identify and resolve issues (MTTR)
- Improve software quality
- Improve customer satisfaction
- Correlate application performance to business goals

Some specific questions you, or others in your company, may want answers to include:

- Why am I losing customers?
- Where are the bottlenecks in my system?
- How can I figure out if the vendors I’m using are giving me poor service at certain times?
- Who’s killing my database with their thousands of SQL statements?
- How much revenue do I lose when the Credit Card gateway breaks down?



Once App Agents are installed and configured, then you start to see Flow Maps in the Controller. Getting agents into your systems is a key early step that has to happen before monitoring can begin to work. Even once the App Agents are working, there is going to need to be time spent creating Business Transactions and updating the BT Rule Detection configurations so you see the specific BTs that are important for you.

Once Agents are installed and Flow Maps appear in the Controller, you should review the Flow Map(s) for a 'sanity check' to confirm the mapped architecture makes sense for your application(s) landscape.

Steps to AppDynamics Implementation

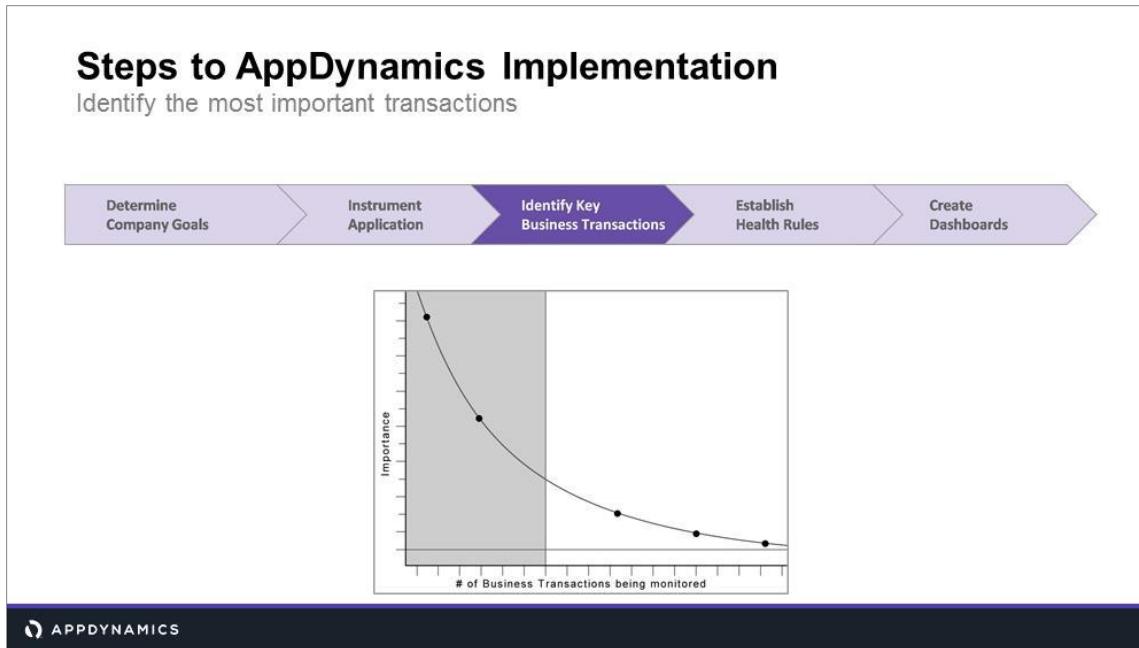
Identify the most important transactions



- Business Transactions are the first and most significant indicator of application performance
- BTs should:
 - Be defined and tuned to reflect business needs
 - Be tied to Health Rules



Our goal is not to provide you with a Business Transaction for every single little action in your system, or to somehow monitor every single icon on every webpage. We could do that, and there are competitors who do as well, but it comes at a price - overhead. At AppDynamics, we set out to run in Production with the lowest possible overhead to avoid impacting how your applications function. We are built on the principle that you don't need to monitor absolutely everything, just the parts that matter. This drives everything we do, including the limit on the number of BTs and the recommendation to every customer that they identify which customer journeys matter most and don't try to measure every single one.



The idea is to monitor only the most important transactions - the ones that substantially affect your customers and your business.

The more transactions you monitor, the more difficult it is to focus on the ones that really matter.

Steps to AppDynamics Implementation

Identify the most important transactions

```
graph LR; A[Determine Company Goals] --> B[Instrument Application]; B --> C[Identify Key Business Transactions]; C --> D[Establish Health Rules]; D --> E[Create Dashboards]
```

How to think about BTs:

- Understand customer experience
- Center on critical application functionality, as defined by business stakeholders
- Metrics become KPIs
- Every BT has an owner: someone who is responsible for its health

 APPDYNAMICS

Remember, customers have two complaints: it's slow or it's broken.

Each unique Business Transaction maps to a particular (and important) customer/user journey. The aim of the BT is to record the experience of the customers travelling that particular journey, revealing to you how long it took them, whether there were issues during their specific journey, and how many customers are on the particular path through your systems.

These journeys become your KPIs, as they are key details we capture about requests which map to the particular BT.

It should almost be implicit that to have selected a particular BT, the functionality being monitored matters and that someone in your organization is therefore going to care if it doesn't work as expected.

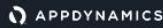
Discussion

Transaction Priority

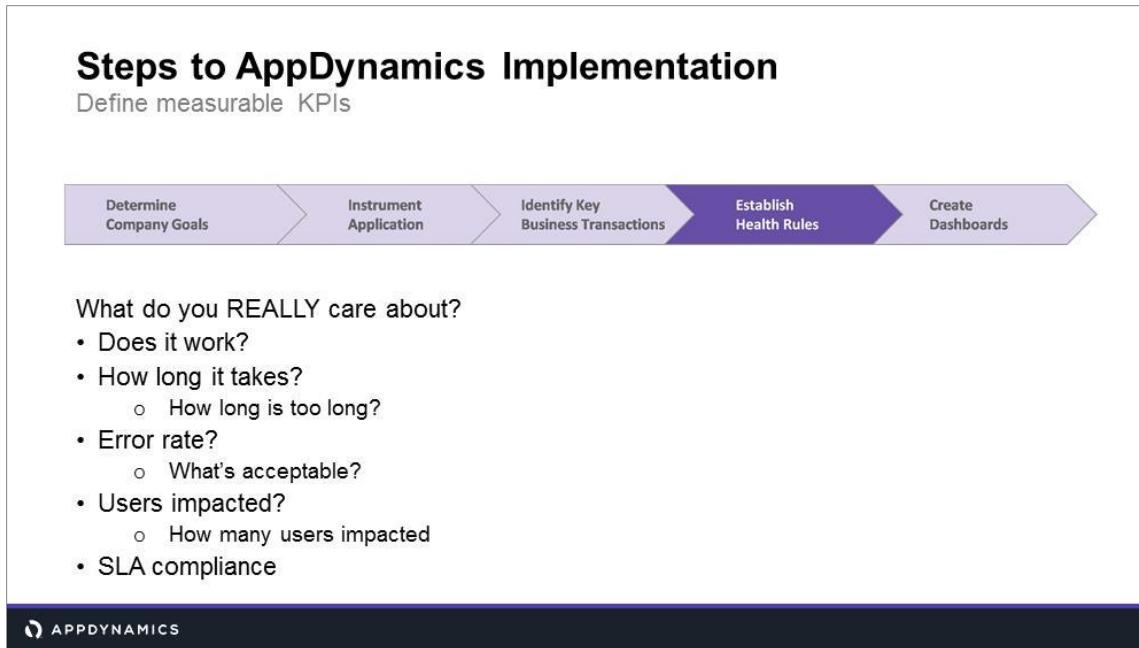


Prioritize the following transactions:

- Login
- Logout
- About
- Rent a Movie
- Rate a Movie
- View Previous Rentals
- View Previous Ratings
- Search for a Movie
- View Movie Report



These transactions represent a UI-driven customer experience. What about a services-based system? What might their key transactions be?



Now that you have worked out what's important to monitor, think about:

- What is the normal level of performance you expect the system to deliver?
- What is not acceptable in terms of response time, number of errors, number of slow (or very slow) calls over time, etc.

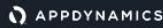
Thinking in these terms, and identifying what your KPIs are, leads you to defining Health Rules to enforce your expected performance levels.

Steps to AppDynamics Implementation

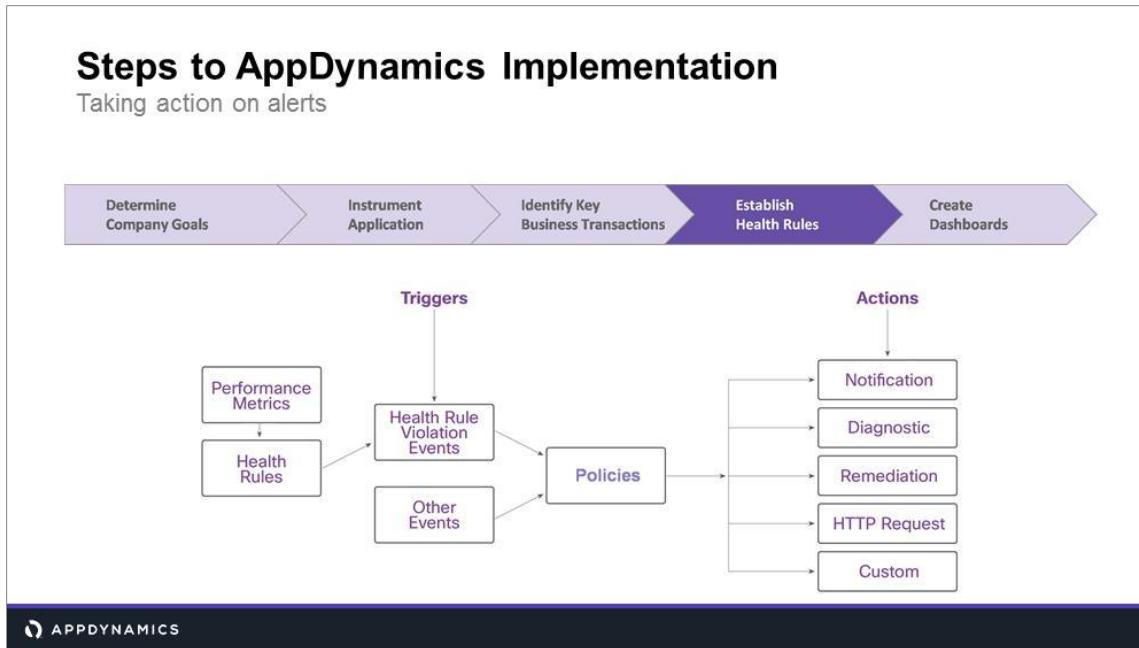
When to wake someone up



- When is it too bad?
- When do you take action?
- What action do you take?
- Is it different for high priority transactions?
- If Login slips from ART of 40ms to 200ms, how quickly do you need to react?



Think about the Login BT. It might typically run at about 40ms with infrequent errors. Now imagine that one night, because of a batch job impacting the database, the Login BT changes to an average of 200ms response time. The customer isn't going to notice this as an issue, but you will want to be aware of it. Is it important enough to wake someone up during the night, or would you be ok waiting until the next day to investigate and resolve the issue?



The first step towards putting in place a fully proactive framework for your system is identifying which BTs to develop Health Rules for.

The next step is to set up Policies and Actions to make AppDynamics do something when a Health Rule is violated (e.g., send out an Email and/or SMS, do a Thread Dump on a given JVM, restart a service using a script, post a HTTP request to an Admin Console to increase scaling, or automatically generate a new ticket in your Jira system).

Discussion

Question



What might be acceptable metric thresholds for the following transactions? What might be the actions for Health Rule violations for each?

- Login
- Run a report
- Take payment



Login

A transaction such as Login should be very reliable and responsive. Depending on the system, it might be an issue if it starts to take 500ms, but for many users that still wouldn't prompt them to immediately go elsewhere (i.e., your competitor), however if it's failing or taking several seconds, customers will likely start to get annoyed and begin submitting support tickets or even not bothering to come back.

In the event that Login slows down, you might just want to be notified at a certain performance threshold, and then actually wake someone up if it crosses a higher threshold. Similarly, if the number of Login Errors rises above a certain point, you might want to take immediate action such as waking someone up to figure out the problem.

Run a Report

Depends on whether or not this is an internal function used by employees, or something that customers expect will be immediate or not: if perception is that there is a lot of data to process to create the report, then several seconds duration before the response is returned will be seen as ok.

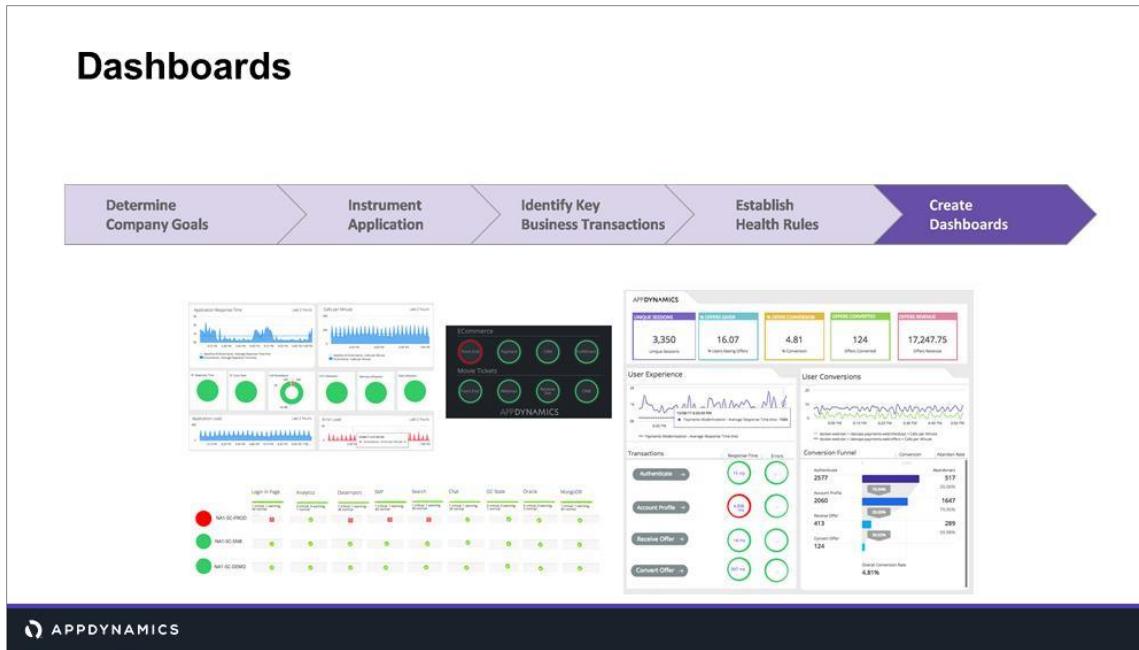
Depending on the type of report and who the user is (employee or customer) you might want to either just send a notification, or generate a ticket in Jira, or potentially message someone to look into it right away.

Take payment

In this case it very much depends on what customer perception is of how long payment takes. E.g., if they assume that their bank or other financial institution has to accept the

debit before your system confirms the payment has been successfully taken, then there is potentially some leeway in how long is “too long.” You certainly do not want any errors to occur during the transaction, leaving the customer unsure of whether the payment went through or not.

If your payment gateway stops working, and your system is supposed to allow purchases at any time of the day, then clearly immediate action is required, either in the form of notifying the appropriate support staff immediately, or perhaps restarting a service (if you know that this will resolve the problem).



Once you have identified and configured the right BTs and set up your Health Rules and Actions, you'll want to define Custom Dashboards so that you can see how your systems are running at a glance, and others in your organization can also review how particular systems are running, how many customers are using specific functionality, how much revenue the business is earning, etc.

Review Question - Answer

You must identify your key Business Transactions before instrumenting your application with AppDynamics Agents.

True

False

Business Transaction Instrumentation

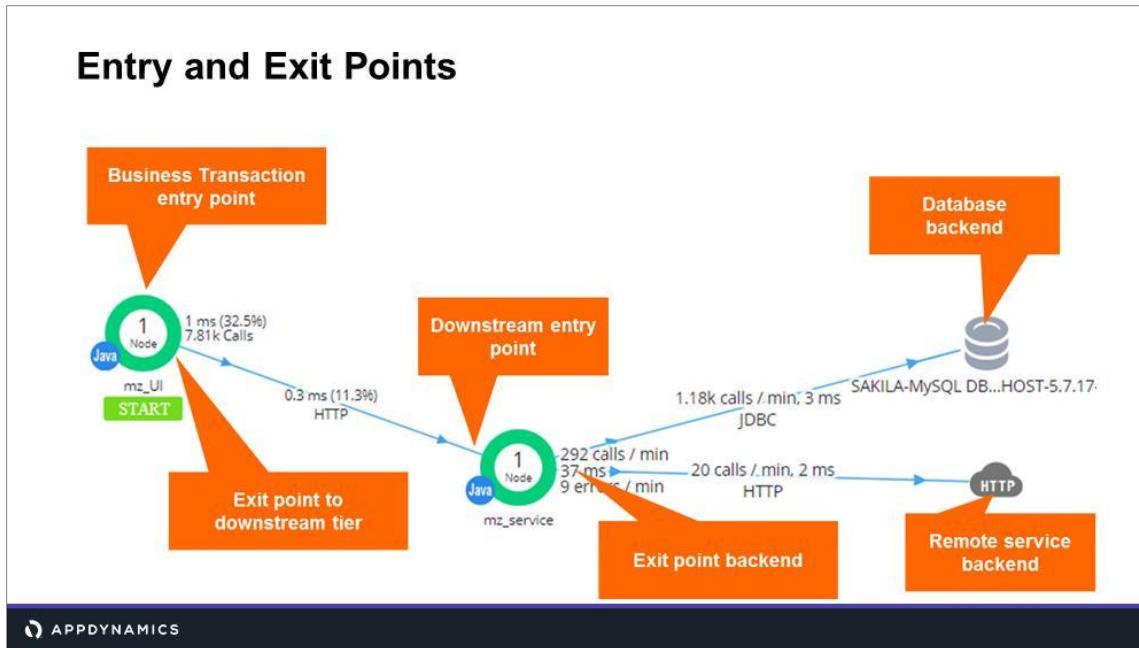
Business Transactions

How to Monitor Key Applications

Although Business Transactions are used to monitor key applications, please keep in mind:

- BTs do not automatically appear for only the functionality that you want to monitor, with meaningful names displayed in the UI.
- There is a series of steps to perform in order to end up with just the BTs that you want to see.
- These steps involve determining what to name individual BTs and instrumenting BT Rules.

Business Transactions				
Details	Filters	Actions	View Options	Configure
Name	Hea...	Response Time (ms)	Calls / min	Errors
Sign In	✓	3	132	
Home	✓	0	90	
Check Balance	✓	37	86	
Transfer Money	✓	30	58	
Apply For Loan	✓	0	45	
CC Payment	✓	0	43	
Check Rates	✓	29	32	



This diagram depicts how distributed applications communicate with one another. Entry point and exit points are key transaction terms as they help you describe specific points in the transaction flow. Note where Entry Points and Exit Points are placed.

Entry Points are the places in a process **where execution of the code begins**. In web applications that have user interfaces these are often Servlet classes for Java applications and ASP.NET classes for .NET applications. Other languages have similar classes that identify Entry Points.

Exit Points are places in a process **where execution of the code calls some external service**. The most common Exit Points are database calls and web service calls.

When the destination of an Exit Point is another service that is monitored by AppDynamics then AppDynamics will employ the **Tag and Follow** mechanism (discussed soon). For Tag and Follow to work, Exit Points from an upstream process must match up to an Entry Point in a downstream process.

AppDynamics excels in providing extensive coverage for Entry Points and Exit Points in modern applications.

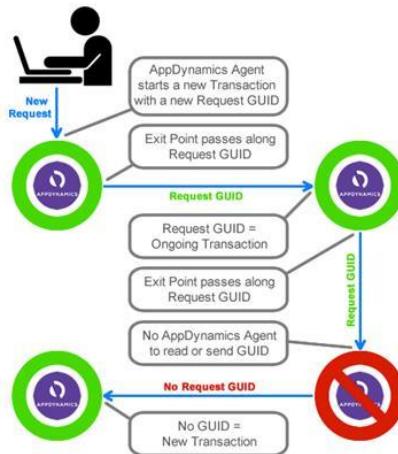
Tag and Follow

AppDynamics uses a technique of decorating calls with a request GUID so we can follow the flow of requests.

AppDynamics supports a very large variety of distributed services frameworks.

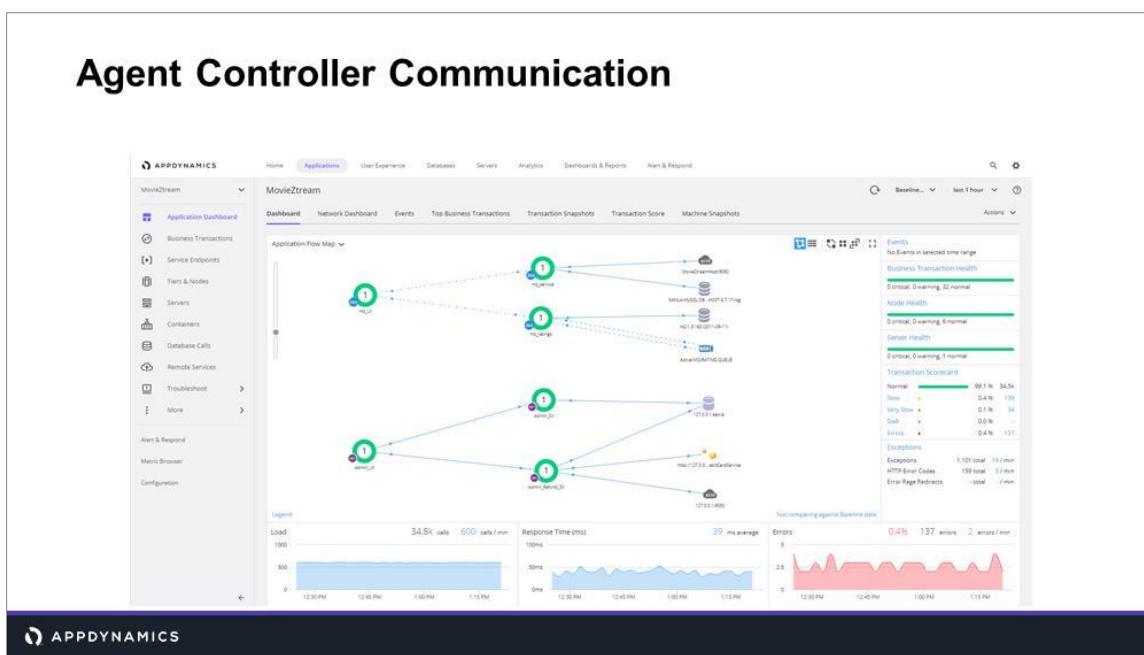
Downstream services become part of the BT that is discovered in an upstream process.

Implementing custom correlation can often resolve failed correlation issues.



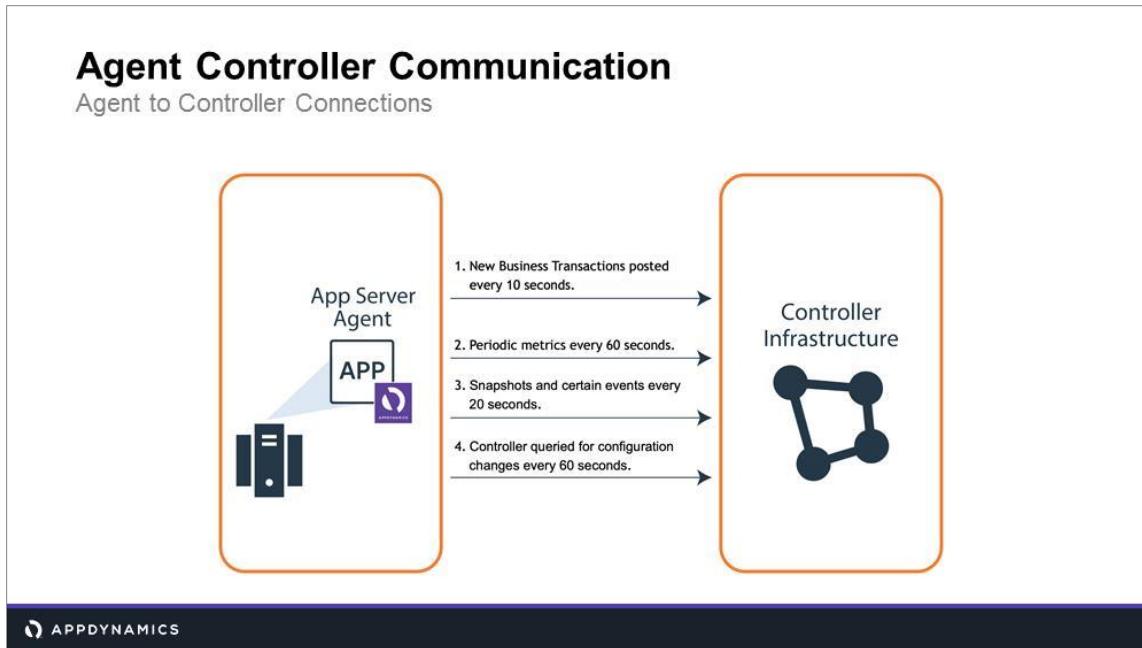
The way that AppDynamics determines how downstream services are related is via a concept called **“Tag and Follow”**. Tag and Follow means that AppDynamics **decorates outbound calls via an AppDynamics GUID** (like for example from ECommerce-Services to Order-Processing-Services and from ECommerce-Services to Inventory-Services) to downstream tiers. The AppDynamics agents in the downstream systems see that there is a GUID in the incoming call. This information is used to stitch together a transaction topology.

Agent Controller Communication



AppDynamics' architecture is deliberately designed to avoid impact on the performance of the systems it is monitoring and remain safe from attack. To this end, communication between the agents and Controller is initiated by the agents. This helps to minimize interference with the system that hosts the agent and limits the load on the network. App agents query the Controller for any configuration changes every 60 seconds. This helps to explain the typical 2-3 minute delay that you observe between making configuration changes and seeing the result of those changes back in the Controller.

Because our architecture consciously avoids unnecessary system load or impact on application performance, many of the top banks and insurance companies have adopted AppDynamics. Numerous government agencies and industry leaders have also implemented our monitoring solutions.



Each AppDynamics agent has multiple communication channels for different purposes that initiate connections to the Controller independently, and at different time intervals.

- The agent configuration channel queries the Controller for any new configuration changes, and downloads these changes when available, every 60 seconds.
- The agent metric channel posts all new periodic metrics, including JMX, Windows performance counters, and business transaction metrics to the Controller every 60 seconds.
- If there are new business transactions that have not been seen before by the agent, they are posted to the Controller for registration every 10 seconds.
- The App Server Agent sends any new snapshots to the Controller every 20 seconds. Certain events will also be sent to the Controller every 20 seconds. (Not all events are generated at the App Server Agent level; the other events are created in the Controller).

For more info see: <https://docs.appdynamics.com/display/PRO45/Agent-to-Controller+Connections>

Review Question - Answer

A Business Transaction is defined and named:

- A) At the first entry point
- B) At the first exit point
- C) By the customer

Automatic Discovery

Use Case

Automatic Discovery

- Start by understanding the functionality of the app to be monitored
- Identify what are the most important functions that users/customers do in the system
- Which functions will significantly impact your business if they are slow or broken?
- For the MovieZtream app, what are the most important functions?



 APPDYNAMICS

When looking at a system that is going to be monitored using AppDynamics, beyond the technical implementation phase of installing and configuring the App Agents, you have to start by exploring the functionality.

You want to understand what the key transactions are, i.e., what are the most important functions that customers do in the system? What might customers complain about?

If they were complaining that “It’s slow” or “it’s broken”, then “it” is the customer experience of what they’re trying to do, which matters to them e.g., rent movie, rate movie, etc.

These are going to be what you want your Business Transactions to monitor, to give you the detailed insight into how things seem from the customer perspective.

Entry Point Types

Type	Naming Formula	Auto-Discovery or Custom?
Java Servlet / .Net ASP	URI Segments	Auto + Optional Custom
Struts Actions	ActionName.MethodName	Auto + Optional Custom
Spring Beans	BeanName.MethodName	Auto + Optional Custom
Web Service / WCF	ServiceName.OperationName	Auto + Optional Custom
JMS / Message Queues	Dest Name (or Listener Name if Dest n/a)	Auto + Optional Custom

Transaction Detection		
Further Frameworks supported		
Java Frameworks	JVM Language Frameworks	.NET Frameworks
Apache Cassandra with Thrift	Akka Actor	ASP.NET
Google Web Toolkit	Akka HTTP	ASP.NET MVC 2...5
Open Source CometD	Groovy	ASP.NET Core on full framework
Open Source Java Server Faces	Play for Scala	.NET Core on Windows
Open Source Jersey	Spray Toolkit	.NET Core on Linux
...	...	Profiler Build
		Infrastructure (SDK)
		.NET Remoting
		...



Beyond those mentioned on the previous slide, AppDynamics offers a varying level of support for other Java Frameworks and JVM Language Frameworks. In some cases, auto detection is not enabled by default, or perhaps auto-naming isn't available.

For .NET there is also extensive support for transaction detection for a range of frameworks and Queuing technologies.

For a full list of supported frameworks, see the Java Supported Environments and .NET Supported Environments pages of the AppDynamics Product documentation.

Rent Movie Business Transaction



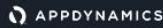
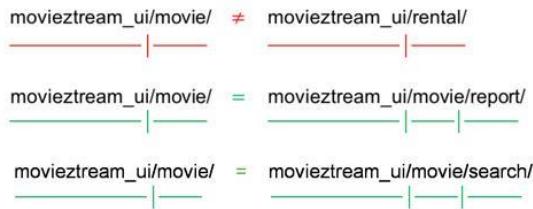
The screenshot shows a web browser displaying a 'Rent Movie Form' on the 'MOVIE ZTREAM' website. The URL in the address bar is: `ec2-54-186-250-77.us-west-2.compute.amazonaws.com/moviezstream_ui/customer/rent.htm`. Three orange callout boxes label the URL segments: '1st SEGMENT' points to 'ec2-54-186-250-77.us-west-2.compute.amazonaws.com', '2nd SEGMENT' points to 'moviezstream_ui', and '3rd SEGMENT' points to 'customer/rent.htm'. The page content includes a navigation menu on the left with links to 'Home', 'Customers', 'Movies', and 'Admin'. The main form has fields for 'Movie Title*', 'Credit Card Number*', and 'To be returned by'. Buttons for 'Rent Movie' and 'Clear' are at the bottom. The AppDynamics logo is in the bottom right corner.

Even with a simple app like MovieZtream, examining the third segment in the URL uncovers important information for some BTs.

Transaction Detection

Servlets & ASPX

- Update automatic discovery rules to regroup browser-based requests into different custom arrangements of Business Transactions
- Configures an entry point to the originating tier on the invocation of the service method of a Servlet
- Uses first two segments of URL by default, but can be modified



AppDynamics allows you to detect browser-based transactions on the invocation of the service method of a Servlet or an ASPX page. Then, the response time for the transaction is measured when the Servlet entry point is invoked for Java, or when the transaction detection is invoked for .NET.

By default, AppDynamics identifies all Servlet-based and ASPX-based transactions using the first TWO segments of the URL, but you can modify that setting and go deeper. Also, you have the option of setting up custom match rules to identify some Servlet-based and ASPX-based Business Transactions as needed.

First Two Segment URLs

Default

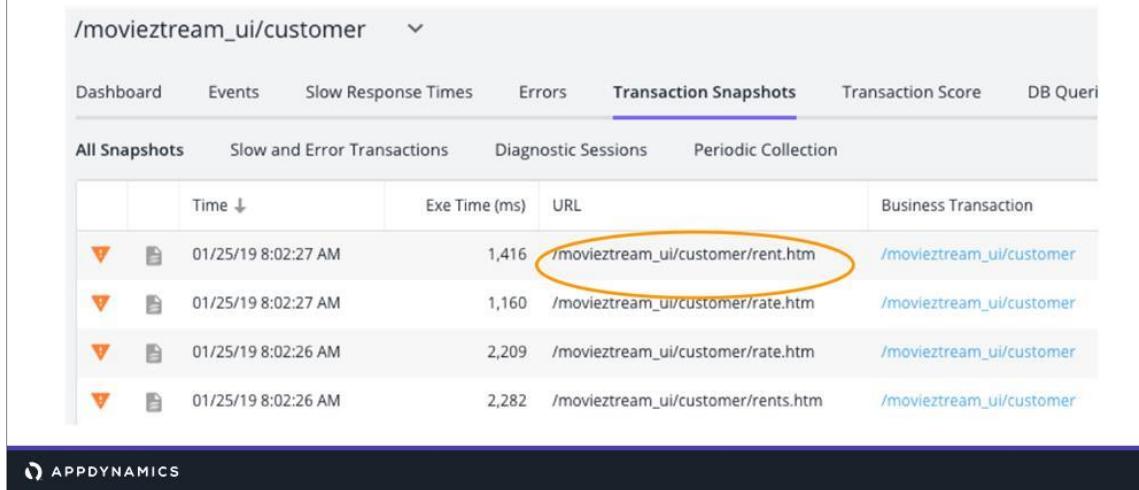
Do these BTs match the Customer Experience?

Name	He...	Response Time (ms)	Calls / min
/movieztream_ui/customer	✓	41	172
/movieztream_ui/login.htm	✓	13	90
/movieztream_ui/home.htm	✓	0	88
/movieztream_ui/movie	✓	72	57
/movieztream_ui/logout.htm	✓	0	30
/movieztream_ui/admin	✓	2,877	2

 APPDYNAMICS

The issue with seeing only two segment URLs is that they may not map strongly to the customer experience. MovieZtream is relatively simple and this is reflected in the short-ish URLs it has. Real-world systems are more complex and consequently have more complex resource paths and request types.

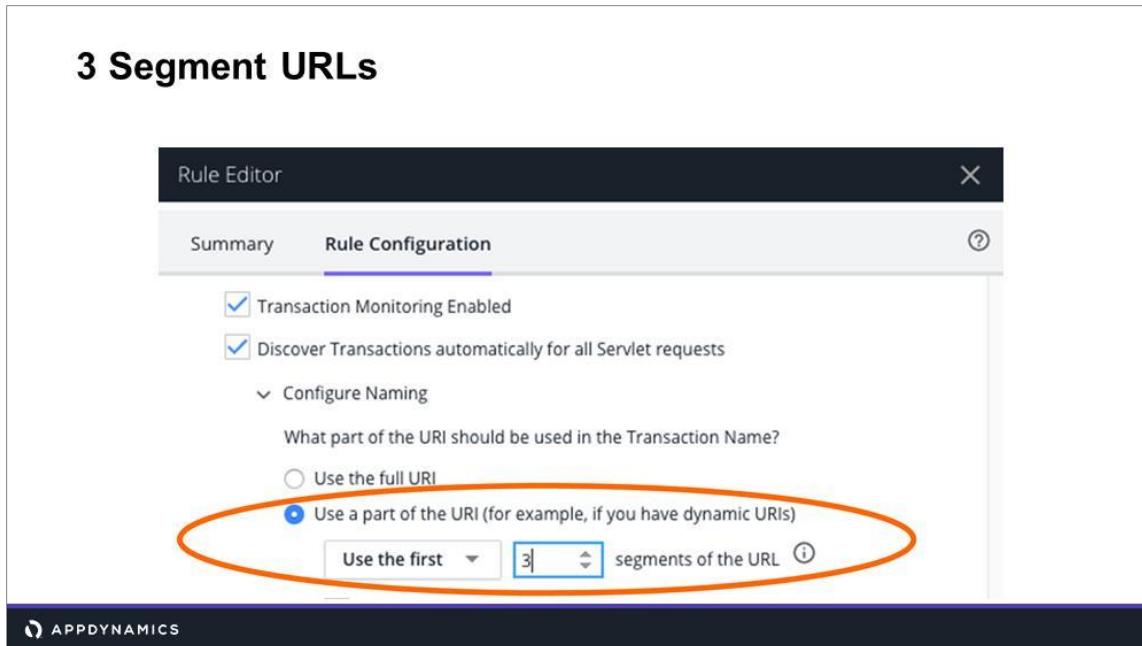
Snapshots Reveal URLs Matching Customer Experience



	Time ↓	Exe Time (ms)	URL	Business Transaction
▼	01/25/19 8:02:27 AM	1,416	/movieztream_ui/customer/rent.htm	/movieztream_ui/customer
▼	01/25/19 8:02:27 AM	1,160	/movieztream_ui/customer/rate.htm	/movieztream_ui/customer
▼	01/25/19 8:02:26 AM	2,209	/movieztream_ui/customer/rate.htm	/movieztream_ui/customer
▼	01/25/19 8:02:26 AM	2,282	/movieztream_ui/customer/rents.htm	/movieztream_ui/customer

In the case of MovieZstream, changing the number of segments to look at in the URL from two to three, reveals additional transaction-specific information. The 'customer' BT would include Rent, Rate, and Rents if limited to two URL segments. By changing it to three segments we uncover addition BTs that are relevant to the customer experience.

3 Segment URLs



Rule Editor

Summary Rule Configuration ?

Transaction Monitoring Enabled

Discover Transactions automatically for all Servlet requests

Configure Naming

What part of the URI should be used in the Transaction Name?

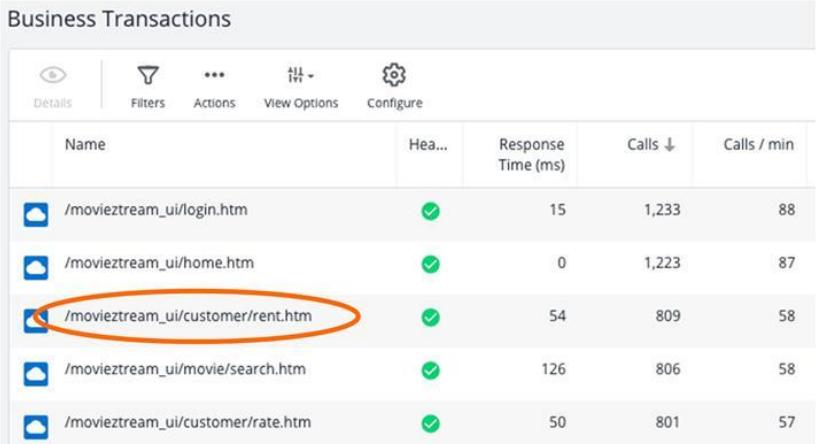
Use the full URI

Use a part of the URI (for example, if you have dynamic URLs)

Use the first 3 segments of the URL ?

You can update the number of URL segments to use in the Auto Discovery Rules (Java Servlet or .NET ASP) under Configure Naming on the Rule Configuration tab.

Now the BTs Match the Customer Experience



Name	Health	Response Time (ms)	Calls ↓	Calls / min
/movieztream_ui/login.htm	✓	15	1,233	88
/movieztream_ui/home.htm	✓	0	1,223	87
/movieztream_ui/customer/rent.htm	✓	54	809	58
/movieztream_ui/movie/search.htm	✓	126	806	58
/movieztream_ui/customer/rate.htm	✓	50	801	57

A few minutes after the configuration change (switching to 3 segments), we should start to see the new, more fine-grained BTs appearing. The old 2 segment BTs are not lost, however, but gradually slip out of view as the Controller is typically only showing the last X minutes of information. By default, the list of Business Transactions only shows ones that have executed in the time frame selected. This can be changed under Filters.

Additional BT Operations

These operations can be performed by right-clicking one or more Business Transactions:

Rename

- BTs do not automatically show up with user friendly names
- You want to rename your key Business Transaction

Delete

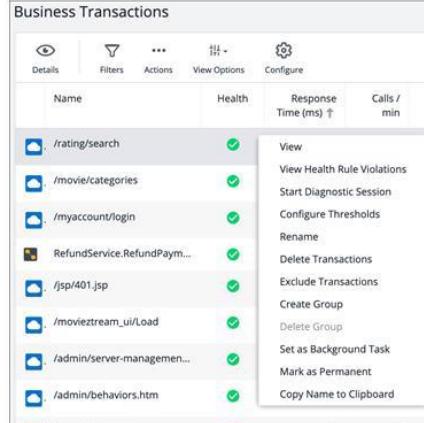
- Purge all historical information for a BT

Exclude

- Disable and hide a BT

Group

- Aggregate the summary stats for a BT



Delete - AppDynamics may discover some Business Transactions that are not significant to your business. You can remove those and just keep the transactions that have higher business impact. Select any of the Business Transactions on the list, and access **More Actions > Delete Transactions**.

Exclude - If you delete a transaction it may be re-discovered. If you don't want that, use Exclude Transaction instead.

Rename - You can rename Business Transactions to give them 'human-friendly' names for easier reference. When you rename a BT you rename it for everyone.

Group - Grouping helps you access the consolidated key data, while keeping the metrics for each Business Transaction still accessible. To create a group, you must select more than one BT and then go to **More Actions > Create Group**.

We will look at the application of these operations as we begin to define Custom Match Rules later in this course.

Review Question - Answer

For most applications, Automatic Discovery finds the optimal Business Transactions and no other configuration is needed.

True

False



The Auto Discovery Rules are there when anyone installs AppDynamics - they are a "one-size fits all" best estimation of what might be useful Business Transactions, but never a good fit for any one specific customer.

Best Practice: Select Business Transactions by Hand

Notes From the Field

The argument against Auto-discovery

Business Problem

- After instrumenting their app with AppDynamics and using Auto-discovery, a customer found hundreds of business transactions showing up in the controller.
- The list of BTs did not sort by importance to highlight which were the transactions they identified as critical to their business operations.

Solution

- They removed the Auto-discovered transactions and followed the process laid out in the following slides.



BT Selection Best Practice

Selecting Business Transactions by Hand



After identifying what you want to monitor, you can assume that Auto-Discovery won't give you precisely what you want. Follow this process to define the specific Business Transactions that are important to you:

1. Apply BT Lockdown to freeze the current set of BTs
2. Switch to Full URI naming to get unique BTs for every unique URI
3. Delete all existing BTs so all traffic will now go into the All Other Traffic buckets
4. Cherry-pick and register the BTs you want from the All Other Traffic buckets



The generally recommended strategy in the case of NEW installs of AppDynamics is to expect that you will ultimately need to create your own BT Detection rules, because while the Auto Discovery rules which are present Out-Of-The-Box are impressive, they won't necessarily work perfectly for every single customer.

Therefore, you should expect that you will need to define Custom Match Rules to achieve the specific BTs that you want. Because the Auto Discovery rules are operational as soon as the agents begin reporting to the Controller, the first step in this process is to stop the agents from revealing new BTs, by using the BT Lockdown feature, which 'freezes' the current set of BTs so that even if new traffic is detected, no new BTs will show up.

Deleting the current list of BTs showing in your Controller then forces everything to filter into the relevant All Other Traffic buckets, where you can review specific request paths and identify exactly which ones matter to you.

Now let's walk through this process in detail.

The All Other Traffic Buckets

The first set of 200 Business Transactions:

Operate on a first-come, first-served basis.

(The first 200 BTs that the system detects are registered.)

The system uses an overflow collector for traffic if:

- >200 BTs detected on application
- >50 BTs detected on a node (agent)
- New traffic is detected when BT Lockdown is enabled

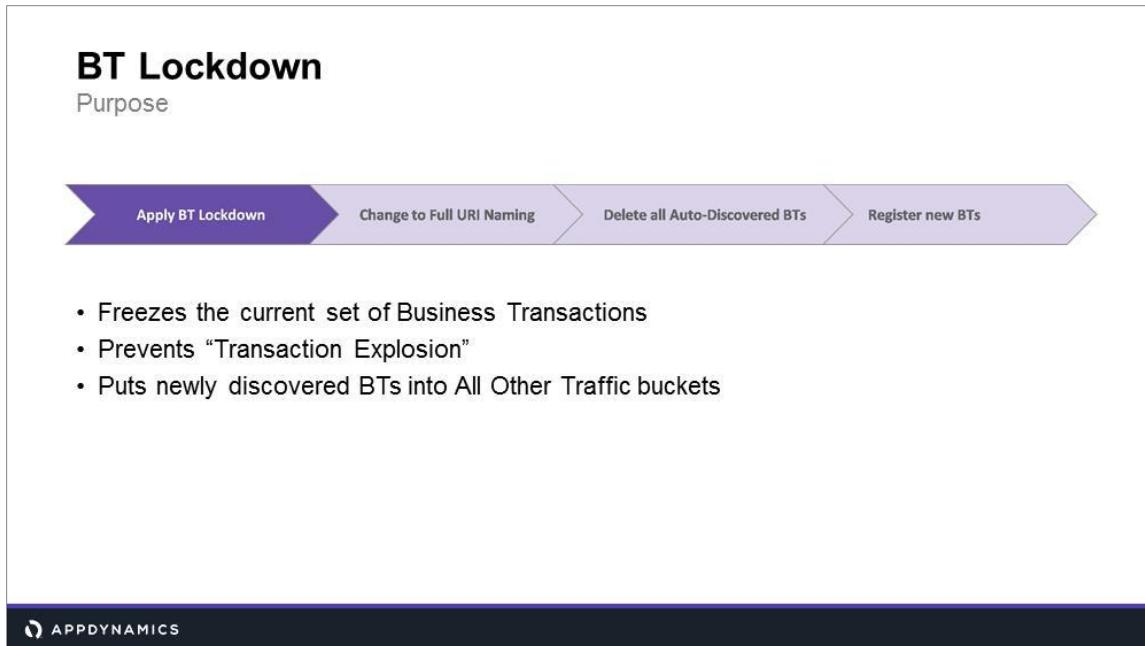
Name	Header	Req. Time (ms)	Calls / min
20-Home	0	89	
50-Logout	1	44	
Admin Customer	877	10	
All Other Traffic - mz_UI	20	87	

First it's important to understand the All Other Traffic buckets and how they work. Auto-Discovery Rules may put many of your 'valuable' transactions in the All Other Traffic buckets, and some unwanted BTs would show up in your BT list; both of which you want to avoid.

AppDynamics will assign what *would* be independent BTs into the *All Other Traffic* buckets, once it reaches the standard limits at either Application level (200) or Node level (50).

In addition, if BT Lockdown is enabled, then any subsequent traffic (i.e., not being captured by an existing BT) gets directed into the All Other Traffic bucket.

Notice that there is not just one bucket, but one per unique Tier where one or more BTs have Entry Points.



The first step in hand-selecting BTs is turning on BT Lockdown. This freezes the current set of BTs so unwanted BTs don't continue to show up.

Apply BT Lockdown



Instrumentation

Scopes Transaction Detection Backend Detection Error Detection Service Endpoints

Rules Tiers More

Business Transaction Lock Down

Enable Business Transaction Lock Down

Business Transaction Automatic Cleanup

Enable Business Transaction Automatic Cleanup

Monitor Business Transactions for 15 minutes since creation

Remove Business Transactions that have less than or equal to 1 Calls

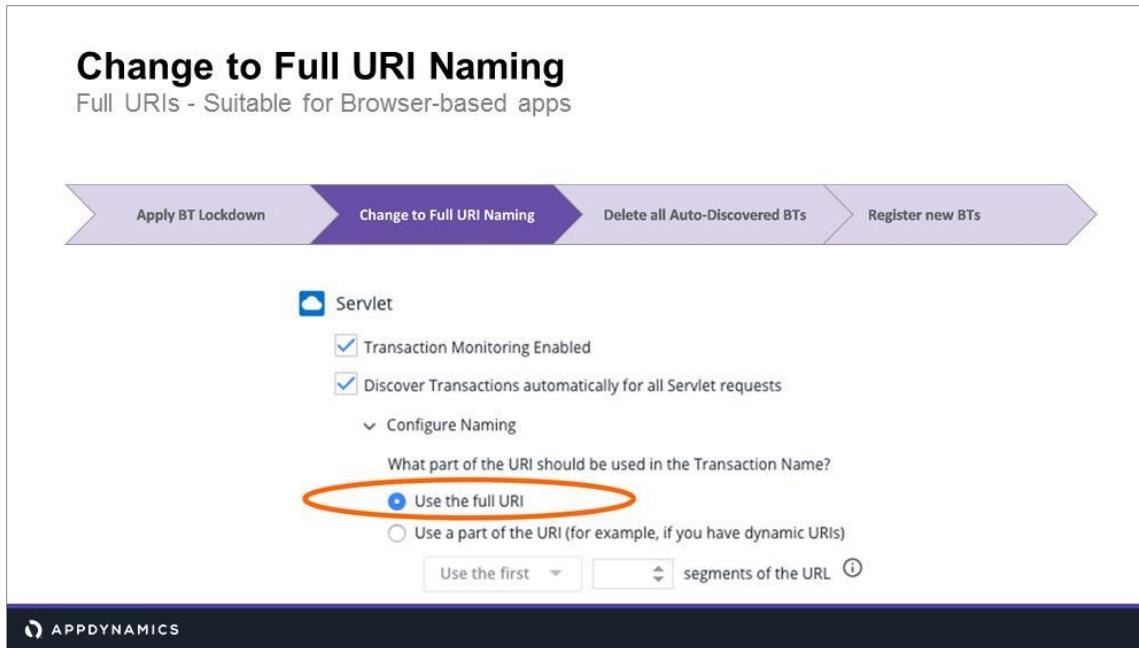


APPDYNAMICS

Go to **Configuration > Instrumentation > Transaction Detection > More**

Change to Full URI Naming

Full URIs - Suitable for Browser-based apps



Servlet

Transaction Monitoring Enabled

Discover Transactions automatically for all Servlet requests

Configure Naming

What part of the URI should be used in the Transaction Name?

Use the full URI

Use a part of the URI (for example, if you have dynamic URIs)

Use the first segments of the URL

APPDYNAMICS

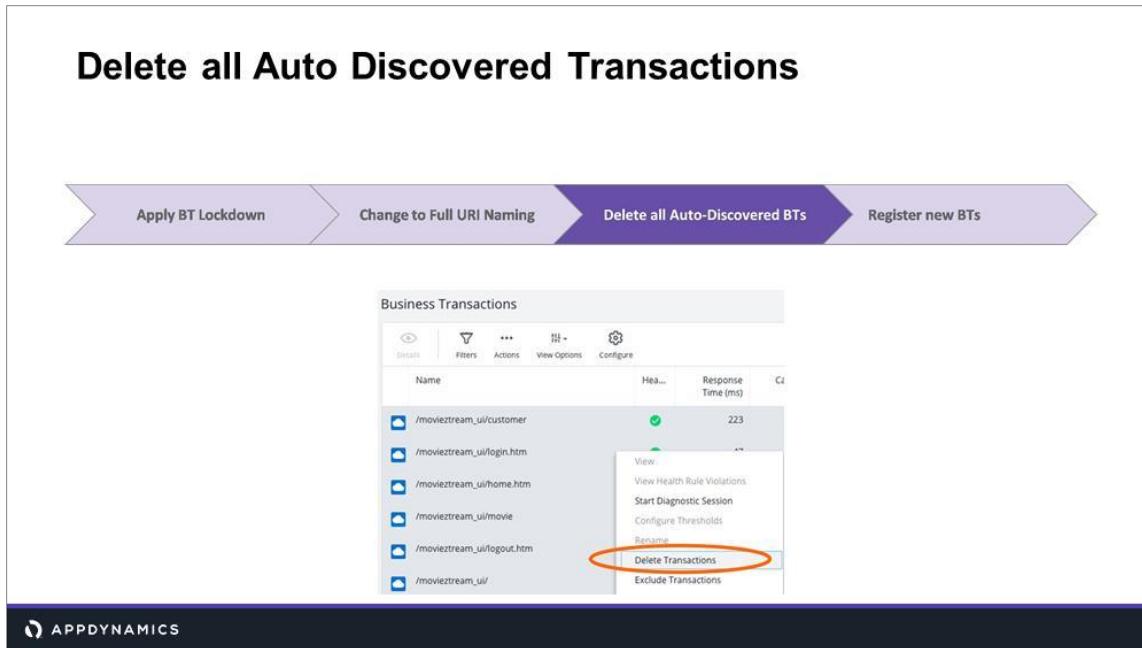
The next step in hand-selecting BTs is to switch from the default two-segment URI detection setting to Full URI detection. This will give you a unique BT for every unique URI detected.

While this may sound great (and you may think that is what you want), keep in mind the general principle of aiming to have only a small number of BTs showing in your Controller, which should reflect the key functions offered by your application. In other words, we're not done. We will continue to refine our BT selection in the next steps.

It is also worth mentioning the default limit of 200 BTs that a Controller account can monitor (there is also a limit of 50 originating BTs on any 1 Node). Once the AppDynamics detects traffic in excess of one of these limits, the surplus is dumped into the All Other Traffic bucket.

We also recognize that your system may not be primarily driven by requests from your end users' web browsers, in fact it may be specifically focused on service requests - providing it is built using a technology which we auto detect, then you will still see unique BTs showing up for unique request paths coming into your application. Review the other technologies listed in the Rule Configuration page for the .NET and Java Auto Discovery rules to see the logic used to determine unique BTs.

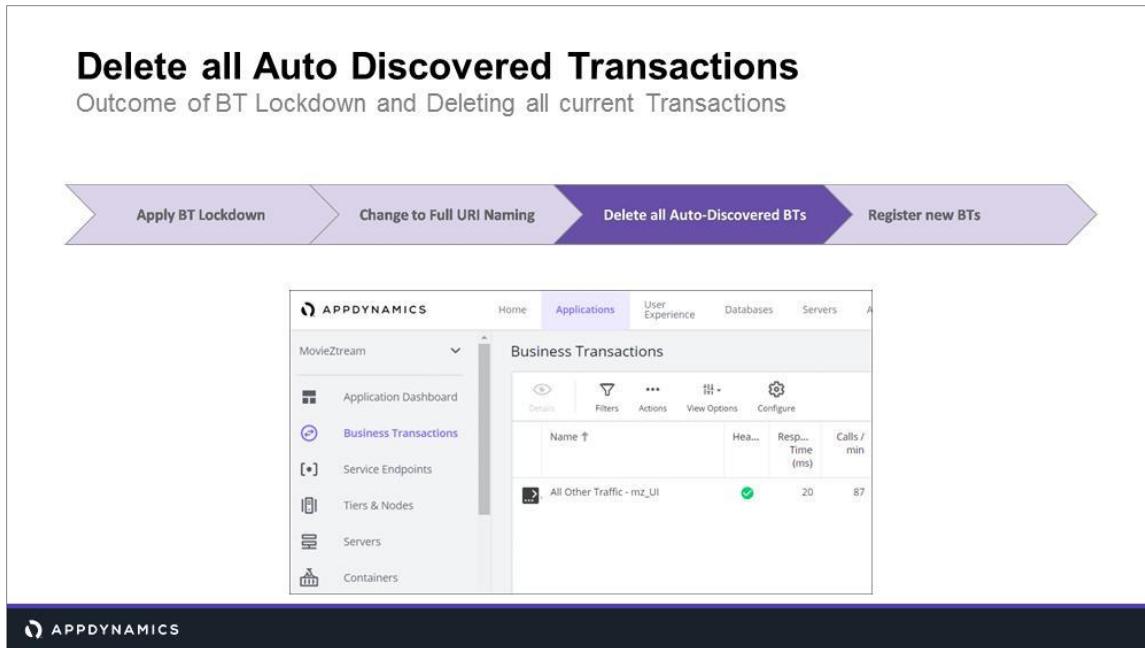
Configuration of other detectable non-browser-based protocols/frameworks is less granular and already set to ON by default.



After engaging BT Lockdown and configuring URI Naming to create unique BTs for every unique URI, the next step in hand-selecting BTs is to delete all the current BTs so all traffic will automatically go into the All Other Traffic bucket from this point forward.

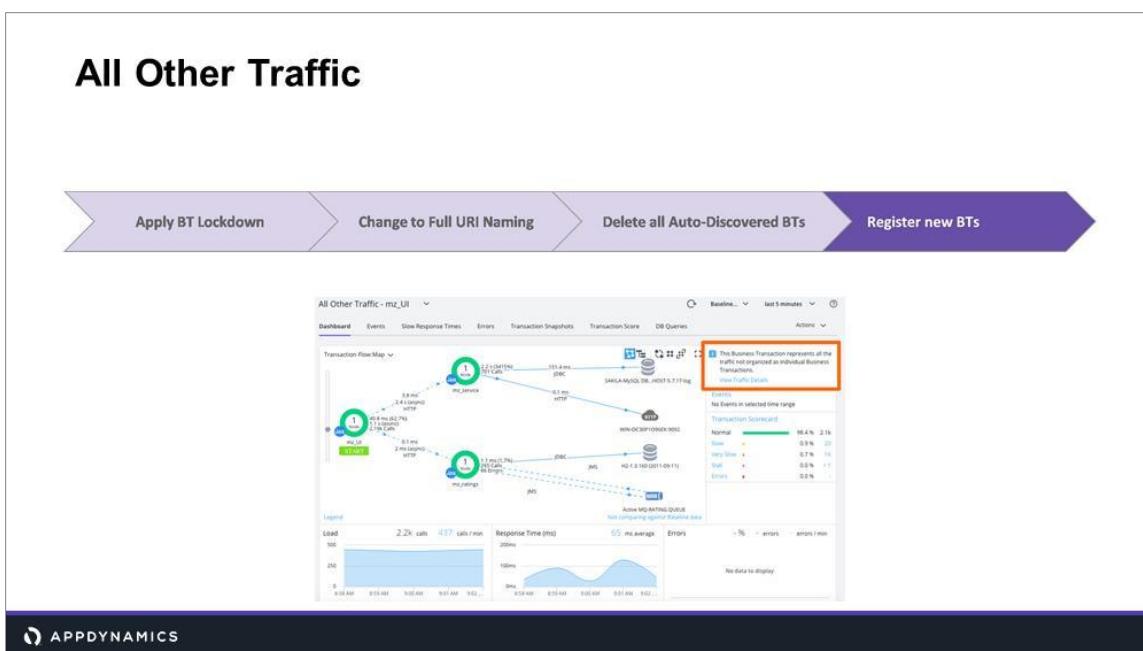
NOTE: Be sure to turn all filters off before selecting BTs (including Performance Data filter). Otherwise you will not delete filtered BTs.

Also available on this screen is the ability to exclude transactions. Excluding the transaction disables the BT for metric processing purposes. When you delete a BT from the list, the system removes its accumulated transaction metrics; when you exclude a BT, the system retains its accumulated metrics along with its underlying configuration.



Once the Business Transactions are deleted, if the load on the monitored application continues, then new traffic will end up in the All Other Traffic buckets (according to Tier of Entry Point).

All Other Traffic



In the 'All Other Traffic - mz_UI' bucket, you see a familiar BT Flow Map. In the top right corner there is a message informing you that this is a representation of all the traffic not organized as individual Business Transactions, and a link to View Traffic Details.

The next step in hand-selecting BTs is to go into the All Other Traffic bucket and begin to pick out the specific BTs you want and register them to upgrade them to first class unique BTs.

Details of Potential Business Transactions



All Other Traffic - mz_UI

If traffic exceeds the 50 business transaction per agent limit, the overflow traffic is collected into a default overflow business transaction called "All Other Traffic - mz_UI".

last 5 minutes

Showing traffic details from 1/25/2019, 9:02:00 AM to 1/25/2019, 9:05:00 AM. [Fetch more...](#)

Business Transaction Name	Count	Type
/movietream_ui/customer/rate.htm	231	SERVLET
/movietream_ui/login.htm	355	SERVLET
/movietream_ui/movie/search.htm	220	SERVLET
/movietream_ui/customer/rent.htm	226	SERVLET

OK

APPDYNAMICS

Clicking on the View Traffic Details link displays a pop-up, with the actual unique requests which *COULD* be upgraded to specific BTs.

Review the requests to determine which should be monitored with their own BT, and which can potentially be grouped together for monitoring purposes.

Register Key Transactions



All Other Traffic - mz_UI

If traffic exceeds the 50 business transaction per agent limit, the overflow traffic is collected into a default overflow business transaction called "All Other Traffic - mz_UI"

last 5 minutes

Showing traffic details from 1/25/2019, 9:02:00 AM to 1/25/2019, 9:05:00 AM. [Fetch more...](#)

Business Transaction Name

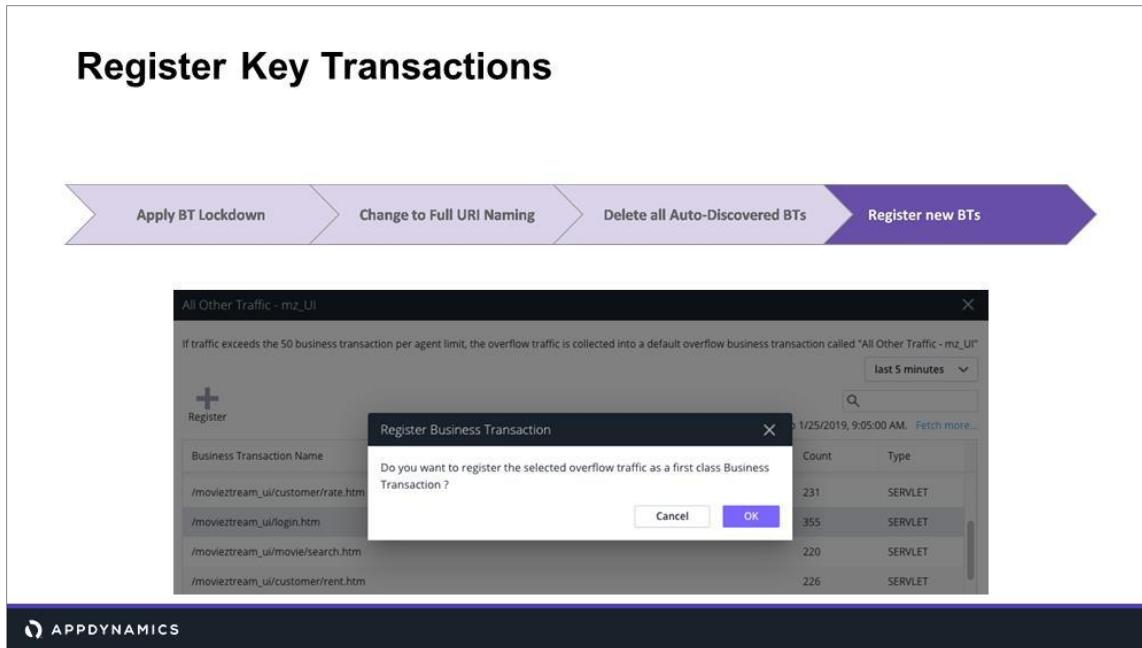
Business Transaction Name	Count	Type
/movieztream_ui/customer/rate.htm	231	SERVLET
/movieztream_ui/login.htm	355	SERVLET
/movieztream_ui/movie/search.htm	220	SERVLET

Register

APPDYNAMICS

An orange arrow points from the text 'Select a request to be monitored with its own unique BT and click the Register button.' to the 'Register' button in the screenshot.

Select a request to be monitored with its own unique BT and click the Register button.



The selected request becomes a specific BT in the Controller, and it will no longer show up in the All Other Traffic bucket.

Remember, AppDynamics sets a limit of 200 unique BTs per account, and 50 per originating Node, so in fact any time that your BT configuration allows more than those limits, once the first 200 (or 50) BTs have been detected, anything over and above will always end up in the All Other Traffic bucket.

Register Key Transactions



Business Transactions

Name	Hea...	Response Time (ms)	Calls ↓	Calls / min
All Other Traffic - mz_UI	✓	36	1,506	377
/movieztream_ui/login.htm	✓	3	92	92

APPDYNAMICS

While this results in this specific BT being displayed in the Controller for this environment, it hasn't created the Custom Match Rule needed for that the unique BT to be found in case you ever want to delete all BTs again. Further, creating BTs via Custom Match rules provides better control over how BTs are defined as well as an easy mechanism to port rules from one controller to another.

Registering a BT by Hand is Not the Full Story

Registering the Business Transaction does not:

- Create or save BT detection rules
- Show anything in BT configuration to:
 - Help you understand what AppDynamics should be monitoring
 - Allow you to migrate the BT to another environment



Clicking the Register button creates a BT for the chosen request, with no extra work or input. What does not happen at this point, however, is the creation of the logic necessary to be able to discover the request again in future; should the BT ever be deleted, or if you move AppDynamics configuration to a new environment. At this stage, the BTs would have to be registered again by hand from the All Other Traffic bucket.

The missing piece to this puzzle is the creation by you of new Custom Match Rules which provide the concrete logic for identifying a specific request and monitoring it from then on with a particular BT. We will discuss creating Custom Match Rules in the next topic.

Review Question - Answer

Business Transaction Lockdown prevents new BTs from showing up.

True

False



Requests that show up in All Other Traffic buckets are not BTs.

Review Question - Answer

Which of the following are features of an All Other Traffic bucket?

- A) The All Other Traffic dashboard displays a Flow Map of the combined paths of the requests it contains.
- B) You can right-click a request in the bucket to create new BT Rules.
- C) The all other traffic bucket continues to capture requests even though BT Lockdown is enabled.

Custom Match Rules

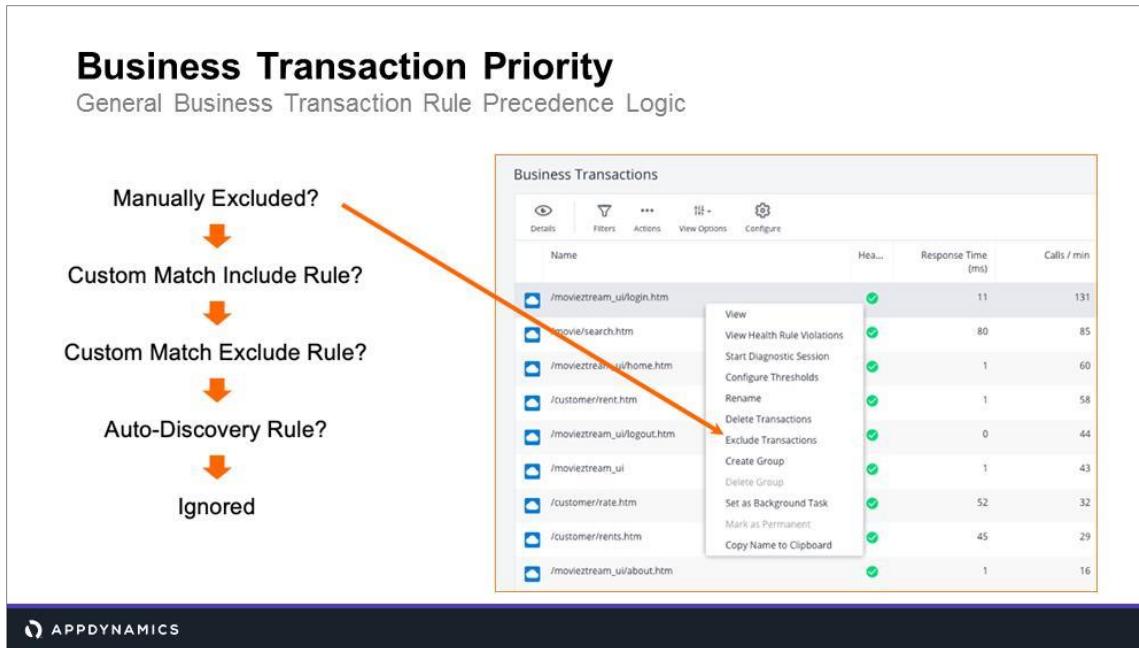
Auto-discovery vs. Custom Match Rules

- AppDynamics is designed to discover most BTs automatically
- More flexibility is available by using Custom Match Rules
- Custom Match Rules are evaluated first

Agent Type	Name	Scope	Enabled	Priority
Web S...	Web Server Auto Di...	Default Sco...	✓	0
Python	Python Auto Discov...	Default Sco...	✓	0
Python	Python Static Conte...	Default Sco...	✓	0
PHP	PHP Auto Discovery ...	Default Sco...	✓	0
Node.js	Node.js Auto Discov...	Default Sco...	✓	0
Node.js	NodeJS Static Conte...	Default Sco...	✓	0
Java	Java Auto Discovery ...	Default Sco...	✓	0
Java	Cron4j	Default Sco...	✗	0

Auto Discovery rules are a great way to potentially detect lots of traffic and get lots of BTs without much configuration effort. However, they are relatively coarse-grained in nature, meaning that they apply an 'all or nothing' logic to detecting and creating BTs, which does not work perfectly for everyone.

More fine-grained control over detection of specific requests is available by using Custom Match Rules (Include and Exclude Types).



AppDynamics is designed to avoid ambiguity over which BT Rule configuration is responsible for detecting any specific transaction. To enforce this, there is an order of precedence across the broad BT Rule Types, as well as Priority logic between similar Rule Types (next slide).

The first order of precedence states that if a BT has been manually excluded from the BT List page, that BT will not be monitored.

Next in order of precedence are the Custom Match Include rules. If no Include rules detect a certain transaction, then it is filtered through the Custom Match Exclude rules. If the Exclude rules don't handle the transaction, it is filtered through the Auto-Discovery rules to see if it may be set up as a first-class BT or not.

Business Transaction Priority

Precedence Logic within similar Rule Types

What happens if two or more BT Custom Match Include Rules could detect the same transaction?

- The rule with highest Priority wins (default '0' is lowest possible)
- Priority settings only apply to similar Rule Types
- Priority '10' in a Custom Match Exclude rule does not override a priority '0' Custom Match Include rule
- Priority '20' in an Auto Discover rule does not override a priority '0' Custom Match Exclude rule

Custom Match Include Rule?

Priority HIGH to LOW



Custom Match Exclude Rule?

Priority HIGH to LOW



Auto-Discover Rule?

Priority HIGH to LOW



Custom Match rules can have priorities assigned to them. This allows you to create rules that create individual Business Transactions.

Higher priority numbers get evaluated first. The idea is that the “catch-all” rule will get evaluated last and will catch all of the Business Transactions that do not match any of the previous rules.

If deployed correctly this methodology will only create Business Transactions for which there are Custom Match rules and will never create an “All Other Traffic – *tiername*” category. This is because the “catch-all” rule will catch all of the overflow Business Transactions and they will never fall into the “All Other Traffic - *tiername*” category.

Custom Match Rules to Break out Key Transactions

Create Custom Match Rules BEFORE Registering



All Other Traffic - mz_UI

If traffic exceeds the 50 business transaction per agent limit, the overflow traffic is collected into a default overflow business transaction called "All Other Traffic - mz_UI"

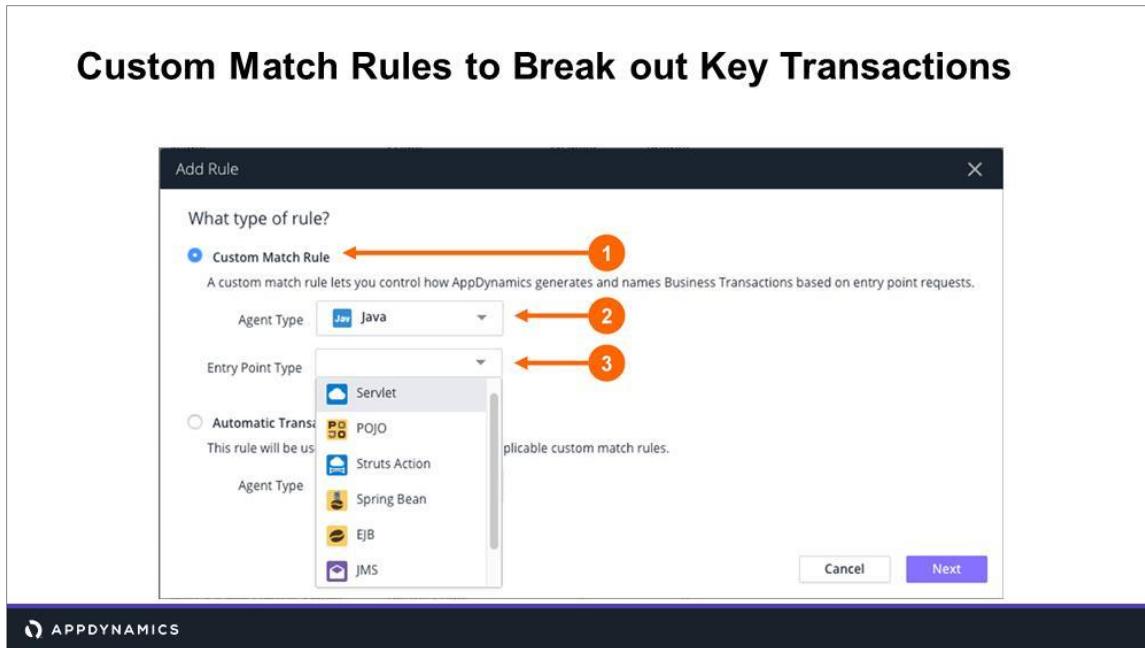
last 5 minutes

Showing traffic details from 1/25/2019, 9:02:00 AM to 1/25/2019, 9:05:00 AM. [Fetch more...](#)

Business Transaction Name	Count	Type
/movieztream_ui/customer/rate.htm	231	SERVLET
/movieztream_ui/login.htm	355	SERVLET
/movieztream_ui/movie/search.htm	220	SERVLET
/movieztream_ui/customer/rent.htm	226	SERVLET

APPDYNAMICS

If we want to ensure that there is always a BT for the MovieSearch functionality, we can potentially use the full (unique) URL, or at least some part which uniquely distinguishes it from other requests, to define a Custom Match Rule, as we will see in the coming demonstration and slides.



First, select the radio option of Custom Match Rule.

Next, select what Agent Type and Entry Point Type your rule is going to adhere to.

The required configuration on the next screen depends on what options you select on this first page, particularly which Entry Point Type you choose.

Custom Match Rules to Break out Key Transactions

Add Rule

Summary Rule Configuration 3

Rule Type: Custom Match Rule

Include/Exclude: 1 Include Transactions discovered by this rule
 Exclude Transactions discovered by this rule

Agent Type: Java

Entry Point Type: Servlet

Name: 2

Enabled:

Priority: 3

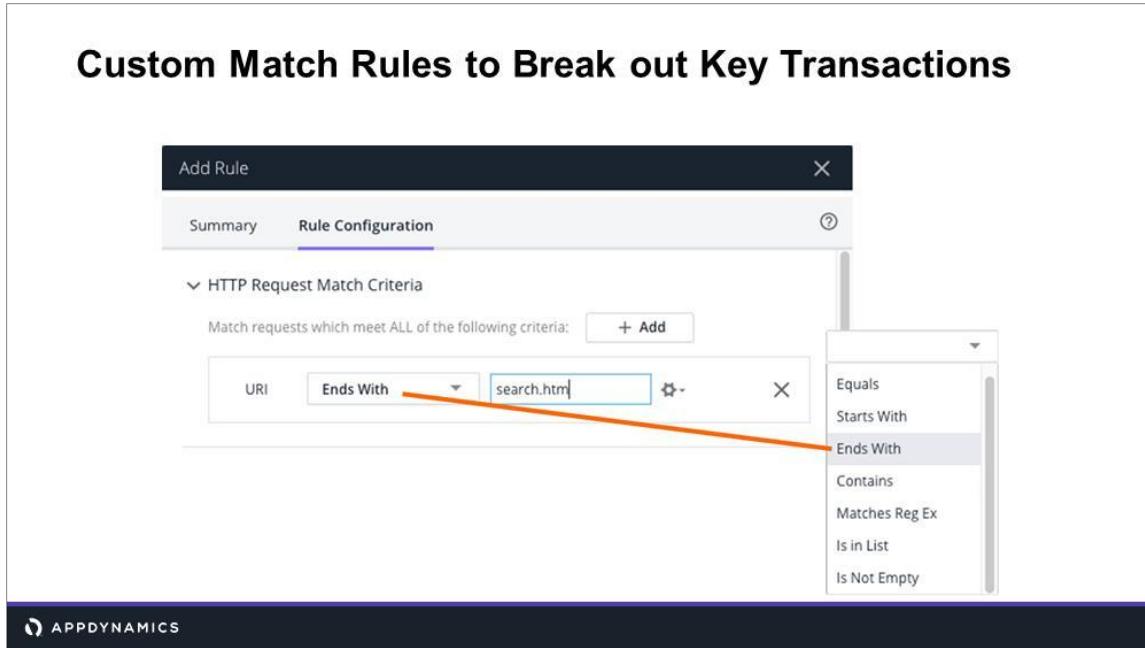
APPDYNAMICS

Now decide if it is an 'Include' rule (default) or an 'Exclude' rule.

Then provide a name for the BT. We were previously looking at the /moviezstream_ui/customer/rent.htm transaction so we could use 'Search Movie' for the name. Then select a Scope (which we will discuss soon).

Then click on the Rule Configuration tab.

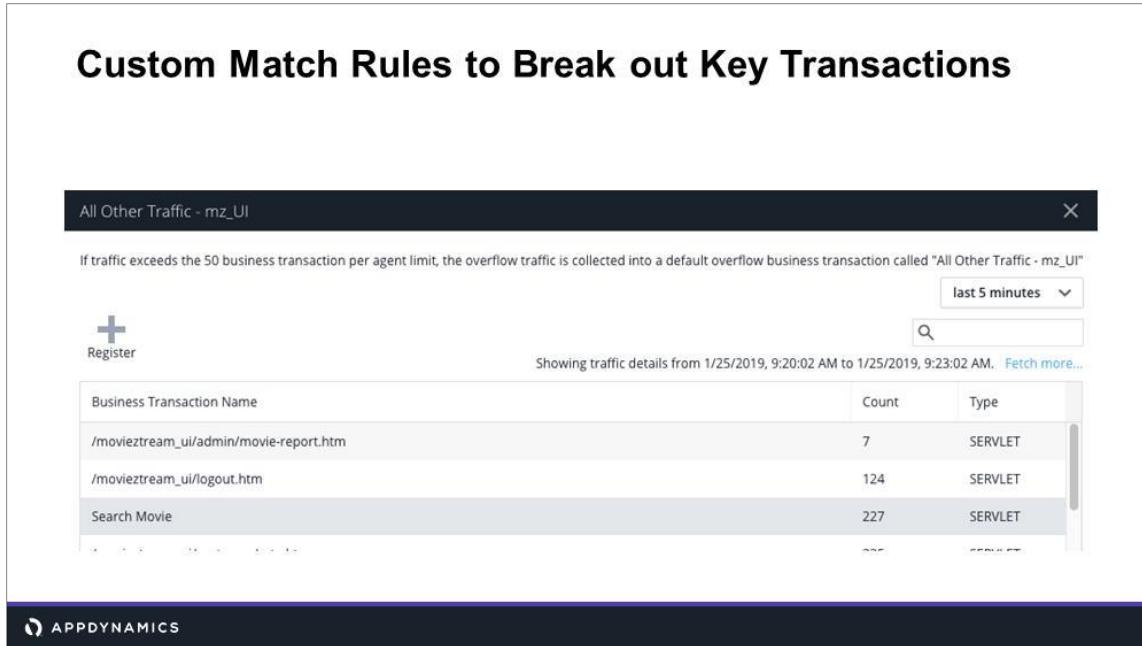
Custom Match Rules to Break out Key Transactions



The screenshot shows the 'Add Rule' dialog with the 'Rule Configuration' tab selected. Under 'HTTP Request Match Criteria', a dropdown menu is open, showing various match conditions. The 'Ends With' option is highlighted with a red arrow, indicating it is the selected condition for the URI field.

On the Rule Configuration tab you have options based on the Agent Type and Entry Point Type that you previously selected. If you chose Java and Servlet Entry Point Type, then you are presented with options to allow you to focus on the detection of the request based on some detail of the URI. Notice how you have a range of ways to specify the Match Criteria for the URI.

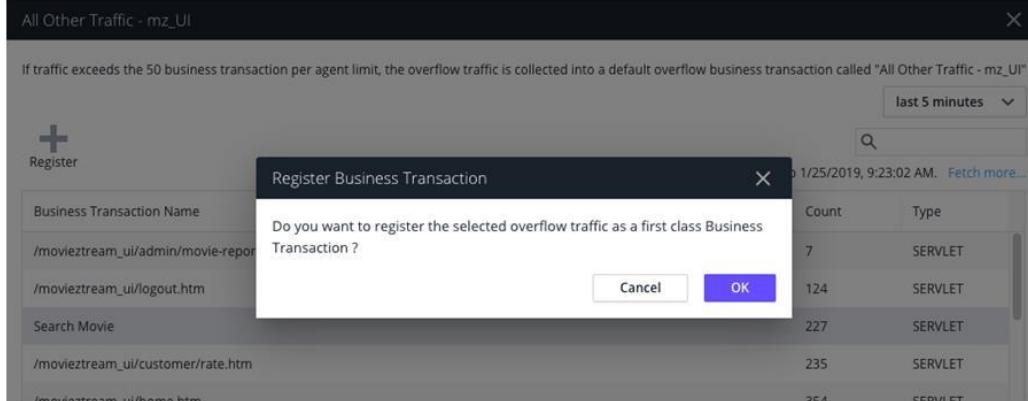
Custom Match Rules to Break out Key Transactions



Business Transaction Name	Count	Type
/movieztream_ui/admin/movie-report.htm	7	SERVLET
/movieztream_ui/logout.htm	124	SERVLET
Search Movie	227	SERVLET

After a few minutes (remember, the Agent-Controller Communication is always going to delay seeing immediate results), returning to the All Other Traffic details window should reveal the Search Movie BT showing up in the list, because BT Lockdown has been enabled.

Custom Match Rules to Break out Key Transactions



If traffic exceeds the 50 business transaction per agent limit, the overflow traffic is collected into a default overflow business transaction called "All Other Traffic - mz_UI"

last 5 minutes

1/25/2019, 9:23:02 AM. Fetch more...

Count	Type
7	SERVLET
124	SERVLET
227	SERVLET
235	SERVLET
354	SERVLET

Business Transaction Name: /moviezstream_ui/admin/movie-report

Do you want to register the selected overflow traffic as a first class Business Transaction?

Cancel OK

Search Movie: /moviezstream_ui/customer/rate.htm
/moviezstream_ui/home.htm

APPDYNAMICS

Now all that remains to do is to highlight the new BT and Register it.

Custom Match Rules to Break out Key Transactions

Business Transactions

	Name	He...	Response Time (ms)	Calls ↓	Calls / min
	All Other Traffic - mz_UI		43	1,182	296
	/movieztream_ui/login.htm		20	349	87
	Search Movie		99	233	58

APPDYNAMICS

Returning to the BT List page, our new BT will now show up alongside any other registered BTs and the All Other Traffic bucket within a few minutes.

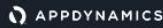
The difference now is that even if we deleted the Search Movie BT, or migrated the BT configuration to another environment, Search Movie would not need to be re-discovered and re-registered.

If the BT configuration were migrated to another Controller Application, and BT Lockdown was active in the new environment, then while the BTs with specific rules would be discovered, they would still need to be registered once more.

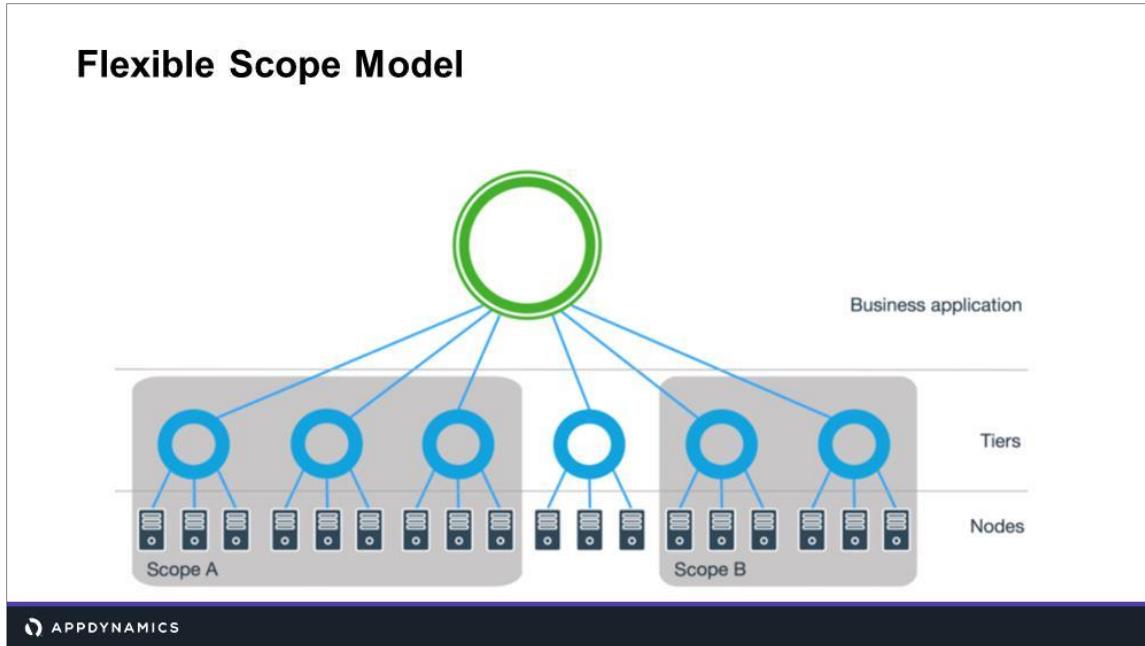
Scope Configuration Model

Scopes:

- Allow you to apply Business Transaction detection rules to any group of tiers
 - Default Scope includes all Tiers
 - Custom scopes include one or more tiers
- Reduce redundant operations
- Provide quicker and more flexible configuration of Business Transactions



A Scope is a way to define a component or set of components of the application landscape, so that Business Transaction Detection rules can be applied to those components as a whole.



Scopes can hold any combination of tiers within an application. A Default Scope, including all tiers in the application, is included out of the box.

A scope can only contain Agents of the same language type.

One set of rules can be applied everywhere or rules can be customized per scope.

Rules can be modified and/or copied from one scope to another.

Review Question - Answer

Which of the following can you do with Custom Match Rules?

- A) Create rules which explicitly define a Business Transaction
- B) Create rules that include the default entry point types and naming configuration for each app agent type
- C) Create rules to exclude specific transactions

Review Question - Answer

Which of the following are true of Business Transaction Priority?

- A) An Exclude rule with a priority of 20 will outrank an Exclude rule with a priority of 10.
- B) An Include Rule with a priority of 0 will outrank an Exclude Rule with a priority of 10.
- C) The lower the Priority number, the higher the precedence of the Rule.
- D) An Auto Detect rule with a priority of 30 will outrank an Include Rule with a priority of 0.

What You Learned

- How to define your company goals and the key steps for implementing AppDynamics
- The significance of prioritizing Business Transactions
- The principle of BT Entry and Exit Points
- How Controller communication works and the technique of Tag and Follow
- How Auto Discovery works and the range of technologies automatically discovered
- How to change default URL-based discovery rules
- How to select Business Transactions by hand
- How to create Custom Match Rules to control BT discovery
- How to use Scopes to control application of BT Rules

Differentiate yourself and your company with AppDynamics Certification

APPDYNAMICS | Certification Program

General Certification Information

<https://learn.appdynamics.com/certifications>

AppDynamics Certified Associate Performance Analyst

<https://learn.appdynamics.com/certifications/performance-analyst>

AppDynamics Certified Associate Administrator

<https://learn.appdynamics.com/certifications/administrator>

AppDynamics Certified Implementation Professional

<https://learn.appdynamics.com/certifications/implmenter>



To differentiate yourself and your company in an increasingly competitive market, please consider becoming AppDynamics certified.

For more information, please copy and paste this URL into a separate tab in your browser: <https://learn.appdynamics.com/certifications>

For information on specific certification tracks, please copy and paste the appropriate URLs below:

AppDynamics Certified Associate Performance Analyst:

<https://learn.appdynamics.com/certifications/performance-analyst>

AppDynamics Certified Associate Administrator:

<https://learn.appdynamics.com/certifications/administrator>

AppDynamics Certified Implementation Professional:

<https://learn.appdynamics.com/certifications/implmenter>

Managing Business Transactions (APM222)



University

APM222 - Managing Business Transactions

Core APM II: Advanced - Module 2

Objectives

Course

After completing this course, you will be able to:

- Employ multiple strategies for managing Business Transactions
- Determine when to use Service Endpoints instead of Business Transactions
- Replace BTs with Service Endpoints
- Split Business Transactions
- Use Live Preview while configuring Custom Match Rules
- Test BT rules in a Discovery Session
- Search for running Classes and Methods, and Uninstrumented Code
- Use Automatic and Custom backend detection

Labs

In this course's labs, you will:

- Aggregate and Promote Business Transactions using Custom Match Rules
- Rename and Prioritize Key Transactions
- Replace Business Transactions with Service Endpoints
- Create a Business Transaction in a Discovery Session
- Detect a Custom Backend

Business Transaction Management Strategies

Other Business Transaction Management Best Practices

Three Strategies

Strategy 1: Aggregating and Breaking out Business Transactions

Strategy 2: Business Transaction QA

Strategy 3: Prioritizing BTs

Historical 'Catch All' Business Transaction strategy

BT Best Practices: Strategy 1

Aggregating and Breaking out Business Transactions

Ways to manage when you really need to monitor 200+ functions

1. Aggregate via Custom Match Rules
2. Replace certain BTs with Service Endpoints



This is useful for existing environments where there is a lot of existing data, or large environments with lots of important functions.

If you believe there are 200 or more important function points that need to be monitored, there are a couple of options to consider.

1. Aggregate some fine-grained BTs together into a single one, by adjusting the BT Detection Rules.
2. Replace certain BTs with Service Endpoints (we'll look at this in a later topic).

BT Best Practices: Strategy 1

Aggregating and Breaking out Business Transactions

- Identify which current BTs could logically be aggregated together
- Consider what common characteristics might be used in a single Custom Match Rule
- Individual high priority functions can still be picked out

Name	Header	Response Time (ms)	Calls / min
/movieztream_admin	✓	2	12
/movieztream_admin/Account/LogOn	✓	201	7
/movieztream_admin/Payment/Refund	✓	1,197	7
/movieztream_admin/Customer	✓	1,278	5
/movieztream_admin/Account/LogOff	✓	1	4
/movieztream_admin/Payment/PaymentDone	✓	2	4
/movieztream_admin/Customer/Payments	✓	1,390	3
/movieztream_admin/Customer/Rentals	✓	2,662	3
/movieztream_admin/Rentals/Payments	✓	434	3
/movieztream_admin/Sales	✓	585	2

 APPDYNAMICS

In the example shown, we want to aggregate all requests starting with 'movieztream_admin/Customer' as a single BT, although we would still like to monitor the 'Payments' functionality independently. The first step will be to define an aggregating Custom Match Rule.

BT Best Practices: Strategy 1

Aggregating and Breaking out Business Transactions

Add a Scope

Create Scope

Name: M2 Admin UI Scope

Description:

Include the following Tiers: These specific Tiers

Type	Name
MT	Admin_UI

Selected Tiers (1)

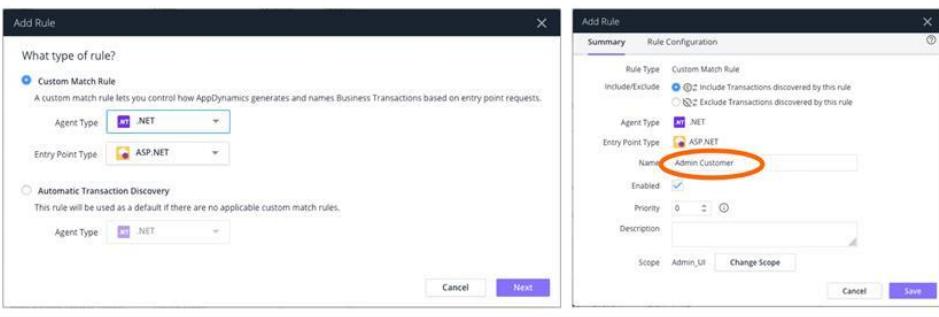
APPDYNAMICS

If this is the first time defining any Custom Match Rules for the Admin_UI Tier, then a Scope should be set up before creating the new Custom Match Rule (or the Scope can be added as a part of creating the rule).

BT Best Practices: Strategy 1

Aggregating and Breaking out Business Transactions

Create an aggregating Custom Match Rule



APPDYNAMICS

Knowing that the aggregating BT is going to be 'containing' requests that have part of their path in common, and are all hitting the ASP Server on the Admin_UI Tier, we can define a single Custom Match Rule accordingly.

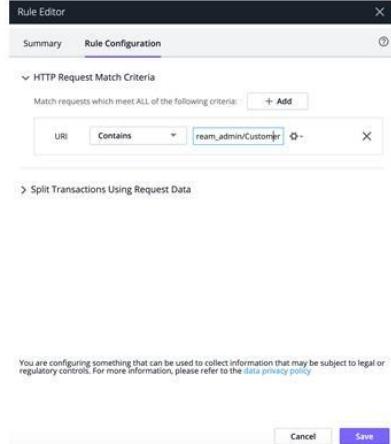
BT Best Practices: Strategy 1

Aggregating and Breaking out Business Transactions

Configure the Custom Match Rule

Specify suitably unique matching criteria in appropriate sections:

- HTTP Request Match Criteria
- Split Transactions Using Request Data (if required)



The screenshot shows the 'Rule Editor' window with the 'Rule Configuration' tab selected. Under 'HTTP Request Match Criteria', there is a single rule: 'Match requests which meet ALL of the following criteria: + Add' with 'URI Contains ream_admin/Customer'. Below this, a note states: 'You are configuring something that can be used to collect information that may be subject to legal or regulatory controls. For more information, please refer to the [data privacy policy](#)'. At the bottom are 'Cancel' and 'Save' buttons.

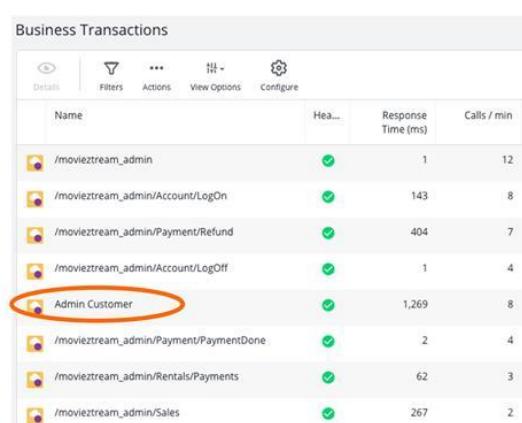
For the Custom Match Rule in this instance, we only need to specify HTTP Request Match Criteria that will pick up any requests containing the text 'moviezstream_admin/Customer'.

BT Best Practices: Strategy 1

Aggregating and Breaking out Business Transactions

Reduce the overall number of BTs.

Old Business Transactions are replaced with new, singular, aggregating Business Transaction.



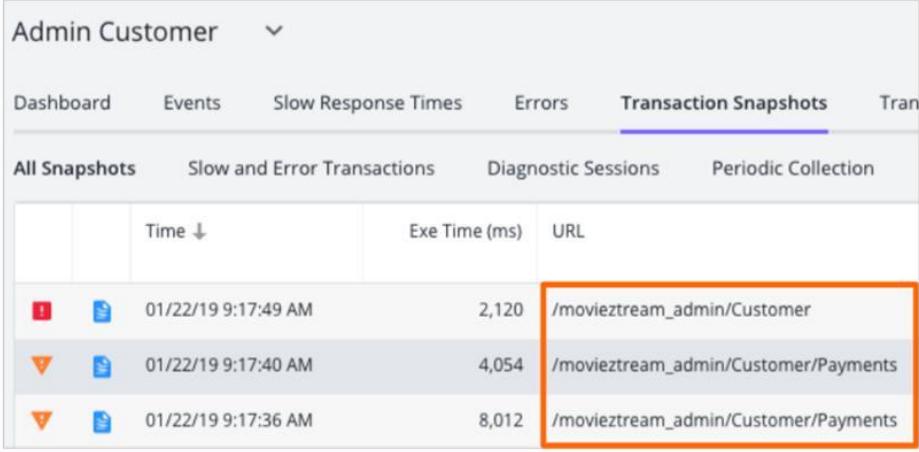
Name	He...	Response Time (ms)	Calls / min
/movietream_admin	✓	1	12
/movietream_admin/Account/LogOn	✓	143	8
/movietream_admin/Payment/Refund	✓	404	7
/movietream_admin/Account/LogOff	✓	1	4
Admin Customer	✓	1,269	8
/movietream_admin/Payment/PaymentDone	✓	2	4
/movietream_admin/Rentals/Payments	✓	62	3
/movietream_admin/Sales	✓	267	2

APPDYNAMICS

After the usual few minutes waiting for the Agent(s) to adopt the new BT Detection configuration, eventually the new aggregating BT will show up.

BT Best Practices: Strategy 1

Aggregating and Breaking out Business Transactions



	Time ↓	Exe Time (ms)	URL
!	01/22/19 9:17:49 AM	2,120	/movieztream_admin/Customer
▼	01/22/19 9:17:40 AM	4,054	/movieztream_admin/Customer/Payments
▼	01/22/19 9:17:36 AM	8,012	/movieztream_admin/Customer/Payments

APPDYNAMICS

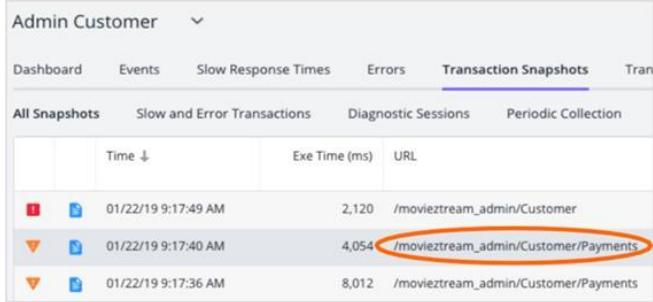
Any time that you want to check and verify the actual full URLs, you need only go and look at the Snapshots page for a particular Business Transaction.

Note the different URLs in the details for the Admin Customer snapshots.

BT Best Practices: Strategy 1

Aggregating and Breaking out Business Transactions

Break out Customer Payments



	Time ↓	Exe Time (ms)	URL
01/22/19 9:17:49 AM	2,120	/movieztream_admin/Customer	
01/22/19 9:17:40 AM	4,054	"/movieztream_admin/Customer/Payments" 	
01/22/19 9:17:36 AM	8,012	/movieztream_admin/Customer/Payments	

APPDYNAMICS

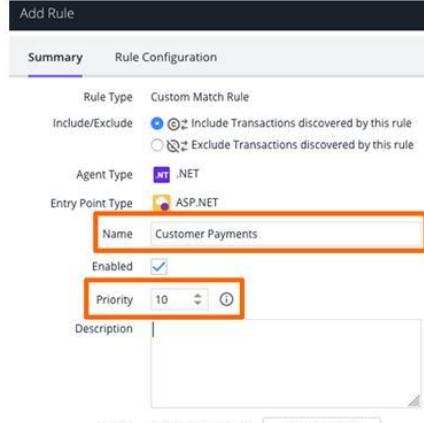
We just have to look at the URL for `/movieztream_admin/Customer/Payments` to see that it differs from requests such as `/movieztream_admin/Customer` in the Transaction Snapshots list page. Armed with this information, we can plan to create a specific BT rule to detect just the Payments requests.

BT Best Practices: Strategy 1

Aggregating and Breaking out Business Transactions

Define a specific Custom Match Rule for unique requests

- Set up new Custom Match Rule
- Provide appropriate name
- Set Scope accordingly
- Add Priority > 0, in case there is chance of overlapping rule



APPDYNAMICS

Adding a value > 0 for the Priority is a best practice tip, rather than a strict technical requirement.

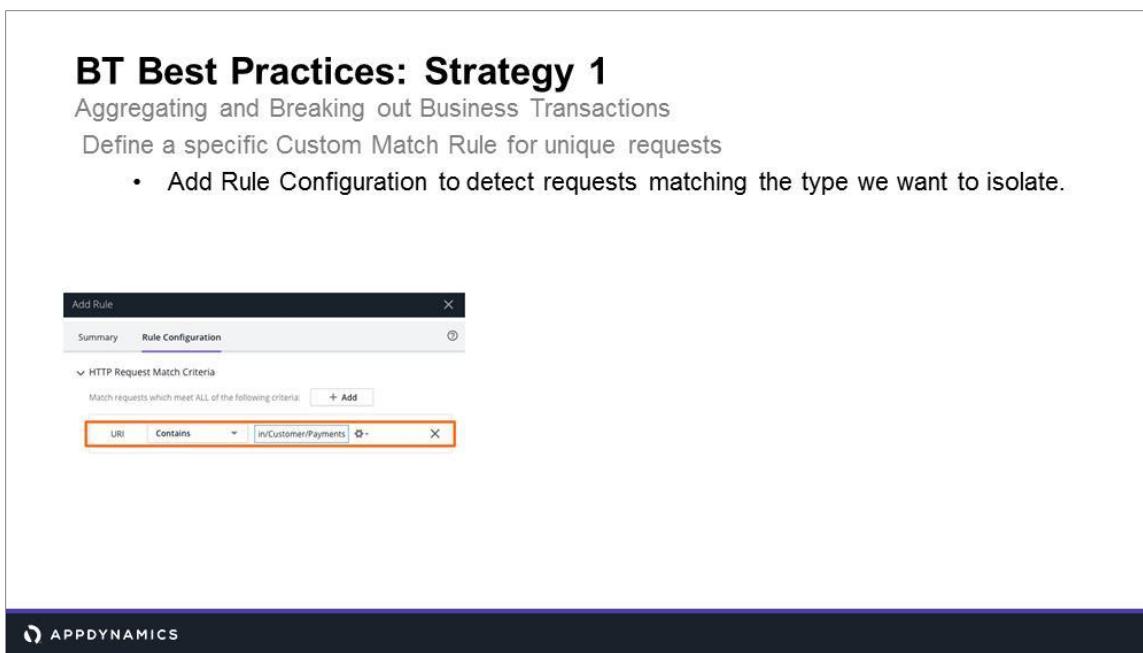
If any other rule, either currently in the Controller, or in future, were also to detect this transaction, our BT Custom Match Rule with the higher priority will “win”.

BT Best Practices: Strategy 1

Aggregating and Breaking out Business Transactions

Define a specific Custom Match Rule for unique requests

- Add Rule Configuration to detect requests matching the type we want to isolate.



The screenshot shows the 'Add Rule' dialog box with the 'Rule Configuration' tab selected. Under 'HTTP Request Match Criteria', there is a list with one item: 'Match requests which meet ALL of the following criteria.' followed by an 'Add' button. The item listed is 'URI Contains /v1/Customer/Payments'. The AppDynamics logo is visible at the bottom of the interface.

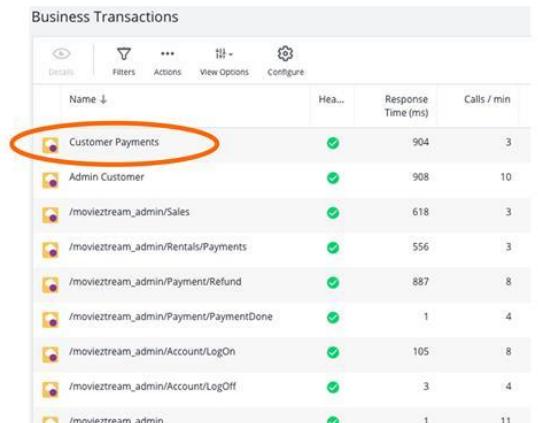
In the example shown, note that the HTTP Request Match Criteria of this new rule is more precise than the aggregating rule we previously defined, and seeks to isolate just the Customer Payment requests.

BT Best Practices: Strategy 1

Aggregating and Breaking out Business Transactions

Verify the new Business Transaction

- Confirm new BT shows up
- Check Snapshots to verify no unwanted requests show up



Name	Health	Response Time (ms)	Calls / min
Customer Payments	Green	904	3
Admin Customer	Green	908	10
/movieztream_admin/Sales	Green	618	3
/movieztream_admin/Rentals/Payments	Green	556	3
/movieztream_admin/Payment/Refund	Green	887	8
/movieztream_admin/Payment/PaymentDone	Green	1	4
/movieztream_admin/Account/LogOn	Green	105	8
/movieztream_admin/Account/LogOff	Green	3	4
/movieztream_admin	Green	1	11

APPDYNAMICS

The final step is to review the new BT and ensure that it is not detecting traffic that you didn't expect, meaning that the Custom Match Rule criteria might need refining.

BT Best Practices: Strategy 2

QA Strategy

Discover BT Rules one at a time

1. Set up agents on Test systems
2. Define separate AppDynamics Application for Test systems
3. Execute an individual function you want to monitor
4. See if anything detected through Auto Discovery Rules
5. Define Custom Match Rule to deliberately detect the request
6. Continue process for all key functions

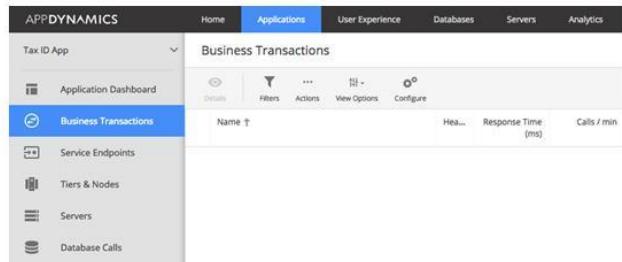


In a pre-Production environment such as QA, you can run particular functions one at a time and work out specific Custom Match Rules in a controlled environment, without the worry of all traffic hitting the server all the time. This allows you to build up an exact set of Rules which should find just what you want to monitor, that you can then migrate to your Prod env, using e.g., Config Exporter.

BT Best Practices: Strategy 2

QA Strategy

The starting point for the QA strategy should be to have no BTs showing up at all.



The screenshot shows the AppDynamics Applications dashboard. The main navigation bar is at the top with tabs: Home, Applications (which is selected and highlighted in blue), User Experience, Databases, Servers, and Analytics. Below the navigation is a dropdown menu for the application 'Tax ID App' with options: Application Dashboard, Business Transactions (which is also highlighted in blue), Service Endpoints, Tiers & Nodes, Servers, and Database Calls. The main content area is titled 'Business Transactions' and contains a table with the following columns: Name (sorted by ascending name), Headers, Response Time (ms), and Calls / min. There are no rows in the table.

In this strategy, the idea is to start with no BTs at all, and no traffic causing App Agents to detect BTs via the Auto Discovery Rules. Keep in mind that you can always enable BT Lockdown and then delete existing BTs if you want to get back to a clean slate.

BT Best Practices: Strategy 2

QA Strategy

Run functionality for a specific Business Transaction

Example: Login

- Invoke Login on the instrumented Test app
- If running a test script, ensure that it performs no other functions



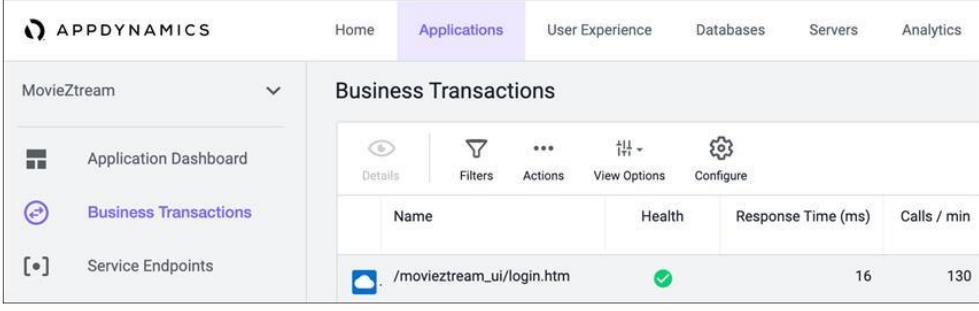
APPDYNAMICS

We need to ensure that we don't run any other functionality after login, and we also don't want to execute an obscure or specialized version of the Login, which might not accurately represent the Entry Point.

BT Best Practices: Strategy 2

QA Strategy

- Trigger more traffic for the specific functionality
- Verify if Custom BT shows up as it should



Name	Health	Response Time (ms)	Calls / min
/moviezstream_ui/login.htm	✓	16	130

The desired result is to have only the targeted BT show up in the Business Transactions list.

BT Best Practices: Strategy 2

QA Strategy

Continue on to the next functionality and repeat the process to select the next targeted BT.



The screenshot shows a web application titled "MOVIEZTREAM" with a marquee-style header. The main menu includes "Home", "Customers", "Movies" (which is the active tab), and "Admin". A search bar is present with fields for "Description" and "Category". The search results table has columns for "Title", "Description", and "Category". The data in the table is as follows:

Title	Description	Category
AMADEUS HOLY	A Emotional Display of a Pioneer And a Technical Writer who must Battle a Man in A Balloon	
AMERICAN CIRCUS	A Heartwarming Drama of a Girl And a Astronaut who must Face a Database Administrator in A Shark Tank	
ANTITRUST TOMATOES	A Fateful Yarn of a Womanizer And a Feminist who must Succumb a Database Administrator in Ancient India	

BT Best Practices: Strategy 2

QA Strategy

Check that the new BT Rule works before moving on to invoke the next function and its Custom Match Rule



Name	Original Name	He...	Response Time (ms)	Calls / min
1- Login	/movieztream_ui/login.htm	!	1,715	0
3- Search Movie	/movieztream_ui/movie/search.htm	✓	361	2

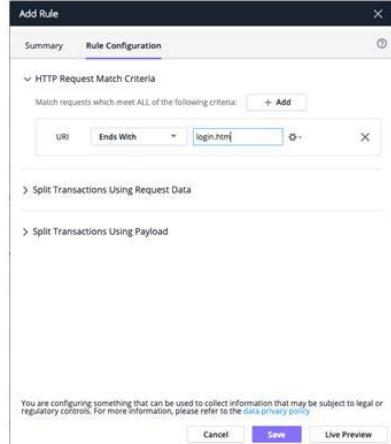
Iterate the process and check each rule before moving on to ensure you are capturing only the activity you want.

BT Best Practices: Strategy 2

QA Strategy

Define Rule Configuration for new Custom Match Rule

- Match Criteria should only detect desired request types



APPDYNAMICS

Once we have figured out what our Match Criteria need to be to give us a BT that detects only traffic for the transactions we want it to, then we can set up the Custom Match Rule and see if it works as we expect.

Even if existing Auto Discovery Rules might detect these requests, the BT Rule Precedence logic ensures that a Custom Match Rule will match before any Auto Discovery Rules.

BT Best Practices: Strategy 3

Prioritizing BTs

Scenario

- Your ops team got the CIO out of bed for a relatively minor issue that looked like a major issue.
- How do you help everyone understand what the different BTs are and what priority they have?

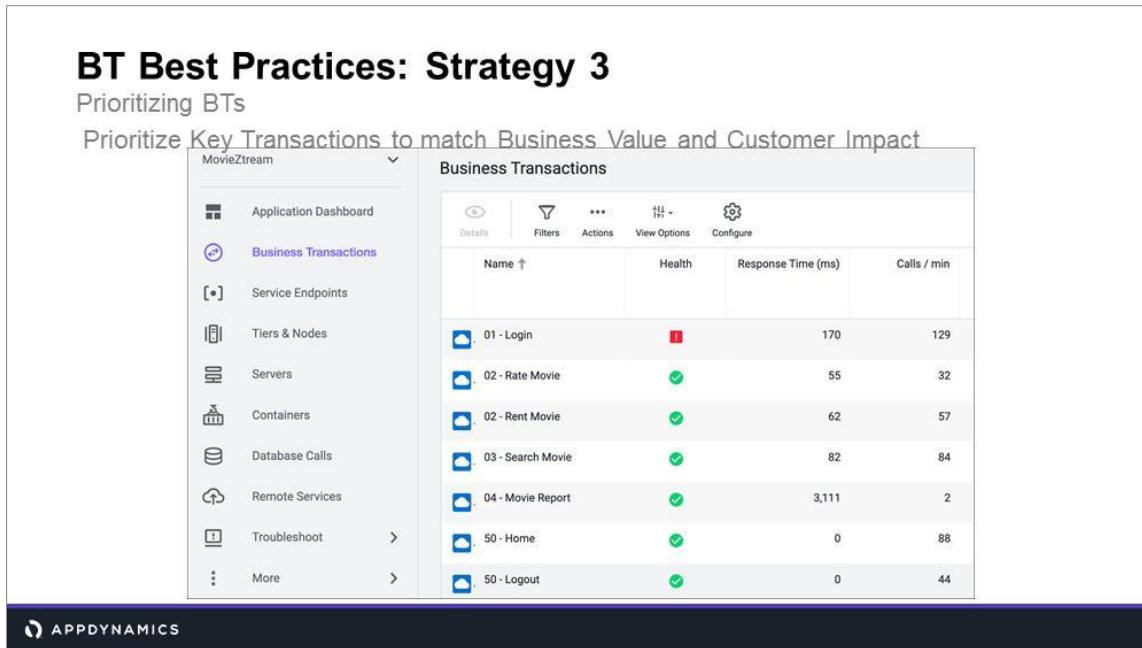


To wrap up BT Best Practices, let's look at how you can organize your newly-discovered BTs and name them to help everyone understand which matter most and which don't need the CIO to be woken up during the night.

BT Best Practices: Strategy 3

Prioritizing BTs

Prioritize Key Transactions to match Business Value and Customer Impact



Name	Health	Response Time (ms)	Calls / min
01 - Login	!	170	129
02 - Rate Movie	✓	55	32
02 - Rent Movie	✓	62	57
03 - Search Movie	✓	82	84
04 - Movie Report	✓	3,111	2
50 - Home	✓	0	88
50 - Logout	✓	0	44

APPDYNAMICS

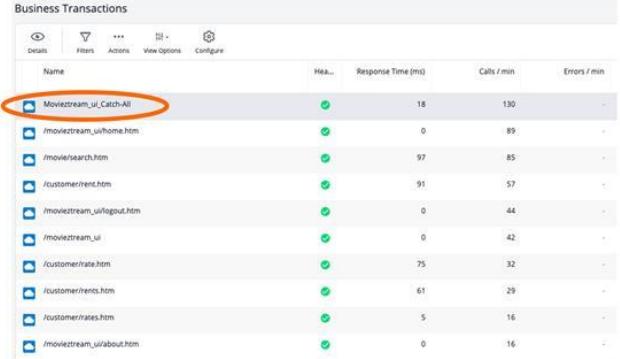
Identifying the most important BTs with a clear naming strategy is a great way to ensure that everyone understands which BTs matter most. It means that when someone less familiar with AppDynamics sees some performance details in the Controller, they also get a clear indicator of whether or not the situation needs urgent attention or is less critical.

In the example in the slide, the ability for customers to log in to the app is clearly the highest priority, and hence named as such, which also has the useful benefit of popping to the top of any list of BTs when sorted by name.

BT Best Practices: Legacy Catch-all Approach

No longer recommended

- Prior to BT Lockdown, the Catch-all method was used to pick up whatever didn't get detected by more specific BT Rules.
- Rules would be prioritized so that important BTs are captured first.
- All other BTs would then fall into the Catch-all BT



Name	Health	Response Time (ms)	Calls / min	Errors / min
Moviestream_ui_Catch-All	Green	18	130	-
/moviestream_ui/home.htm	Green	0	89	-
/movie/search.htm	Green	97	85	-
/customer/rent.htm	Green	91	57	-
/moviestream_ui/logout.htm	Green	0	44	-
/moviestream_ui	Green	0	42	-
/customer/rate.htm	Green	75	32	-
/customer/rents.htm	Green	61	29	-
/customer/rates.htm	Green	5	16	-
/moviestream_ui/about.htm	Green	0	16	-

APPDYNAMICS

This practice was used frequently before we added BT Lockdown to the system. It is good to be aware of in case it was used in your environment. If you are/have a new customer/environment, this can be ignored.

Custom Match rules can have priorities assigned to them. This allows you to create rules that create individual Business Transactions. Higher priority numbers get evaluated first. The idea is that the “catch-all” rule will get evaluated last and will catch all of the Business Transactions that do not match any of the previous rules.

If deployed correctly this methodology will only create Business Transactions for which there are Custom Match rules and will never create an “All Other Traffic – tiername” category. This is because the “catch-all” rule will catch all of the overflow Business Transactions and they will never fall into the “All Other Traffic - tiername” category.

For example, these Custom Match Rules will create individual Business Transactions for BDR-Product-Furniture, BDR-Product-Indoor, BDR-Product-Outdoor and BDR-Quote-Request. All other Servlets will fall into the BDR-Servlet-Catchall Business Transaction.

Review Question - Answer

Match the transaction with the number that would most likely be used at the beginning of its name.

Transaction	Number
02 - Shopping Cart Checkout	02
50 - Logout	50
01 - Login	01
50 - Project_Manager.jpg Load	50

Review Question - Answer

Which of the following are true statements about how AppDynamics handles BT detection?

- A) A Business Transaction can only map to one detection rule.**
- B) Creating a group of transactions in the Controller eliminates the ability to see each transaction individually.
- C) Deleting a Business Transaction prevents it from appearing any longer, whereas excluding a BT gets rid of the entire history.



On an incoming request, only one detection rule will fire to create the Business Transaction. Whichever rule fires first, based on precedence, will create the BT.

Excluding a BT means that it is no longer monitored but does not delete the history. Deleting a BT disposes of the history to this point in time, but does not stop it reappearing.

Service Endpoints

Service Endpoints

Use Case Examples

Monitor how a particular service is performing

- Provide Dashboards to monitor one particular piece of the system

Replace Business Transactions

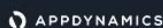
- Collect metrics without the overhead of BTs

Monitor performance of vendor services

- Keep an eye on Third Party Providers in case they switch you to a slower system

Monitor performance by region

- Split Service Endpoints according to varying territory values



Here are some examples of when Service Endpoints are useful. We'll discuss the first three in this section and then look more closely at the performance by region example when we talk about agent properties.

Service Endpoints

Use Case: Monitoring a particular service

- Monitor performance at a specific point
- No visibility of downstream activity
- Can monitor anything, not just 'services'
- Not limited to monitoring where a service 'ends' (think of them more as 'Service Points')

The diagram illustrates the concept of Service Endpoints. It shows two external entities, 'Client web' and 'Partner web', represented by light blue squares. Arrows point from both of these entities to a central component labeled 'Payments Engine', which is highlighted with an orange box. This central component is situated within a larger box labeled 'Payment Processing'. A callout box to the right of the central component states: 'E.g. Independent monitoring of key service used by multiple components'. The AppDynamics logo is visible at the bottom left of the slide.

Service Endpoints allow you to obtain a subset of metrics and associated snapshots for a particular service so that you can focus on the information truly of interest to you, as opposed to metrics across an entire Business Transaction or entire Tier.

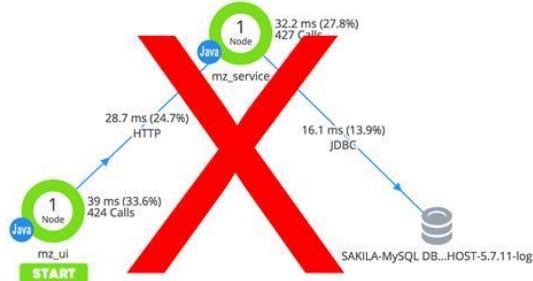
For example, you may be in charge of an authentication piece that services the Login Business Transaction as well as the Upsell Business Transaction. In that case, neither Business Transaction can give you pure performance information of your piece.

Service Endpoints are also commonly used in place of Business Transactions. Adding additional BTs over the 200 limit can negatively affect performance and ease of monitoring. Adding a Service Endpoint gives you some of the metric collection of a BTs without the overhead.

Service Endpoints

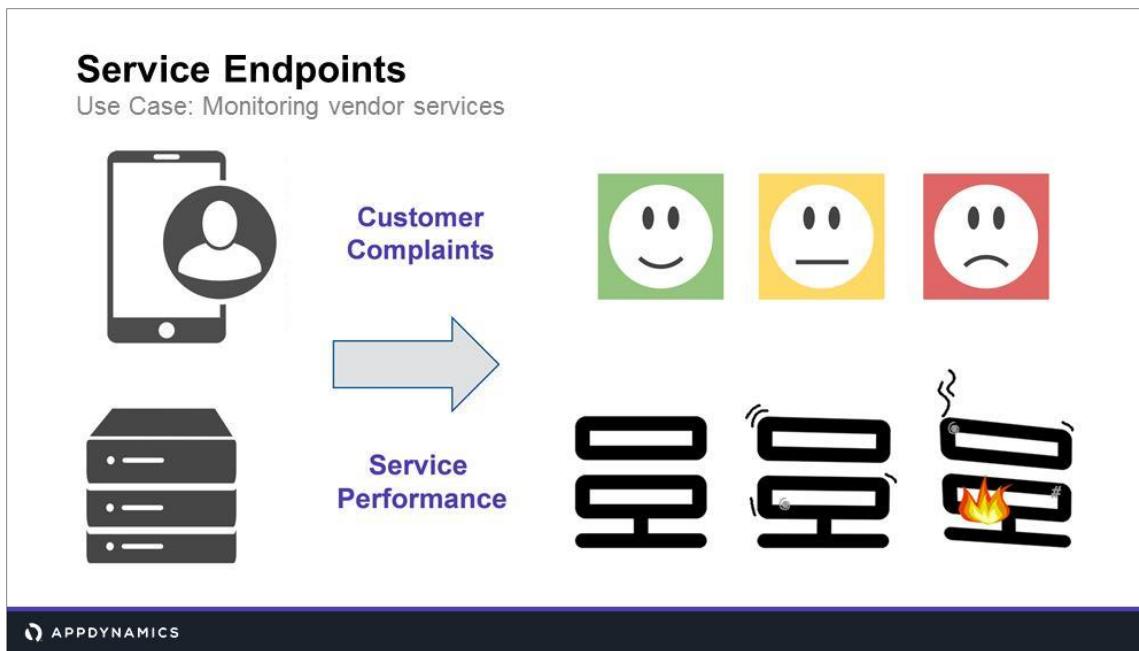
Use Case: Replace Business Transactions

- Service Endpoints do not track downstream activity
- So you get NO FLOW MAPS
- You get no Snapshots, though the Service Endpoint may feature in a BT Snapshot
- No User Experience Thresholds, as with a BT



A Business Transaction offers a view of application performance that cuts across the environment, reflecting all services that participate in fulfilling the transaction. Another way of viewing application performance, however, focuses on the performance of a particular service independently of the BTs that use it.

Like Business Transactions, Service Endpoints give you key performance indicators, metrics, and snapshots. However, they provide that information exclusively in the context of that service, omitting BT context or downstream performance data.

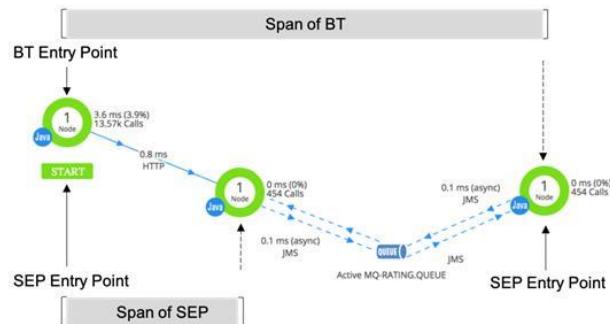


Service Endpoints provide a great, lightweight way to make sure that a vendor you may be relying on isn't secretly moving you to a slower server over time, to save themselves money, while continuing to charge you the same for what you got when you first signed up with them!

This can happen - vendors hook you in with superfast service, which means your app runs great for the first 3 months, then they move servers around and you end up on the Level-3 performance environment and your app consequently runs slower. How can you tell where the performance drop-off is coming from? Set up a SEP against the call out to the vendor service from Day 1, and put some Alerting in place so you know when vendor performance degrades noticeably.

Scenario: Combining BTs and SEPs

- BT for overall end to end performance
 - All hops, including through MQ
- SEPs for monitoring performance of discreet functions
 - Only includes performance of synchronous hops



It is important to appreciate that there is nothing to prevent the setup of a Business Transaction as well as a separate Service Endpoint that monitor the same Entry Point.

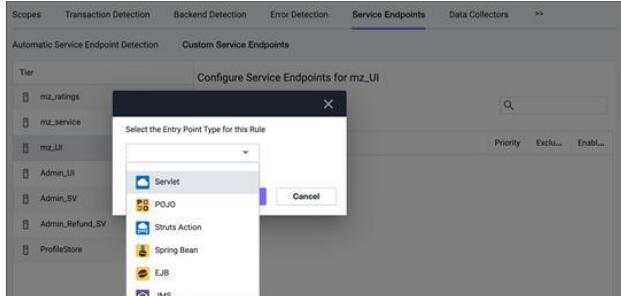
In the above use case, the first SEP provides us with visibility of the performance of the first 'leg' of the request, up to the point where the asynchronous call to MQ is made, whereas the BT allows us to monitor the full request journey from the same starting point but right across the asynchronous MQ segment and beyond.

The final (furthest right) SEP gives us independent monitoring of the performance of functionality which happens AFTER the MQ segment.

Configuring a Service Endpoint

Defining a Custom Service Endpoint

- Set up a Custom Service Endpoint from **Custom Service Endpoints** tab
- Select **Tier**, then choose the **Entry Point Type**
- Subsequent Configuration parameters depend on selected Entry Point Type



APPDYNAMICS

Like with Business Transactions, Service Endpoints can be discovered automatically by AppDynamics. They are then listed under Service Endpoints > Automatic Service Endpoint Detection. From here you can access the dashboard view of any Service Endpoint. Be aware of Automatically Detected Service Endpoints' potential to overlap with BTs.

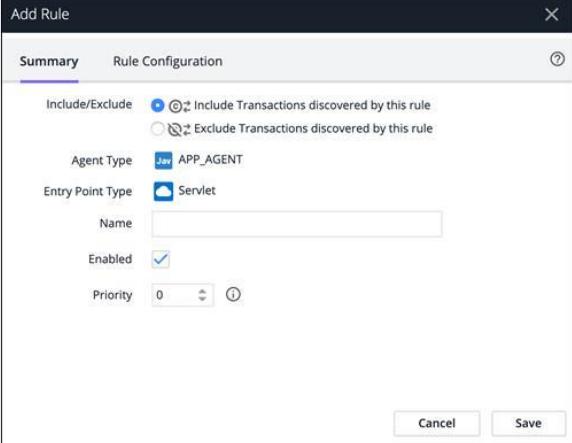
More commonly though, Service Endpoints are configured manually by going to: Configuration > Instrumentation > Service Endpoints > Custom Service Endpoints to define a rule.

Now you can access basic metrics such as calls-per-minute, average response time, and errors ONLY for this service. You can focus exclusively on requests that pass through your service endpoint.

Configuring a Service Endpoint

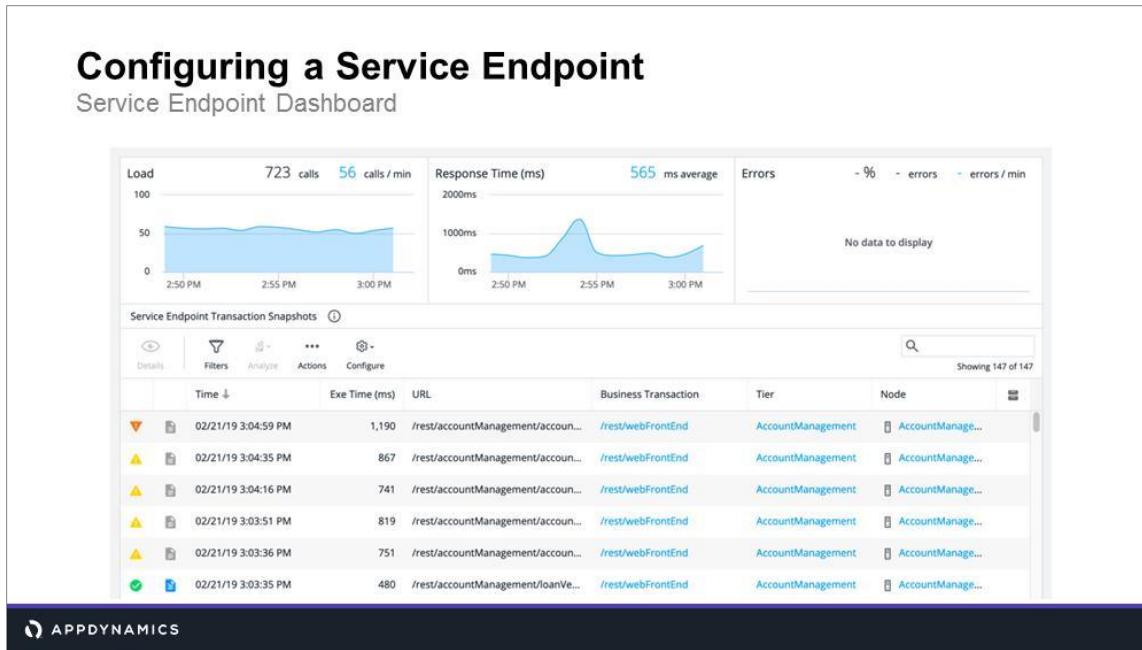
Defining a Custom Service Endpoint

- Define Name and Priority (optional)
- Specify Transaction Match Criteria



The screenshot shows the 'Add Rule' dialog box with the 'Summary' tab selected. It includes fields for 'Include/Exclude' (with 'Include Transactions discovered by this rule' selected), 'Agent Type' (Java APP_AGENT), 'Entry Point Type' (Servlet), 'Name' (empty text box), 'Enabled' (checked), 'Priority' (0), and 'Save' and 'Cancel' buttons.

The range of options for configuring the Match Criteria will mirror the options available when defining a new Business Transaction Custom Match Rule.

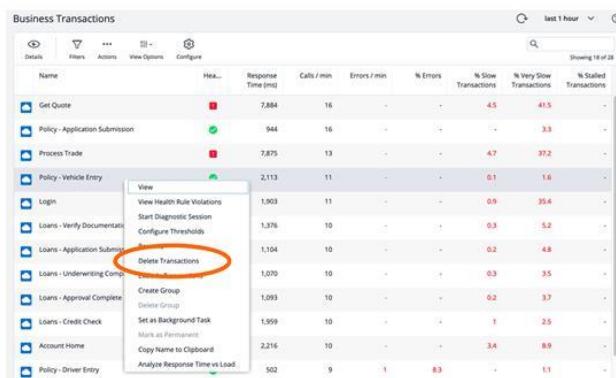


With a Service Endpoint, all you really get is the Load, Average Response Time, and Number of Errors.

The fact that you see Snapshots listed here merely reflects the fact that in this example there are Business Transactions which happen to flow through the Service Endpoint. Because of this, AppDynamics lists out any Snapshots for you, although these are not Service Endpoint Snapshots, they are always BT Snapshots.

Replacing a BT with a Service Endpoint

- Right click – Delete Transactions
- Remove Custom Match Rule



Name	Health	Response Time (ms)	Calls / min	Errors / min	% Errors	% Slow Transactions	% Very Slow Transactions	% Stalled Transactions
Get Quote	Red	7,884	16	-	-	45	41.5	-
Policy - Application Submission	Green	944	16	-	-	-	3.3	-
Process Trade	Red	7,875	13	-	-	4.7	37.2	-
Policy - Vehicle Entry	Green	2,113	11	-	-	61	1.6	-
Login	Green	1,903	11	-	-	0.9	35.4	-
Loans - Verify Documentation	Green	1,376	10	-	-	0.3	5.2	-
Loans - Application Submission	Green	1,104	10	-	-	0.2	4.8	-
Loans - Underwriting Complete	Green	1,070	10	-	-	0.3	3.5	-
Loans - Approval Complete	Green	1,093	10	-	-	0.2	3.7	-
Loans - Credit Check	Green	1,959	10	-	-	1	2.5	-
Account Home	Green	2,216	10	-	-	3.4	8.9	-
Policy - Driver Entry	Green	502	9	1	63	-	1.1	-

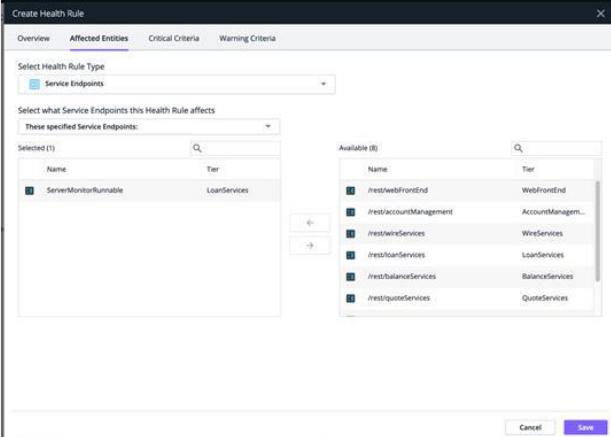
APPDYNAMICS

When replacing a BT with a Service Endpoint, be sure to delete the BT, as opposed to Excluding it. If you exclude the transaction then no snapshots will be gathered. Delete the transaction and then it will roll up into the All Other Traffic bucket. That will enable troubleshooting if needed.

Replacing a BT with a Service Endpoint

Define Alerting for replacement Service Endpoint

- Health Rules are not limited to Business Transactions



APPDYNAMICS

Review Question - Answer

A Service Endpoint Snapshot is captured every 10 minutes by default.

True

False



Snapshots are not captured for Service Endpoints, only for Business Transactions.

Review Question - Answer

Which of the following statements about Service Endpoints are accurate?

- A) They are well-suited to providing lightweight, basic metrics about less crucial functionality.
- B) The range of options for configuring a Service Endpoint's detection logic is equivalent to the options when configuring BT detection rules.
- C) You must delete a BT which has been replaced by a Service Endpoint.

Live Preview in Custom Match Rules

Splitting Business Transactions

Use Request values to deliver distinct BTs from a Rule

- Match Criteria can result in multiple BTs
- Information from the entry point call can be used to split matching transactions into multiple BTs
 - i.e., Servlets/ASP can be split by HTTP parameters, cookie values and other HTTP Request data
 - i.e., POJO/POCO can be split by method parameters, method name
- Be aware of possible outcome of BT Rule Logic.
 - i.e., BT Explosion

Live Preview is valuable in helping define appropriate BT Rules (test rules in Java only; create new rules in either Java or .NET)



The Matching Criteria available when defining a BT Rule (particularly in the case of Servlets/ASP and POJO/POCO) are pretty sophisticated and allow you a wide range of options on what you can detect with a single BT Rule.

While this is very beneficial, as you don't necessarily need to define one Rule per BT, it also means that there are risks if you use Matching Criteria that don't give you exactly what you think you should see.

Live Preview and BT Discovery Sessions both offer a safety net, or sandbox-type environment, where you can explore your actual results depending on your Rule configuration BEFORE saving the configuration and watching what happens to your Business Transactions.

There are two "modes" within Live Preview: testing an existing BT or creating a new BT. Please note that when it comes to splitting a BT, the only way to generate a split within Live Preview is to be in the creating a new BT "mode."

Live Preview in Custom Match Rules (Java only)

Introduction

- Test BT Rules before moving them to Production to avoid the possibility of a BT Explosion
- Fix typos quickly when defining new BT Rule before they propagate in Production
- Test BT Rule changes safely in a Pre-Production environment

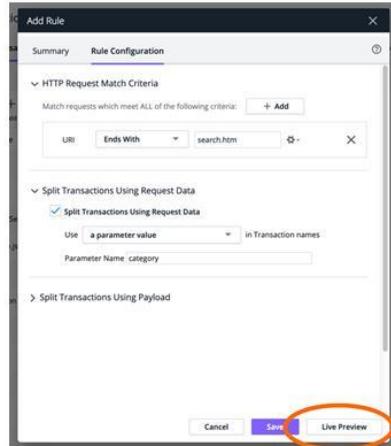


The default transaction discovery capabilities in the AppDynamics platform can get you up and running quickly. However, in most cases, you will want to tune and refine how Business Transactions are discovered. This ensures you are monitoring the Business Transactions that reflect the traffic that is most important to you.

Live Preview in Custom Match Rules

Preview Java Business Transactions

- Works with Java Servlet and POJO Custom Match Rules
- Ability to test out rule logic against real-time application load
- Make adjustments to rule logic and get instant feedback
- Available from Rule Configuration tab

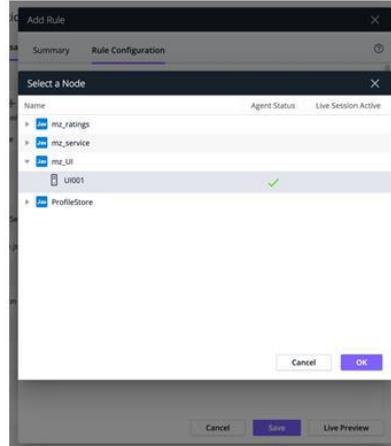


Live Preview for Custom Match Rules is not available for .NET Rules at this time.

Live Preview in Custom Match Rules

Preview Java Business Transactions

- Select Node appropriate to expected BT Entry Point



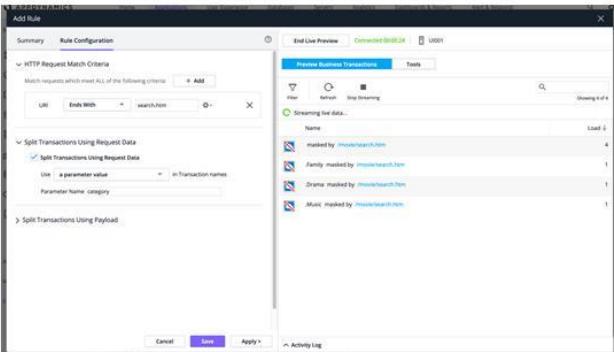
APPDYNAMICS

The Node must be specified so that AppDynamics knows where the real-time load data should be streamed from, for the BT Rule logic that you want to verify.

Live Preview in Custom Match Rules

Preview Java Business Transactions

- Preview results to confirm correct configuration
- Adjust rule logic on-the-fly
- Confirm against real-time app traffic
- Masked transactions show when calls are part of a current Business Transaction
- Save when satisfied



The screenshot shows the AppDynamics interface. On the left, the 'Rule Configuration' window is open, showing a configuration for 'HTTP Request Match Criteria' with a condition 'Ends With' for 'search.htm'. On the right, the 'Live Preview' window is open, showing a list of transactions: 'masked by movieSearch.htm' (4), 'Family masked by movieSearch.htm' (1), 'Drama masked by movieSearch.htm' (1), and 'Music masked by movieSearch.htm' (1). The 'Streaming live data...' button is highlighted in green.

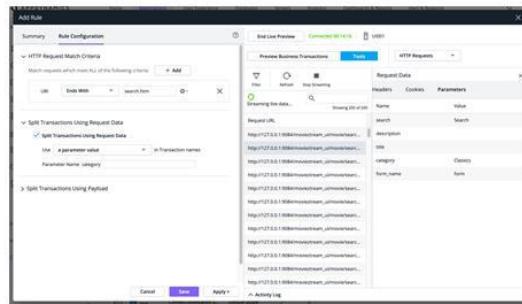
APPDYNAMICS

Click Save to be prompted to apply the new rule changes to the current 'active' set of rules. You can choose to apply or discard rule changes at this point.

Live Preview in Custom Match Rules

Inspecting Servlet Request Data in Live Preview

- Inspect detailed data from live requests
- Right-click a value to add it to the Match or Split criteria
- Data is available per request or grouped



Live Preview in Custom Match Rules

Inspecting Servlet Request Data in Live Preview

All HTTP Requests

- Headers, Parameters, Cookies
- Filtering

Aggregated Data

- URI
- Headers
- HTTP Parameters
- Hostname
- Port
- Class Name
- Servlet Name
- Cookie

© APPDYNAMICS

Selecting an option on the drop-down gives you visibility of all the values that AppDynamics is noticing for the category during the Live Preview session.

Review Question - Answer

When using Live Preview you must save your changes or they will not persist.

True

False



On quitting the Live Preview feature, you have the choice of whether you would like to discard the changes, or apply them.

Managing Business Transactions Using Discovery Sessions

BT Discovery Sessions

Capabilities

- Explore Real-time traffic to help define Business Transaction Rule logic
- Preview results of creating or updating Business Transaction Rules in a safe environment (Java only)
- Look up specific Classes and review Methods running in JVM/CLR and easily define new POJO/POCO Business Transaction Rules
- Find Uninstrumented Code running in JVM/CLR and easily define new POJO/POCO Business Transaction Rules



Adding or changing BT Rules in a Production scenario is somewhat risky. For example, you wouldn't want to put a BT Rule into Production that gives you 50 new BTs, when you only expected 2 from your Rule logic. The Transaction Discovery tool (also known as Live Preview) lets you refine Business Transaction discovery in a safe environment and test before you deploy.

During a Business Transaction Discovery session, you can see the effect of the changes you make to the configuration without having to apply it to the active configuration. When you've finished making your changes, you can then choose which tiers of the application to apply the tuned configuration to or not apply the changes at all.

BT Discovery Sessions

Launching a new session

Instrumentation

Scopes Transaction Detection Backend Detection Error Detection Service Endpoints Data Collectors Call Graph Settings JMX >>

Rules Tiers More

Live Preview

Agent Type	Name	Scope	Enabled	Pr...
Java	Java Auto Discovery R...	Default Scope	✓	0
.NET	.NET Auto Discovery R...	Default Scope	✓	0
Web Se...	Web Server Auto Disc...	Default Scope	✓	0
Node.js	Node.js Auto Discover...	Default Scope	✓	0
PHP	PHP Auto Discovery R...	Default Scope	✓	0
Python	Python Auto Discovery...	Default Scope	✓	0
Java	JavaTimer	Default Scope	✗	0
Java	Websphere web-servi...	Default Scope	✓	0
Java	Struts Action Servlet	Default Scope	✓	0

No Rule Selected

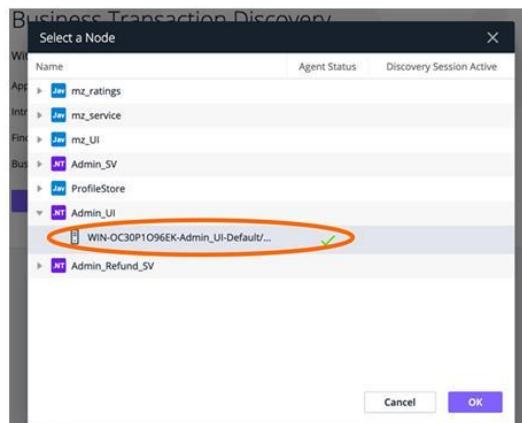
APPDYNAMICS

To start a new BT Discovery Session, click the Live Preview button in the Transaction Detection page.

BT Discovery Sessions

Launching a new Session

- Pick a Node for the Session
- The Node determines the scope of the Session



Business Transaction Discovery

Select a Node

Name	Agent Status	Discovery Session Active
mz_ratings		
mz_service		
mz_UI		
Admin_SV		
ProfileStore		
Admin_UI		
WIN-OC30P1O96EK-Admin_UI-Default/...		✓
Admin_Refund_SV		

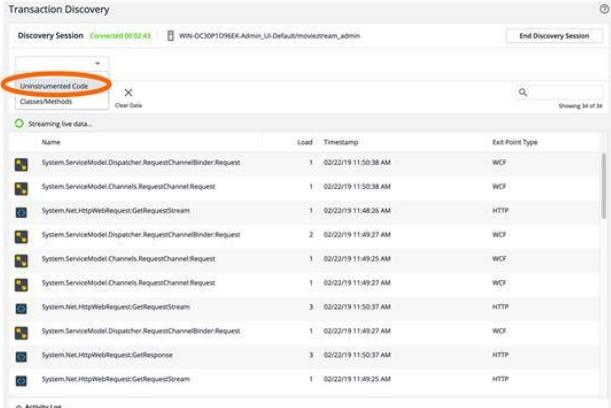
Cancel OK

APPDYNAMICS

BT Discovery Sessions

Un-instrumented Code Detection (Java and .NET)

- Critical Business Transactions, such as batch applications, may not be discovered
- Can help find the entry point of an undiscovered Business Transaction
- Finds exit calls with no corresponding entry point

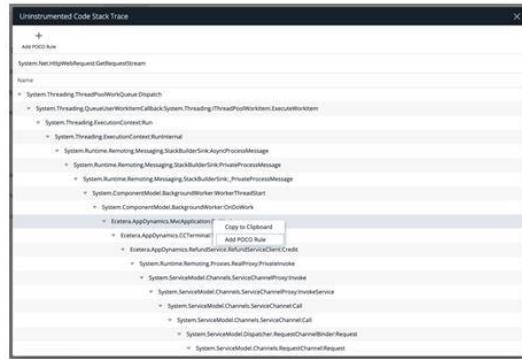


Name	Load	Timestamp	Exit Point Type
System.ServiceModel.Dispatcher.RequestChannelBinder.Request	1	02/22/19 11:50:38 AM	WCF
System.ServiceModel.Channels.RequestChannel.Request	1	02/22/19 11:50:38 AM	WCF
System.Net.HttpWebRequest.GetResponseStream	1	02/22/19 11:48:26 AM	HTTP
System.ServiceModel.Dispatcher.RequestChannelBinder.Request	2	02/22/19 11:49:27 AM	WCF
System.ServiceModel.Channels.RequestChannel.Request	1	02/22/19 11:49:25 AM	WCF
System.Net.HttpWebRequest.GetResponseStream	3	02/22/19 11:50:37 AM	HTTP
System.ServiceModel.Dispatcher.RequestChannelBinder.Request	1	02/22/19 11:49:27 AM	WCF
System.Net.HttpWebRequest.GetResponse	3	02/22/19 11:50:37 AM	HTTP
System.Net.HttpWebRequest.GetResponseStream	1	02/22/19 11:49:25 AM	HTTP

BT Discovery Sessions

Un-instrumented Code Detection (Java and .NET)

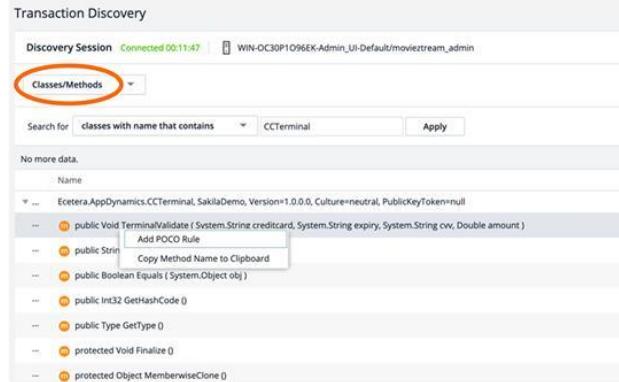
- Right-click on any line in list of Uninstrumented Code
- View the stack trace for any discovered calls
- Right-click to create a custom match rule



BT Discovery Sessions

Class / Method Browser (Java and .NET)

- Search for Classes and Methods to Instrument
- Various Matching options
- Directly add new Rule based on Class-Method
- Classes and Methods that are already instrumented are marked with a check



Transaction Discovery

Discovery Session Connected 00:11:47 WIN-DC30P1096EK-Admin_UI-Default/moviezteam_admin

Search for <classes with name that contains> CCTerminal Apply

No more data.

Name

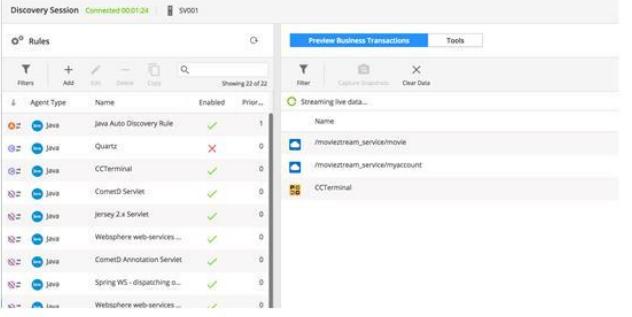
Ecetera.AppDynamics.CCTerminal, SakilaDemo, Version=1.0.0.0, Culture=neutral, PublicKeyToken=null

- public Void TerminalValidate (System.String creditcard, System.String expiry, System.String cvv, Double amount) Add POCO Rule
- public String Copy Method Name to Clipboard
- public Boolean Equals (System.Object obj)
- public Int32 GetHashCode ()
- public Type GetType ()
- protected Void Finalize ()
- protected Object MemberwiseClone ()

BT Discovery Sessions

Preview Business Transactions (Java Only)

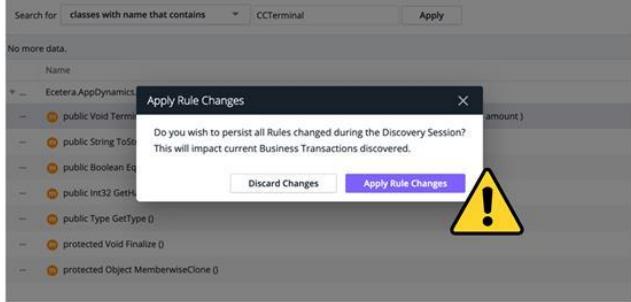
Functionally similar to Live Preview



The screenshot shows the AppDynamics interface with two main panels. The left panel is titled 'Discovery Session' and displays a table of 'Rules'. The table has columns for 'Agent Type', 'Name', 'Enabled', and 'Priority'. There are 22 rules listed, mostly Java-based, including 'Java Auto Discovery Rule', 'Quartz', 'CCTerminal', 'Cometd Servlet', 'Jersey 2.x Servlet', 'Websphere web-services...', 'Cometd Annotation Servlet', 'Spring WS - dispatching o...', and 'Websphere web-services...'. The right panel is titled 'Preview Business Transactions' and shows a list of transactions: '/moviestream_service/movie', '/moviestream_service/myaccount', and 'CCTerminal'. At the bottom left is the AppDynamics logo.

BT Discovery Sessions

Apply the changes made during the session, or discard



The screenshot shows a list of Java methods under the heading 'Ecetera.AppDynamics'. A modal dialog box titled 'Apply Rule Changes' is overlaid on the screen. The dialog contains the text: 'Do you wish to persist all Rules changed during the Discovery Session? This will impact current Business Transactions discovered.' It has two buttons: 'Discard Changes' and 'Apply Rule Changes', with the latter being highlighted. A yellow warning icon with an exclamation mark is positioned to the right of the dialog. The AppDynamics logo is visible at the bottom left of the interface.

When closing the BT Discovery Session, you will be prompted to either Discard Changes or Apply Rule Changes. Before applying the changes, remember at the beginning of the Session AppDynamics effectively copied all the current BT Rules into the Session. When you click Apply Rule Changes, everything done in the Session overwrites the outgoing BT Rule configuration. So take care in what you modify or delete in the Session, since it will be applied to the 'live' Rule configuration if you select to Apply Rule Changes.

Review Question - Answer

BT Discovery Sessions can save you time by:

- A) **Helping you see the Business Transactions detected by a rule so you can test that the rule is accurate**
- B) **Helping you identify undiscovered Business Transactions**
- C) Helping you track the total transaction values of credit card purchases as the purchases occur
- D) All of the above

Configuring Backend Detection

Backend Detection

Exit Point

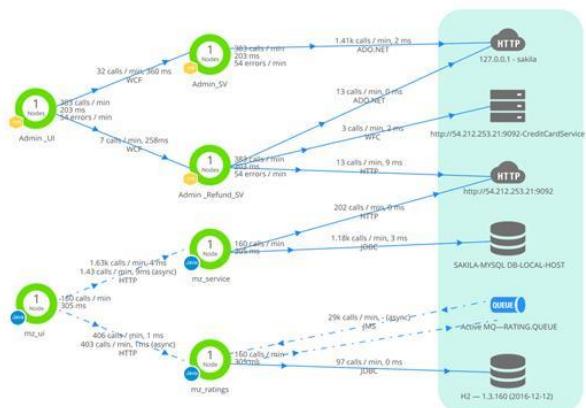
- A call from a node to another node or backend

Backends

- Uninstrumented databases and remote services

Calls to **backend** can be detected for:

- Database instances
 - Messaging services
 - Web services



AppDynamics gives you visibility into calls made to uninstrumented destinations inside or outside your application infrastructure. In AppDynamics, databases and remote services, such as web services message queues, are collectively known as backends.

AppDynamics discovers backends from exit point calls from the instrumented node to a backend. When the destination of an exit point call is not instrumented, the exit call results in a backend discovery event. By default AppDynamics automatically discovers a wide range of backends.

Notes From the Field

Missing backend databases

Business Problem

- New application was instrumented with AppDynamics for a customer.
- No databases showed up for the application after instrumentation.

Solution

- Noticed that the Neo4j databases did not show up.
- Created Custom Exit points to add in DBs.



APPDYNAMICS

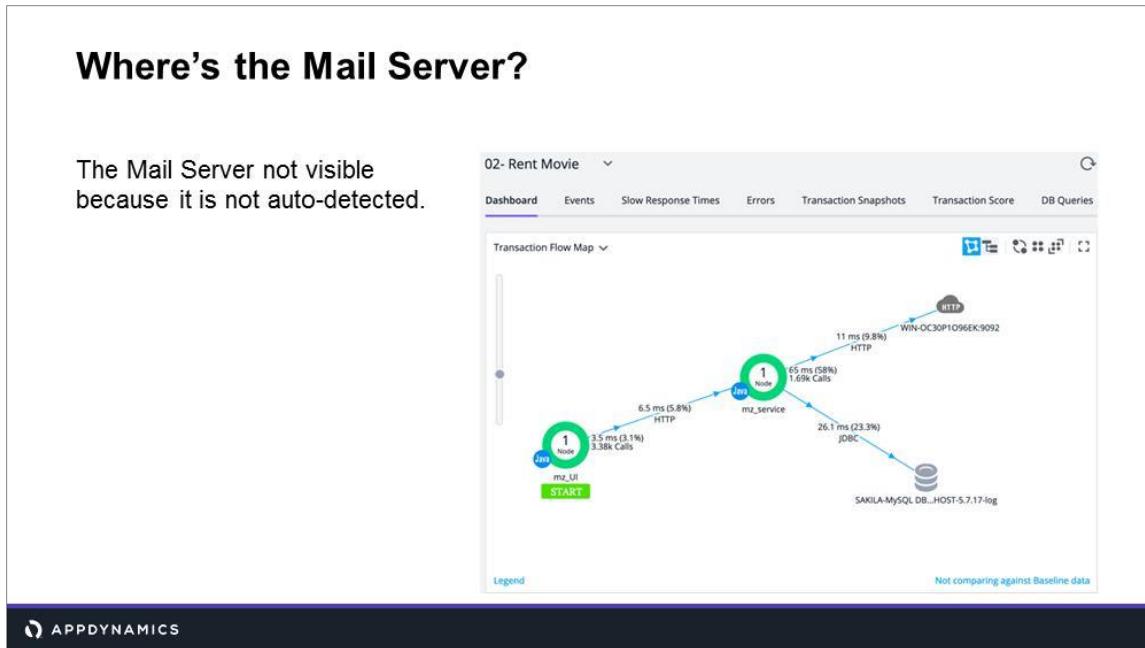
AppDynamics was installed at a customer site, monitoring an existing app.

A new component got built onto the app and then deployed to Prod with App Agents configured and running.

None of the DB calls were being detected, although after looking inside Callgraphs, some Connection calls were visible but without indication of AppD recognizing the DB call.

The new component made calls to a Neo4J DB, not something automatically detected by AppD.

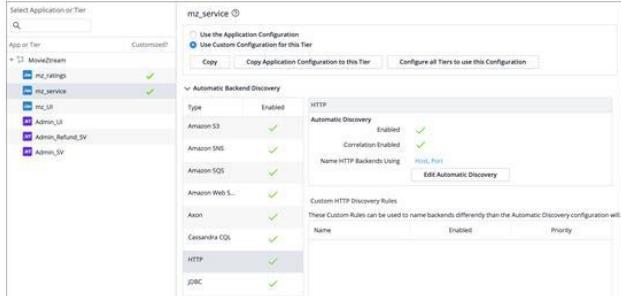
Once the Custom Backend Detection was set up for Neo4J calls, AppD began reporting the DB calls correctly.



In our Movieztream environment, there is in fact a call made from the mz_service Tier out to a Mail Gateway when movies are rented. This doesn't show up at present because it's not being auto-detected.

Backend Detection

- View (and adjust) existing Automatic Backend Discovery rules
- Auto Discovery Rules detect commonly used protocols
- Add new Custom Exit Points (bottom of page)



APPDYNAMICS

AppDynamics automatically discovers backends with common protocols and identifies them by type and related properties. You can view the auto-discovery rules in the Automatic Backend Discovery list in the Backend Detection tab.

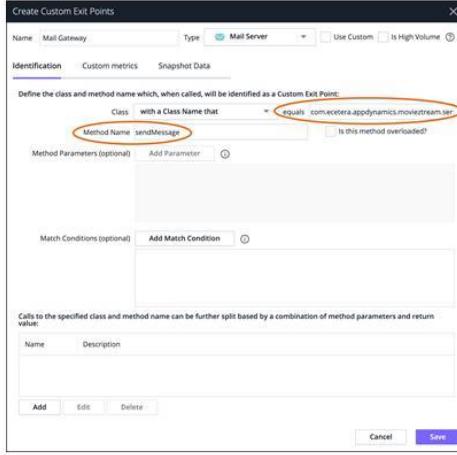
For the types of backends that aren't discovered automatically, you can set up backend detection. For example, you can define a custom exit call to monitor the file system read method. Once configured, these custom exit points from the downstream Tier appear on the flow maps, and the calls will be added to the metric browser. This means you can define health rules to trigger an event based on external resource behavior, and exercise proactive monitoring for these resources.

Backend Detection

How it is instrumented

Set up backend detection to capture unknown protocols

- File system
- Mainframe
- Call to unmonitored or undiscovered tier



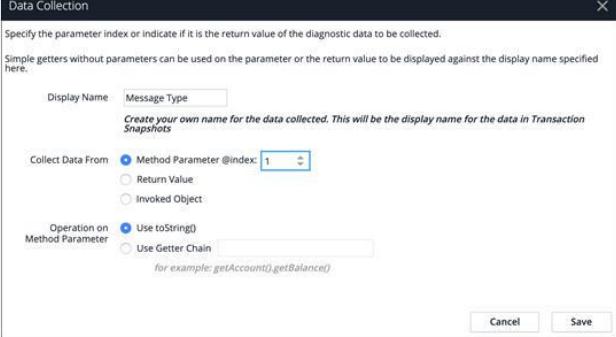
APPDYNAMICS

Actual configuration of the Custom Backend relies simply on configuring the Method on the Class in your application code where the call is made to the Backend system. It therefore does not involve capturing complex details about the protocol used to connect to the remote service, or specifying how your app talks to the remote service: the approach is far simpler.

Backend Detection

How it is instrumented - Split Options

- Optionally split based on Method Parameters, Return Value, or Invoked Object
- Get a unique Backend for every unique value



APPDYNAMICS

If your Backend functionality is actually more than one system, and you can distinguish between the different Backends based on some element of the selected Method parameters or return object, then that can be selected to detect distinct Backends.

Notes From the Field

Database Discovery Event

Business Problem

- A developer had hard-coded a call to a staging DB in a production environment and the new code had been running overnight.
- In addition to the cost, this mistake triggered a massive security audit, as the staging environment shouldn't have been able to connect to prod.

Solution

- AppDynamics was configured to notify appropriate contacts whenever a new backend was discovered.



© APPDYNAMICS

Review Question - Answer

You can set up backend detection to capture calls to file systems, mainframes, and other calls to unmonitored or undiscovered system components.

True

False

What You Learned

- How to employ multiple strategies for managing Business Transactions
- How to determine when to use Service Endpoints instead of Business Transactions
- How to replace BTs with Service Endpoints
- How to split Business Transactions
- How to use Live Preview while configuring Custom Match Rules
- How to test BT rules in a Discovery Session
- How to search for running Classes and Methods, and Uninstrumented Code
- How to use Automatic and Custom backend detection

Differentiate yourself and your company with AppDynamics Certification

APPDYNAMICS | Certification Program

General Certification Information

<https://learn.appdynamics.com/certifications>

AppDynamics Certified Associate Performance Analyst

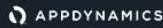
<https://learn.appdynamics.com/certifications/performance-analyst>

AppDynamics Certified Associate Administrator

<https://learn.appdynamics.com/certifications/administrator>

AppDynamics Certified Implementation Professional

<https://learn.appdynamics.com/certifications/implmenter>



To differentiate yourself and your company in an increasingly competitive market, please consider becoming AppDynamics certified.

For more information, please copy and paste this URL into a separate tab in your browser: <https://learn.appdynamics.com/certifications>

For information on specific certification tracks, please copy and paste the appropriate URLs below:

AppDynamics Certified Associate Performance Analyst:

<https://learn.appdynamics.com/certifications/performance-analyst>

AppDynamics Certified Associate Administrator:

<https://learn.appdynamics.com/certifications/administrator>

AppDynamics Certified Implementation Professional:

<https://learn.appdynamics.com/certifications/implmenter>

Proactive Monitoring and Dashboards (APM223)



The logo for AppDynamics University is displayed. It features a white square containing the AppDynamics logo (a stylized 'A' icon) and the word 'APPDYNAMICS' in a sans-serif font. Below this, a vertical line separates the logo from the word 'University' in a larger, bold, sans-serif font.

APM223 - Proactive Monitoring and Dashboards

Core APM II: Advanced - Module 3

Objectives

Course

After completing this course, you will be able to:

- Explain the difference between Errors and Exceptions in AppDynamics
- Configure AppDynamics to ignore specific Exceptions
- Use baselines to evaluate application health over time
- Configure Health Rules
- Configure Policies and Actions to respond to Health Rule Violations
- Create dashboards to share data
- Create scheduled reports

Labs

In this course's labs, you will:

- Customize Error Detection
- Configure Baselines to Help Identify Performance Problems
- Configure Health Rules to monitor Application Performance
- Set up Runbook Automation
- Update a Custom Dashboard

Customizing Error Detection

Errors and Exceptions

Fundamental Differences

Errors and Exceptions are treated differently in AppDynamics:

- Errors indicate an Error Snapshot is taken
- Exceptions may be recorded by AppDynamics but may not result in an Error
- Exceptions can be problems outside the scope of a Business Transaction

Transaction Scorecard			
Category	Count	Percentage	Total
Normal	33.4k	97.9 %	
Slow	94	0.3 %	
Very Slow	472	1.4 %	
Stall	4	0.0 %	
Errors	146	0.4 %	
Exceptions			

Not comparing against Baseline data Exceptions 1,117 total 19 / min

169 ms average Errors 0.4% 146 errors 2 errors / min

AM 11:30 AM

10:45 AM 11:00 AM 11:15 AM 11:30 AM

APPDYNAMICS

An Error transaction can be one of the cases identified above. They will be listed in the Exceptions summary section of the application dashboard, and on the Errors page accessible from the left navigation pane under **Troubleshoot > Errors**.

If a transaction experiences an Error, it is counted as an Error transaction and not as a slow, very slow, or stalled transaction - even if the transaction was also slow or stalled.

An Exception may be where an exception has occurred in the code outside the scope of a BT.

There is not a one-to-one correspondence between the number of Errors and the number of Exceptions. For example, a Business Transaction may experience a single code 500 Error in which several Exceptions were logged as the transaction passed through multiple tiers.

When configuring Error Detection, you decide the circumstances in which the Agent creates an Error Event and sends it to the Controller. An Event is part of the packaging that the Agent sends to the Controller and may or may not affect the user experience of the Business Transaction. If there are no Error Events related to the starting tier of the BT (even if there may be Error Events related to other tiers in the BT), the BT user experience will not be "Error" regardless of the response time.

Notes From the Field

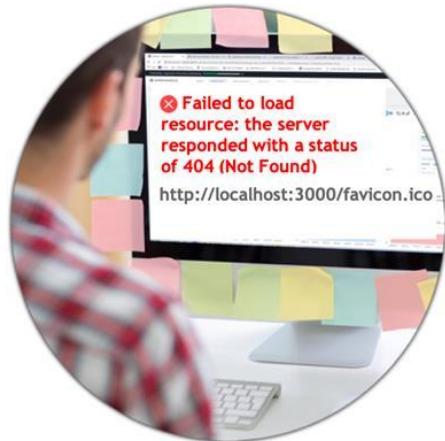
Missing Favicon Error

Business Problem

- Forgot a seemingly trivial image file during deployment
- Controller captured Error snapshots for EVERY request!
- Inflexible 2-week release window, so could not add image back in

Solution

- Added in a new Ignore Exception filter to temporarily mask these Error snapshots

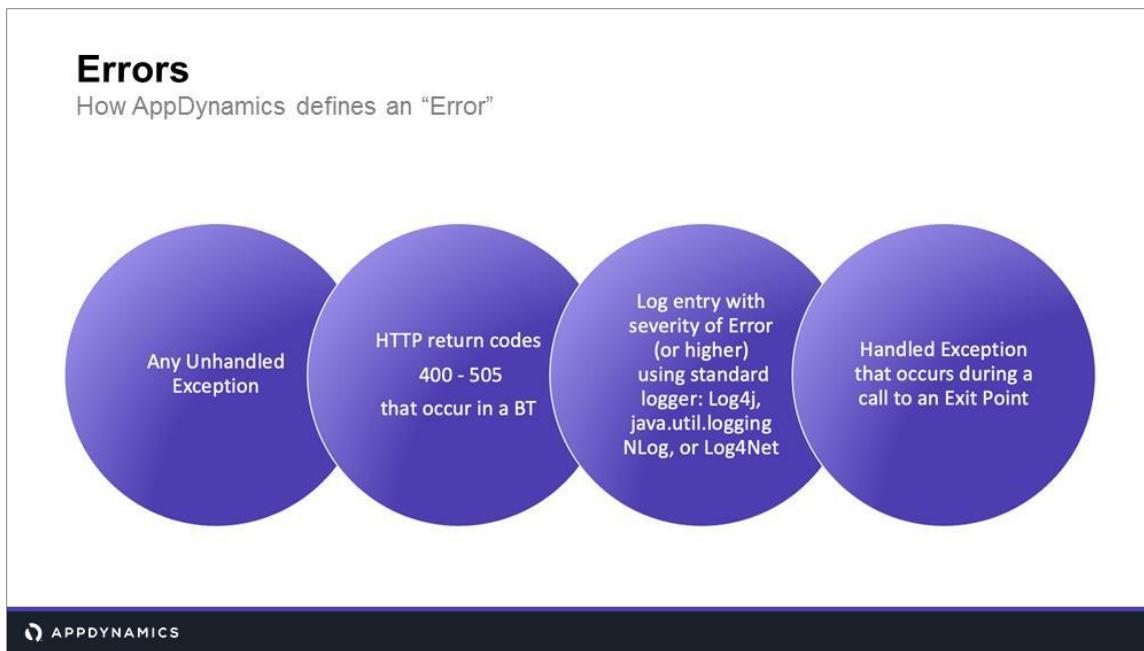


One client we worked with had a well-defined and inflexible release schedule.

In one release they mistakenly deleted the Favicon, and so every time a page loaded, it threw an error.

The next release wasn't for 2 weeks, and they were not allowed to go in and make a tweak.

So they configured an exception to ignore when evaluating errors.



AppDynamics reports errors that occur during the execution of Business Transactions and Error messages outside the context of a Business Transaction that are logged by the application server (called application server exceptions).

Typical examples of errors are: Logged Exceptions or Messages, Errors based on HTTP Return Codes, Errors based on Redirect Pages.

Conditions that result in error detection include Unhandled Exceptions and HTTP error codes.

An unhandled exception that occurs during the execution of a business transaction is reported as an error.

Unhandled exceptions that occur during an exit call, for example, calls to databases, web services, or message queue servers.

Errors

Troubleshoot > Errors page > Error Transactions

Focus on Business Transaction Errors and observe any patterns for the selected time range

The screenshot shows the AppDynamics Errors page. The top navigation bar has tabs for 'Errors', 'Error Transactions' (which is highlighted with a red oval), and 'Exceptions'. Below the tabs is a line chart titled 'Error Transactions' showing the count of errors over time from 3:15 PM to 4:00 PM. The chart includes a legend for response time: Very Slow (light blue), Slow (yellow), Stale (grey), Error (red), and Normal (green). A color bar on the right indicates the percentage of errors: Normal (98.5%), Slow (0.3%), Stale (0.4%), Very Slow (0.8%), and Error (35.0%). Below the chart is a table titled 'Error Transaction Snapshots' with columns: Time, Eve Time (ms), URL, Business Transaction, Tier, and Node. The table lists 10 entries of error transactions, all from the 'RentalPayments' business transaction tier, occurring between 02/22/19 4:03:33 PM and 02/22/19 4:03:27 PM. The Node column shows 'WIN-0C3P109L...' for all entries.

Time	Eve Time (ms)	URL	Business Transaction	Tier	Node
02/22/19 4:03:33 PM	25	/moviestream_admin/Rental/Pay...	RentalPayments	Admin_UI	WIN-0C3P109L...
02/22/19 4:04:50 PM	8	/moviestream_admin/Rental/Pay...	RentalPayments	Admin_UI	WIN-0C3P109L...
02/22/19 4:04:41 PM	346	/moviestream_admin/Customer...	moviestream_admin/Cust...	Admin_UI	WIN-0C3P109L...
02/22/19 4:04:30 PM	7	/moviestream_admin/Rental/Pay...	RentalPayments	Admin_UI	WIN-0C3P109L...
02/22/19 4:03:27 PM	694	/moviestream_admin/Customer...	moviestream_admin/Cust...	Admin_UI	WIN-0C3P109L...
02/22/19 4:02:43 PM	10	/moviestream_admin/Rental/Pay...	RentalPayments	Admin_UI	WIN-0C3P109L...
02/22/19 4:02:37 PM	8	/moviestream_admin/Rental/Pay...	RentalPayments	Admin_UI	WIN-0C3P109L...
02/22/19 4:02:08 PM	8	/moviestream_admin/Rental/Pay...	RentalPayments	Admin_UI	WIN-0C3P109L...
02/22/19 4:02:01 PM	10	/moviestream_admin/Rental/Pay...	RentalPayments	Admin_UI	WIN-0C3P109L...

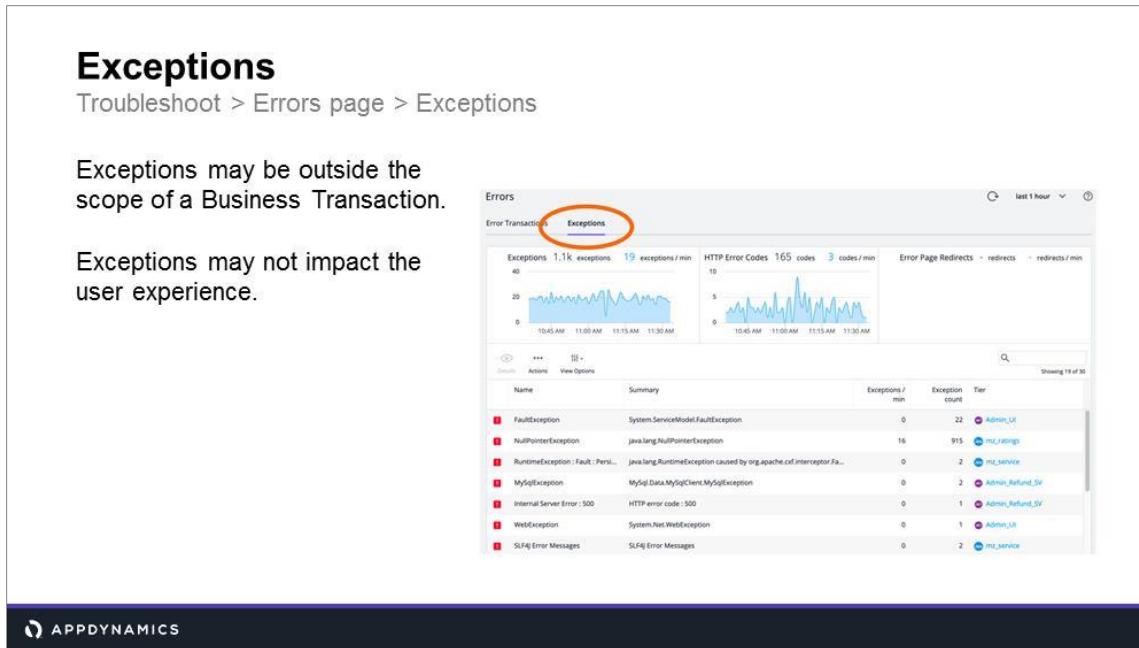
APPDYNAMICS

Exceptions

Troubleshoot > Errors page > Exceptions

Exceptions may be outside the scope of a Business Transaction.

Exceptions may not impact the user experience.

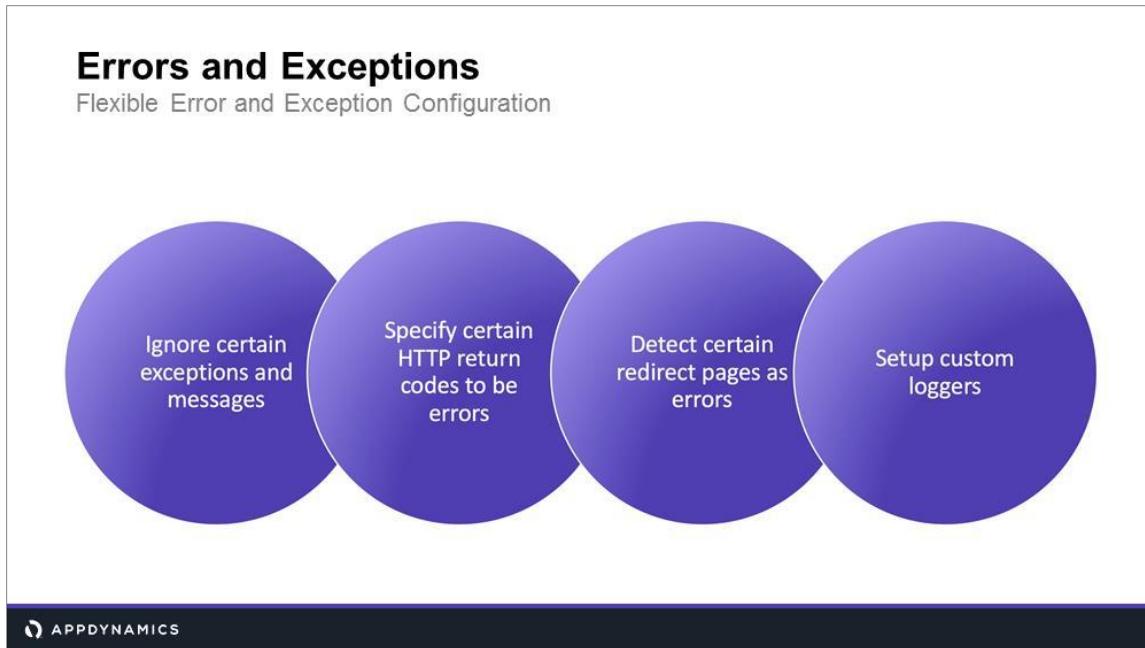


Name	Summary	Exceptions / min	Exception Count	Tier
FaultException	System.ServiceModel.FaultException	0	22	Admin_UI
NullPointerException	java.lang.NullPointerException	16	915	mt_rending
Runtimeexception	java.lang.RuntimeException caused by org.apache.cxf.interceptor.Fa...	0	2	mt_service
MySQLException	MySQL Data.MySQLException	0	2	Admin_Refund_SV
Internal Server Error	HTTP error code : 500	0	1	Admin_Refund_SV
WebException	System.Net.WebException	0	1	Admin_UI
SL4J Error Messages	SL4J Error Messages	0	2	mt_service

When you view the Exceptions tab, you are no longer looking at Error Transactions. Instead, you are looking at the underlying events that are detected by the Agent.

If a stack trace for the Exception is available, you can access it from the Exception tab in the Controller UI. A stack trace is available for a given Exception if the Exception was passed in the log call. For example a logging call in the form of logger.log (Level.ERROR, String msg, Throwable) would include a stack trace, whereas a call in the form of logger.log (Level.ERROR, String msg) would not.

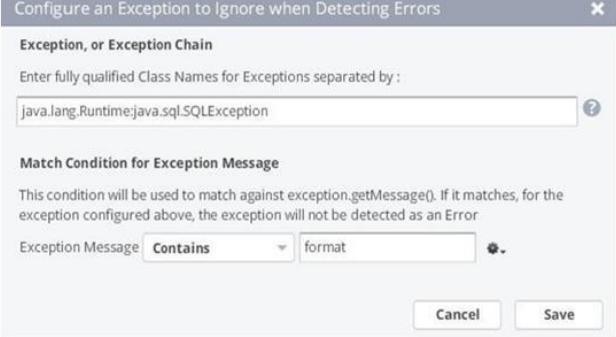
When an Exception occurs in the context of a Business Transaction, it results in an "Error" if it occurs in the beginning tier of the transaction.



AppDynamics error and exception detection settings are flexible, and the configuration changes you make will be applied to all tiers in the application. Access the error configuration screen by navigating to **Configuration > Instrumentation > Error Detection**.

Configuring AppDynamics to Ignore Certain Errors

You can configure AppDynamics to ignore exceptions and log messages as error indicators by adding an exception to ignore.



The dialog box is titled 'Configure an Exception to Ignore when Detecting Errors'. It has a section for 'Exception, or Exception Chain' with a text input field containing 'java.lang.RuntimeException:java.sql.SQLException'. Below that is a section for 'Match Condition for Exception Message' with a dropdown menu set to 'Contains' and a value 'format'. At the bottom are 'Cancel' and 'Save' buttons.

Certain types of exceptions, loggers or log messages as transaction error indicators may not reflect events that should be counted as transaction errors in your environment. They may include exceptions raised by application framework code or by user login failures.

When you configure an exception to be ignored, the agent detects the exception, increments the exception count, and displays the exception in Exceptions lists in the UI, but the Business Transaction in which the error is thrown is not considered an "error transaction" for monitoring purposes. The transaction snapshot would not show the exception in the Summary or Error Details section and the user experience for the transaction instance would be unaffected by the exception.

To configure exceptions for the agent to ignore, click the Add New Exception to Ignore button in the error configuration window.

Enter the class name of the exception to be ignored and the match condition for the exception message. For the match condition, if you do not need to filter for specific messages in the exception, select "Is Not Empty". If you want to filter for Null, select "Is Not Empty" and use the gear icon to select NOT, which, in effect, tests for "is empty".

Notes From the Field

Error Detection Configuration

Business Problem

- Client had a legacy system that made an API call to a 3rd party vendor to make a payment
- Each API response was an exception, and AppDynamics originally tracked each instance of the transaction as 'failed'
- In reality, only some of the exceptions were actually errors

Solution

- Using Error Detection instrumentation, we were able to filter out the false errors



Review Question - Answer

Which of the following are errors in AppDynamics?

- A) An unhandled exception
- B) HTTP error codes from 400 to 505
- C) An exception that occurs during an exit call to a backend
- D) An exception that is handled but logged with a severity of error or higher

Review Question - Answer

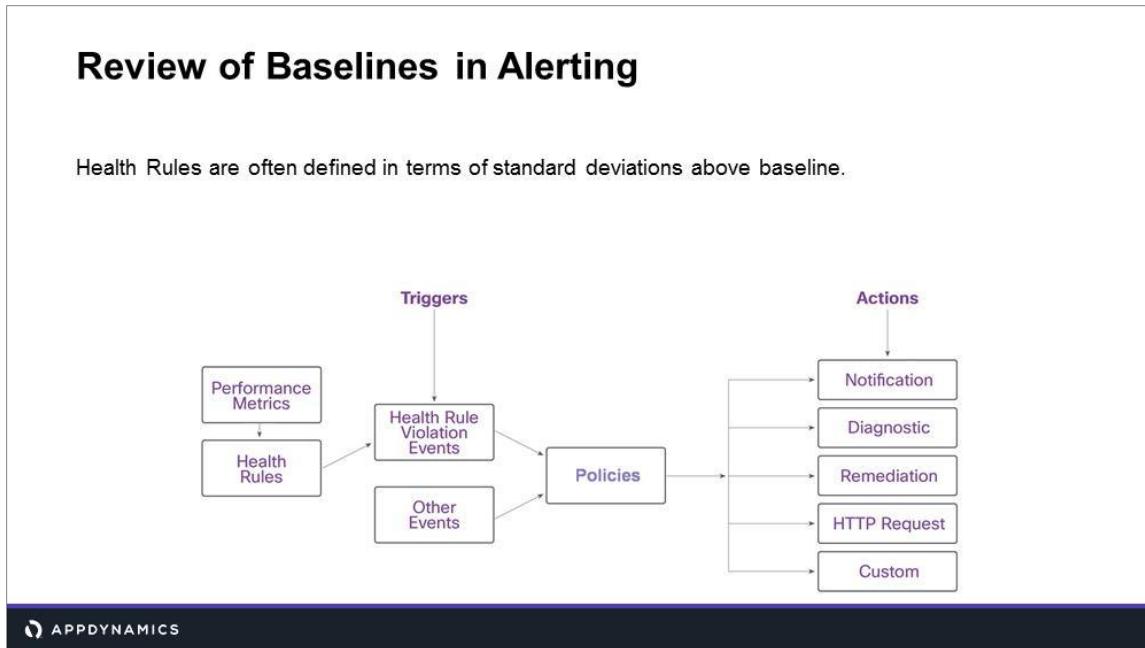
What are some options you can configure for error detection?

- A) Specify certain HTTP return codes to be errors
- B) Ignore certain exceptions and messages
- C) Detect certain redirect pages as errors
- D) Set up custom loggers

Configuring Baselines

Review of Baselines

- The average value of a metric, calculated using its values across a defined time-range, according to a configurable pattern
 - Dynamic or fixed time range
 - None, Daily, Weekly or Monthly Trend
- Used to judge the 'normal' performance of a metric



AppDynamics permits the configuration of Health Rule logic relative to either a given Baseline or a specific value. Once the Health Rule is defined, it can be configured to trigger one or more Actions via one or more Policies.

Baselines and Deviations

Common use

- Baseline evaluate averages and standard deviations for metric data at any given point in time.
- Can be used in health rules to trigger events when metric values deviate from the baseline.
- Each application has one Baseline set as default, this can be configured per application.
- The Controller constantly collects data for the default Baseline and caches the results.

EXAMPLE

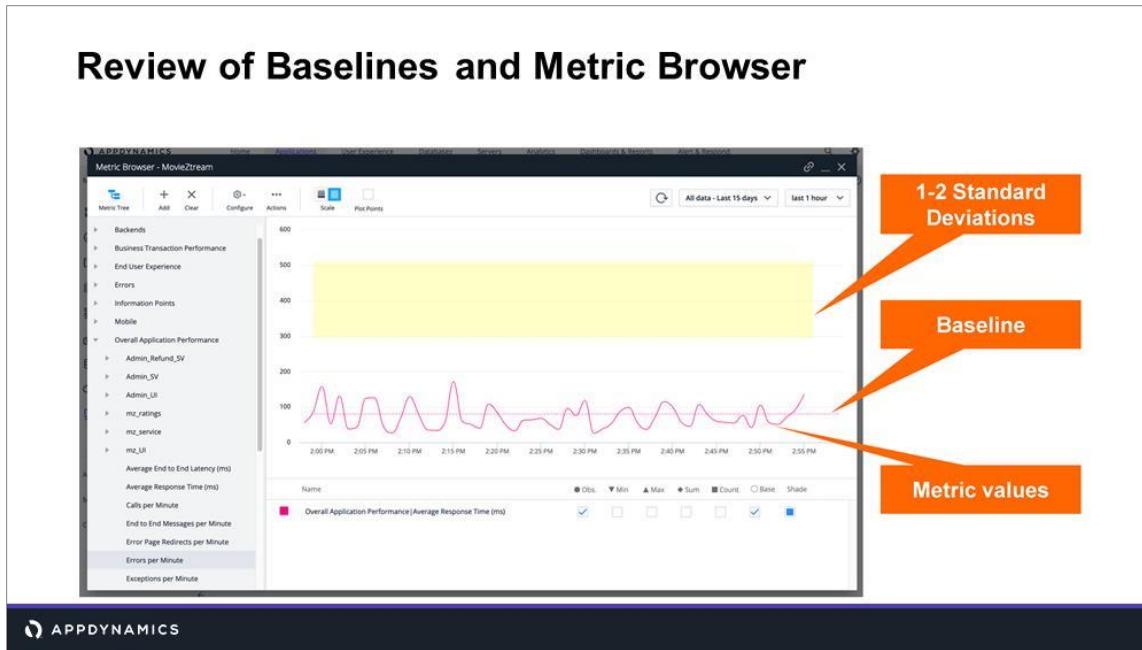
• “Warning” level performance issue is two baseline standard deviations for a period greater than 15 minutes

• “Critical” level performance issue is three baseline standard deviations for a period greater than 15 minutes



Baselines are calculated by a number of standard deviations above the average by default.

Within a health rule configuration you can choose how to define criteria based on baseline deviations as a number of standard deviations, percentages, or as fixed values.



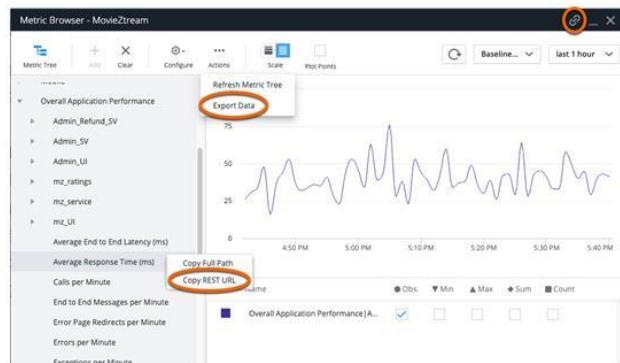
The wonderful thing about the Metric Browser is the ability to see not just the value of the baseline compared to the metrics' values, but also the range of values of 1 - 2, 2 - 3, 3 - 4, 4 - 5 deviations above baseline.

This helps you really understand how far from normal a metric is straying, and judge whether or not the performance of something is acceptable or clearly struggling.

Getting Data out of the Metric Browser

Three options:

1. Select Actions > Export Data
2. Right-click the metric and select **Copy REST URL**
3. Click the **Link** button to copy the link and send it to someone



Notes From the Field

Intermittent overnight issue

Business Problem

- Random inconsistent system slowdown occurring between 2-5AM (but not every night).

Solution

- Use baselines and standard deviation to define a Health Rule and then run a suitable Diagnostic Session Action.



© APPDYNAMICS

A customer was finding that they occasionally had a slowdown in the system, only ever between 2 and 5 am, and not all the time. The issue would last for about 10 minutes, then normal service apparently resumed.

What did we recommend they do? Set up a Health Rule, based on the standard deviation from baseline for the ART of the relevant Business Transaction, and then trigger a Diagnostic Session to capture Full Snapshots.

Without AppDynamics baselines and standard deviations architecture, it would have been tricky knowing how to define the right Health Rule.

Viewing Current Baselines

- Trends
 - Monthly, Weekly, Daily, None
- Time Periods
 - Dynamic – rolling time window
 - Fixed – use a specific time range

APPDYNAMICS

When the Controller retrieves baseline data from the Controller's database, it can incorporate the concept of a Trend, which adds a "where" clause to the query such that the data retrieved for the Baseline might "slice" (using a "where" clause) on [hour], [hour + day-of-week], or [hour + day-of-month]. The context comes from the timing with which the recently observed data is being compared to Baseline data. For example, if Trend is set to weekly, and the recently collected data is collected at 9am on a Tuesday, then the query to find the Baseline values will only retrieve data that was also collected at 9am on a Tuesday across the entire time range of the Baseline (perhaps the last 90 days). Depending on the Trend selection in the Baseline, the query to retrieve the Baseline values works according to the following:

None: Calculated as the unweighted average across the entire time range of the Baseline.

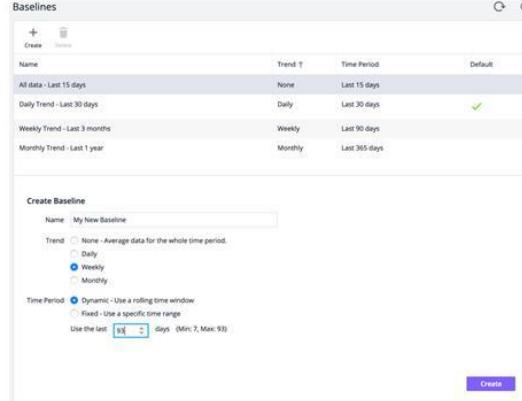
Daily: Calculated as the average of the same hour of each day for up to 31 previous days.

Weekly: Calculated as the average of the same hour for the same day of the week for up to 3 previous months.

Monthly: Calculated as the average of the same hour for the same day of the month for up to 12 previous months.

Creating a Baseline

- Select Trend and Time Period
- Optionally specify new baseline to become the Default baseline (or leave Default setting unchanged)
- Immediately available after creation



Baselines

Name	Trend	Time Period	Default
All data - Last 15 days	None	Last 15 days	
Daily Trend - Last 30 days	Daily	Last 30 days	✓
Weekly Trend - Last 3 months	Weekly	Last 90 days	
Monthly Trend - Last 1 year	Monthly	Last 365 days	

Create Baseline

Name: My New Baseline

Trend: None - Average data for the whole time period. Weekly Daily Monthly

Time Period: Dynamic - Use a rolling time window Fixed - Use a specific time range
Use the last days (Min: 7, Max: 99)

Create

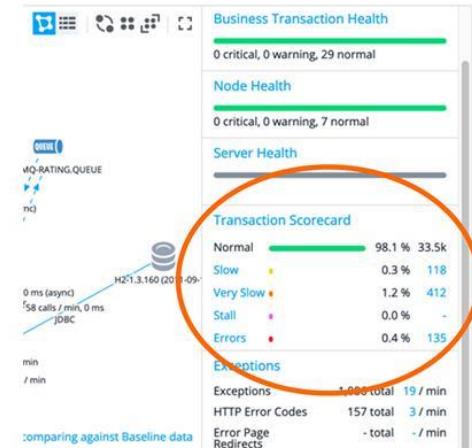
Creating a new Baseline is simple and it becomes available as soon as you have saved it. It is not like collecting metrics, where there is a change to the agent configuration that causes new data to be captured.

Baselines are calculated on the fly as required, based on the historical metric data values.

Business Transaction Thresholds

Q: How does AppDynamics judge Slow, Very Slow, and Stall transactions?

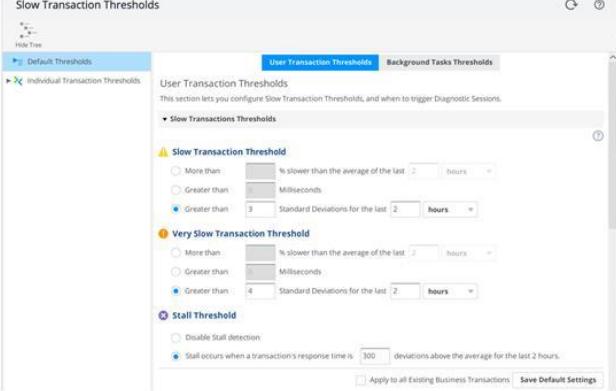
A: Every BT request is compared against configurable Threshold values.



All requests are scored against Slow and Very Slow thresholds, even if there is no Transaction Snapshot.

Configuring Business Transaction Thresholds

- Modify default definitions, or definitions for individual transactions, for Slow, Very Slow, and Stall Thresholds
- Define in terms of:
 - Standard Deviations
 - Fixed Time
 - % Slower
- Stall only defined in terms of deviations above average



Threshold definitions can be viewed under **Configuration > Slow Transaction Thresholds**. If the default settings don't meet your need, you can custom define them to tailor to your environment.

When you define a Threshold in terms of a number of Deviations, you don't need to specify a particular Baseline, because AppDynamics uses a standard average calculation for the selected time window, as the basis for determining the deviation value.

Review Question - Answer

You can change slow, very slow, and stall thresholds for the entire application, or for an individual Business Transaction.

True

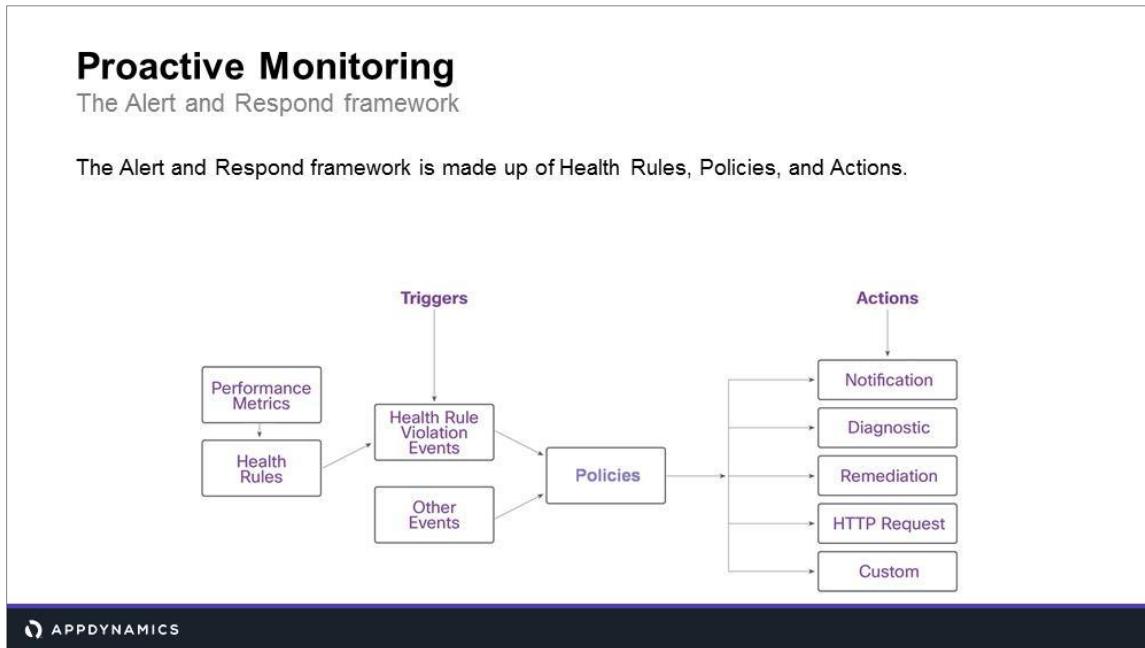
False

Review Question - Answer

Which of the following are true of Baselines?

- A) They are stored by default for one year
- B) Their time periods are configurable**
- C) They can be used to determine the general health of Business Transactions, applications, Tiers, and Nodes
- D) They show historic performance patterns and averages

Configuring Health Rules



Health Rules (or Events) monitor performance and - when violated - provide messages in Controller

Actions define what can be done in a particular situation

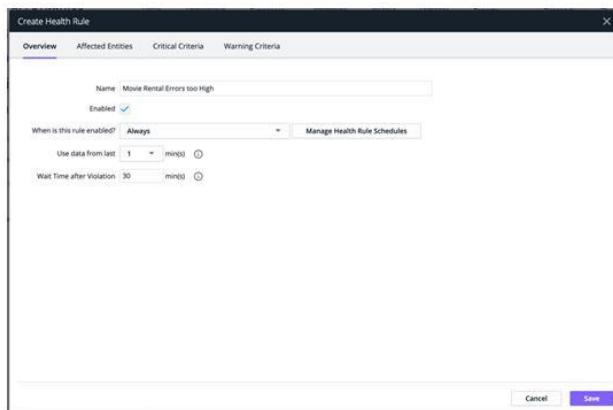
Policies link Health Rules (or Events) to Actions

Now we'll start talking about the proactive alerting mechanism. Three pieces work together to enable this feature, so before we discuss each piece let's take a quick look at the whole to understand the relationship.

We have already discussed 'Events.' Remember, any change in application state that is of potential interest is logged as an Event. You can set up a Policy to respond to these Events in a specific way, so you can take care of potential performance issues proactively. For this to work, you have to have a Policy configured already, and it has to be active when an applicable event occurs. Let's take a closer look at each component.

The Overview Tab

- Select a suitable Health Rule name
- Will be default Enabled
- Go with Always enabled, unless need a specific Schedule
- “Use data from last” lets you control time interval over which metrics are evaluated
- Avoid violation message explosion with “Wait Time after Violation”



© APPDYNAMICS

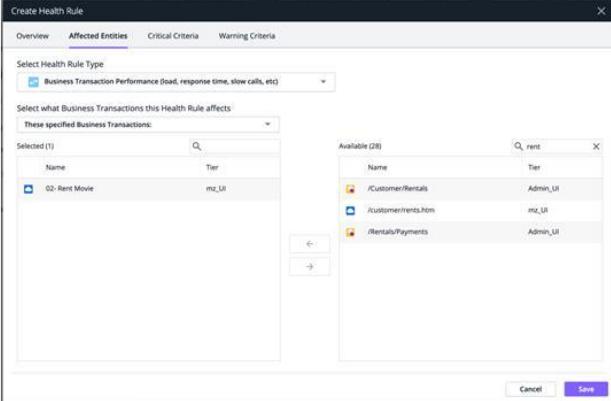
A Health Rule defines a condition or set of conditions in terms of metrics. The condition compares the performance metrics that AppDynamics collects with some static or dynamic threshold that you define. If performance exceeds the threshold, a Health Rule violation Event is triggered. AppDynamics comes pre-loaded with a set of default Health Rules. Those are accessible in **Alert and Respond > Health Rules**, and can be modified as you wish. When the conditions set in those rules are violated, the Health Rule Violation Alert appears on the Events section of the Application Dashboard.

You can also create your own Health Rules, which are highly configurable, so you can set thresholds based on dynamic or static parameters, and assign critical or warning conditions so you can set different actions for each. You can also configure a blanket rule that applies to the whole application, or create ones that only apply to specific entities in your application. Also, you have the ability to choose the times that a health rule is enabled, the evaluation time period, and how much recent data to use to evaluate a Health Rule.

The Affected Entities Tab

What does this Health Rule apply to?

- Overall Application
- Business Transactions
- Tier/Node Health
- Servers
- Database & Remote Services
- Error Rates
- Service Endpoints
- Information Points
- Custom



APPDYNAMICS

Type: The type of metric that the health rule will be monitoring. This can be:

- **Transaction Performance:** Application or Business Transaction performance metrics
- **Node Health:** Metrics related to either Business Transaction performance on a node level, or Node infrastructure metrics like JMX, Hardware Metrics, etc.
- **End User Experience:** Web and Mobile RUM metrics
- **Mobile APM:** iOS or Android mobile app metrics
- **Databases and Remote Services**
- **Error Rates:** Exceptions, HTTP Error codes, etc.
- **Information Points:** Metrics generated by the Information Point feature

Critical Criteria and Warning Criteria Tabs

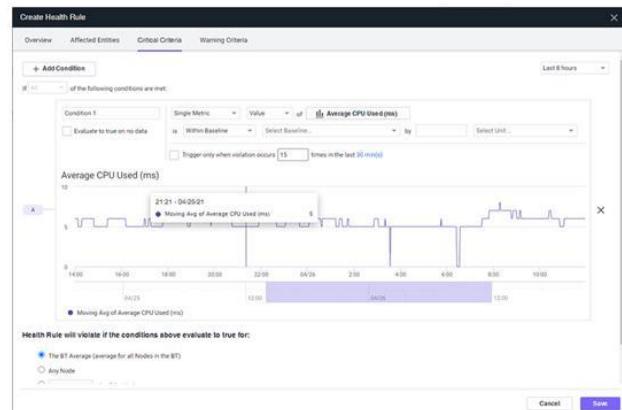
- Thresholds based on one or more condition(s) can fire:
 - If ANY condition applies
 - If ALL conditions apply
 - If CUSTOM conditions apply
 - Boolean expression:
A OR (B AND C)
- Threshold values can be based on:
 - Baselines over a time interval
 - Literal values

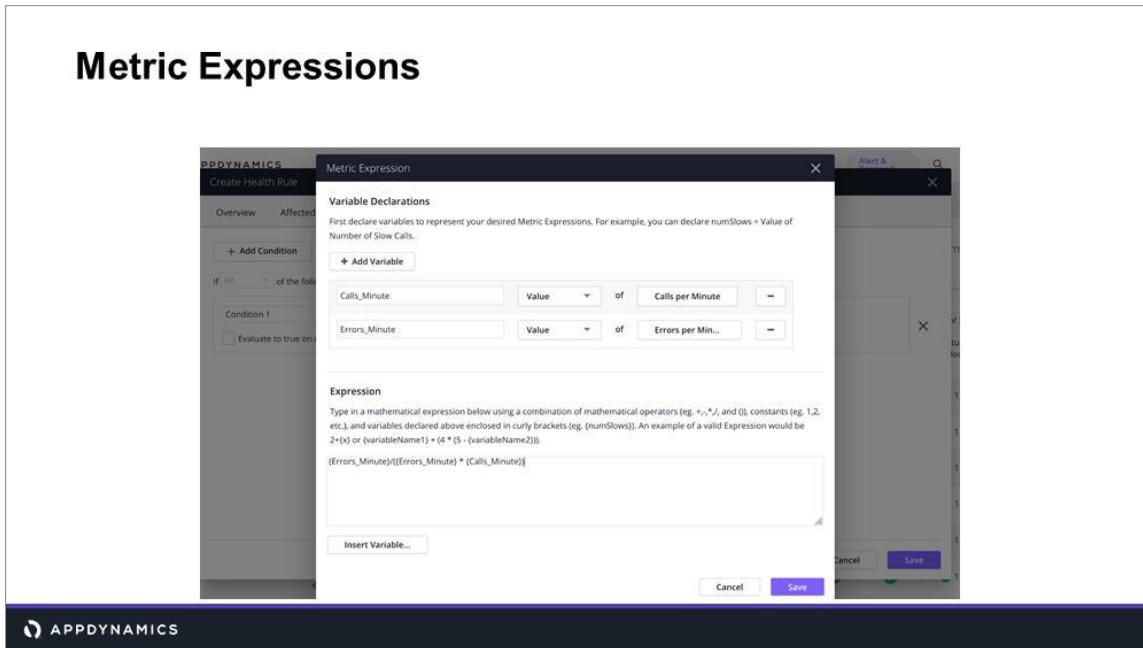
The screenshot shows the 'Create Health Rule' dialog box with the 'Critical Criteria' tab selected. The 'Affected Entities' tab is also visible. The 'Critical Criteria' section contains three conditions (A, B, and C) defined by 'If All' or 'Any' of the following conditions are met. Each condition includes a metric (Single Metric), a comparison operator (Value or > Specific Value), and a threshold value. There are also checkboxes for 'Evaluate to true on no data' and 'Trigger only when violation occurs'.

After the **Affected Entities** tab, you have two almost identical steps to configure the **Critical** and **Warning** criteria.

Alert Sensitivity Tuning (AST)

- Only available on Health Rules that monitor:
 - a business transaction
 - a service endpoint
 - a remote service
- The graphical representation indicates the baseline and the baseline standard deviation
- SaaS only feature



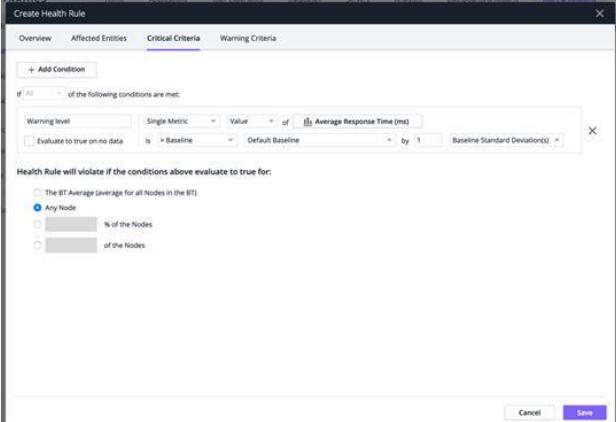


It is also possible to create more complex Critical and Warning rule logic conditions with the use of Variables (based ultimately on available metrics) which you can define in the Metric Expression dialog.

Discussion: Health Rules & Baselines

When do you wake people up if Login is slow?

Is a baseline always the best way to judge if it's slow?



APPDYNAMICS

In a lot of cases, defining the logic of your Health Rule Critical and Warning conditions using comparison of the metric value against Baseline is a good approach, as you gain from the dynamic aspect of Baselines to shift gradually over time.

However, in some situations, you will want to set absolute values in your Health Rule logic.

Imagine a situation where user login usually takes 20ms: do you want a Health Rule to violate if login degrades during the night to 50ms? If that triggers an Action that wakes you (or worse, the Head of Ops) up, there is an argument to be made that the Health Rule is wrong.

In this kind of situation, it can make more sense to use absolute values for your Health Rule condition logic, perhaps reflecting Service Levels that you have already built up, so that the Health Rule only violates when it crosses some hard boundary that you need to be aware of.

Health Rules: Recommended Strategy

Target the Yellow

Don't ignore Warning violation messages!

Assess yellow to stay away from red!

Type	Summary	Time	Business Transaction	Tier	Node	Actions
App Server Restart	Application Server JVM was re-started N...	02/26/19 2:04:35 AM	-	mz_UI	UI001	-
App Server Restart	Application Server JVM was re-started N...	02/26/19 2:05:51 AM	-	mz_service	SV003	-
App Server Restart	Application Server JVM was re-started N...	02/26/19 2:03:28 AM	-	mz_ratings	RT001	-
Health Rule Violation	Health Rule Business Transaction respo...	02/25/19 5:10:55 AM	/movies/re...	mz_UI	-	-
App Server Restart	Application Server JVM was re-started N...	02/25/19 2:04:23 AM	-	mz_UI	UI001	-
App Server Restart	Application Server JVM was re-started N...	02/25/19 2:03:44 AM	-	mz_service	SV003	-
App Server Restart	Application Server JVM was re-started N...	02/25/19 2:03:15 AM	-	mz_ratings	RT001	-
Health Rule Violation	Health Rule Business Transaction respo...	02/24/19 5:09:57 PM	/movies/re...	mz_UI	-	-
Health Rule Violation	Health Rule Business Transaction respo...	02/24/19 8:09:57 AM	/movies/re...	mz_UI	-	-
App Server Restart	Application Server JVM was re-started N...	02/24/19 2:04:34 AM	-	mz_UI	UI001	-
App Server Restart	Application Server JVM was re-started N...	02/24/19 2:03:46 AM	-	mz_service	SV003	-
App Server Restart	Application Server JVM was re-started N...	02/24/19 2:03:25 AM	-	mz_ratings	RT001	-

 APPDYNAMICS

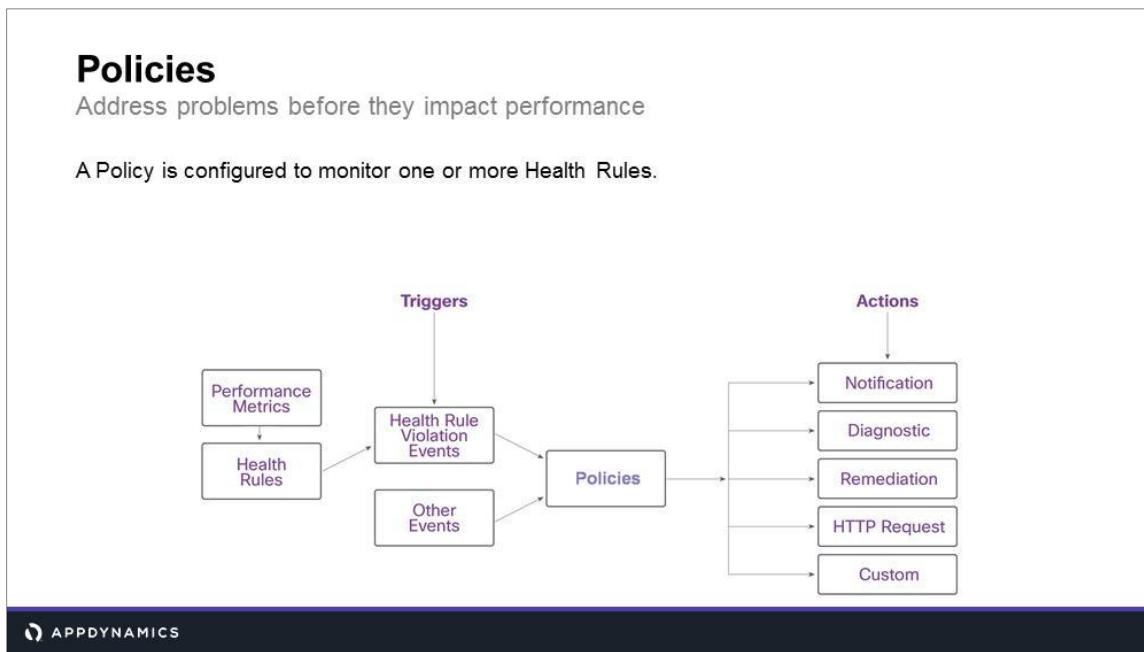
Any Health Rules that you have in your AppDynamics Controller should be configured according to your expected performance thresholds. I.e. you want to get the Warning level (yellow) messages when performance shifts away from normal and towards less than ideal.

And you also want to set your Critical level (red) Health Rule logic at a point where you really now have a problem and need to react quickly.

While the red messages will always need to be attended to, it is worth assessing what is causing the yellow level to violate as well, so that you can hopefully avoid the same Health Rule going to red at some later stage.

Dealing with the yellows will help you avoid the reds.

Policies, Actions, and Runbook Automation



Trigger - defines the Event(s) that causes the Policy to fire.

Action - an activity that is executed when a Policy is triggered.

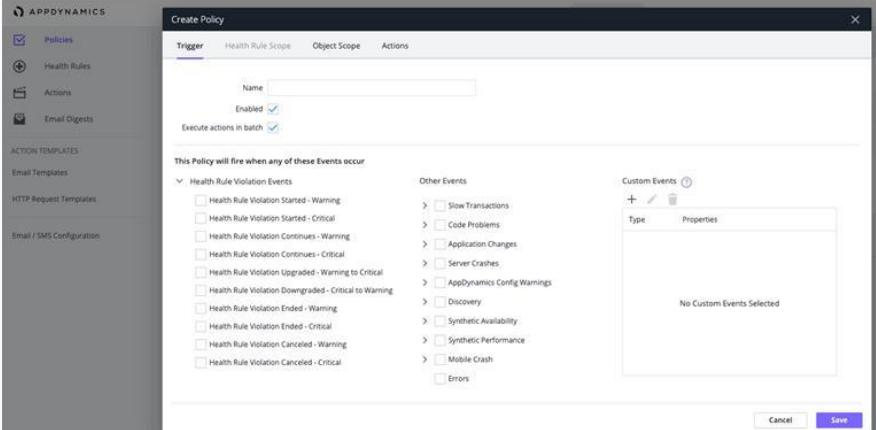
Health Rule Scope - the Health Rule(s) to be monitor by a Policy.

Policies let you anticipate problems and take Actions to address those problems before they cause a severe slowdown or outage.

A Policy has two parts: (1) trigger, and (2) action. With a Policy you are basically connecting a Health Rule violation or other event with an Action, so the Action will be triggered automatically as soon as the Event occurs.

Because the definition of Health Rules is separate from the definition of Actions, and both Health Rules and Actions can be very precisely defined, you can take different actions for breaching the same thresholds based on context, for example, which Tier or Node the violation occurred in.

Triggers



The screenshot shows the 'Create Policy' dialog box in the AppDynamics interface. The 'Trigger' tab is active. The 'Name' field is empty. The 'Enabled' and 'Execute actions in batch' checkboxes are checked. The 'Health Rule Violation Events' section contains 14 checkboxes for various health rule violation states. The 'Other Events' section contains 10 checkboxes for other event types. The 'Custom Events' section is empty. At the bottom, there are 'Cancel' and 'Save' buttons.

An event is a change in application state that can potentially be interesting to you. AppDynamics detects 9 different types of events automatically, and lists them in the Events tab on the application dashboard.

You can drill down into events of any type for more information. In the case of some events, such as health rule violations, you can view the state of the dashboard at the moment when the particular event occurred.

A summary of the event is available on the application dashboard, and you can access the Event List page from the Summary header, or by clicking on Events on the left hand navigation pane. You can apply filters to narrow down the list, and drill down into any of the specific events from here also.

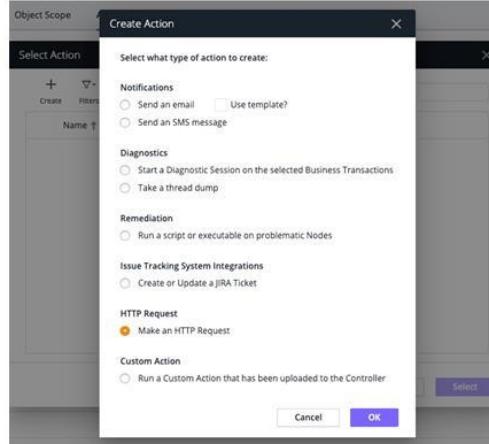
Let's look at a few common events, and the kind of information captured for each.

Actions

Creating a new Action - selecting the kind of activity

Options for an Action include:

- Send an Email (per template)
- Send an SMS
- Start a Diagnostic Session
- Execute Thread Dump
- Run a script
- Make an HTTP Request
- Run a Custom Action



APPDYNAMICS

An action is a predefined, reusable, automated response to an event, and it supports various types of responses as listed above.

A common action is to send a notification by email or SMS to a recipient list. The text of the notification is automatically generated by the event that triggered the action.

Another common action is to start a diagnostic session to collect snapshots or direct the agent to execute a thread dump for a specified number of samples (maximum of 50) with each sample lasting for a specified number of milliseconds (maximum of 500 ms).

Beyond these typical responses, with actions you can also run local scripts in a node. The script executes on the machine from which it was invoked, or on the machine specified by the remediation action configuration. You can use this type of action to automate your runbook procedures. With cloud auto-scaling, you can scale your application to your current capacity requirements, using a workflow for adding or removing one or more application servers.

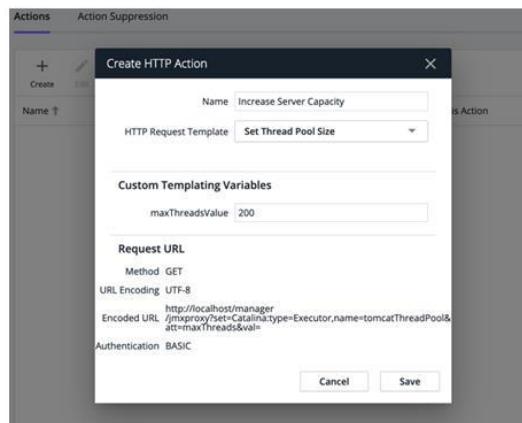
You can also set a custom action that's typically used to integrate with third-party alerting and ticketing systems. Custom actions are commonly used when you want to trigger a human workflow or leverage an existing alerting system that is external to AppDynamics. For example, you could use a custom action to file a support ticket when AppDynamics reports that a connection pool is near saturation.

Actions

Creating a new Action - providing settings for chosen Action type

Depending on the kind of Action selected, there are different options to configure.

For example, an HTTP Action requires an HTTP Request Template to be defined.



APPDYNAMICS

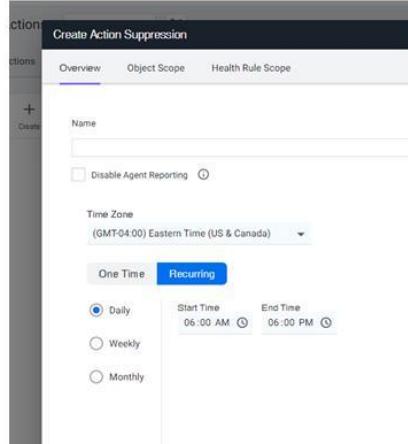
Different kinds of Action will require different settings to be specified.

In the case of HTTP Action, which basically means that the Controller will make an HTTP Request on your behalf, the HTTP Request Template contains all the details necessary to make the request. When you configure the Action you can define variables to control dynamic elements of the HTTP Request, such as defining how many Threads the server should use when the Request is received.

Templating Variables can also be set as required.

Action Suppression

- Temporarily suppress a policy's automatic invocation of actions and alerts
- Useful during troubleshooting or maintenance
- Suppression can occur for:
 - Applications
 - Business Transactions
 - Tiers
 - Nodes
 - Servers



APPDYNAMICS

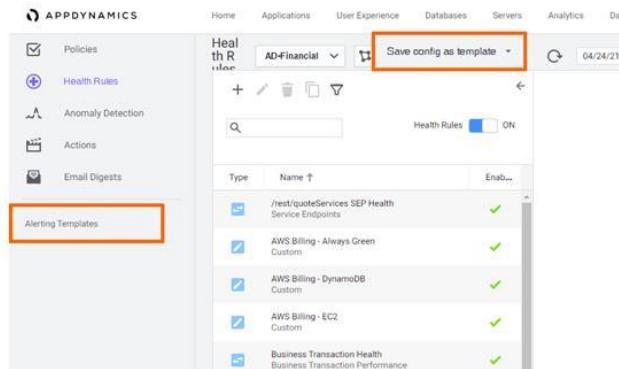
Action Suppression is a separate tab on the Actions page. Select the timerange, the scope (application, transaction, etc). You can also include specific Health Rules to suppress.

You can elect to disable Agent Reporting.

Suppressing an action does not suppress the evaluation of any health rules linked to the action through a policy. Health rule violation events continue to be raised even when actions are suppressed.

Alerting Templates

- A configuration of features such as Health Rules, Policies, Actions, Action Suppression, Schedules, and Email Digests
- Initiate from each Alert and Respond tool, or click Alerting Templates in the left navigation pane

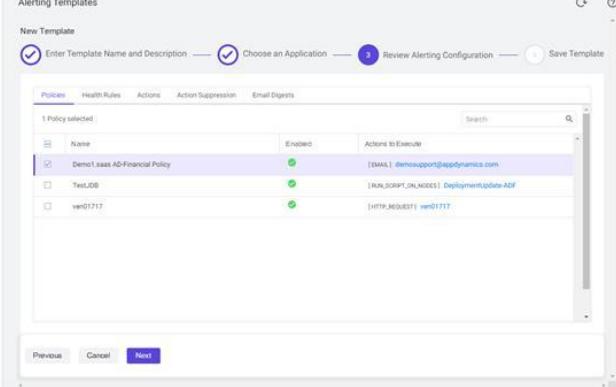


The screenshot shows the AppDynamics interface for managing alerting templates. The left sidebar has a navigation menu with 'Alerting Templates' highlighted by an orange box. The main content area is titled 'Health Rules' and shows a list of rules with columns for Type, Name, and Enabled status. A 'Save config as template' button is highlighted with an orange box. The interface includes a toolbar with various icons and a date stamp of 04/24/21.

Type	Name	Enabled
Custom	/rest/quoteServices SEP Health Service Endpoints	✓
Custom	AWS Billing - Always Green	✓
Custom	AWS Billing - DynamoDB	✓
Custom	AWS Billing - EC2	✓
Business Transaction Health	Business Transaction Performance	✓

Configure and Manage Alerting Templates

- An existing alerting template cannot be edited, but it can be overwritten
- An alerting template can be applied to multiple applications
- Alerting templates can be imported/exported between accounts



APPDYNAMICS

After configuring an alerting template, if you require to change the configuration and apply the changes to all the applications, use the Overwrite an existing template option. You cannot edit an alerting template.

Runbook Automation		
Remediation Actions		
Remediation Script	HTTP Request	Custom
An action that executes an arbitrary script on an individual Node. May require human approval before executing. Requires a Java standalone machine agent.	An action that runs on the Controller, and makes an HTTP request.	A custom Action is typically used to integrate third party alerting and ticketing systems.



Remediation scripts are scripts that you, the customer, define for your environment and applications, to do whatever you want them to do.

The scripts themselves do not need to adhere to any AppDynamics format or layout.

They will be executed for you by the Machine Agent, so they have to be installed under the Machine Agent install directory.

If the remediation script logs output, the log can be viewed in the Controller.

Integrations to third party solutions such as creating a IBM NetCool Alert via an AppDynamics Alert can be accomplished via a Custom Action.

Notes From the Field

Runbook Automation

Business Problem

- Customer had regular issues with vendor service performance.
- Manual RDP login was required to restart the service.

Solution

- Set up Health Rules to monitor and used Runbook Automation to do the Restart when necessary.
- Additional notification sent to vendor and Ops team if issue persisted.
- Also created a JIRA ticket to track the issue.



APPDYNAMICS

Runbook Automation not only makes your life easier, it can potentially save you a small fortune in lost revenue.

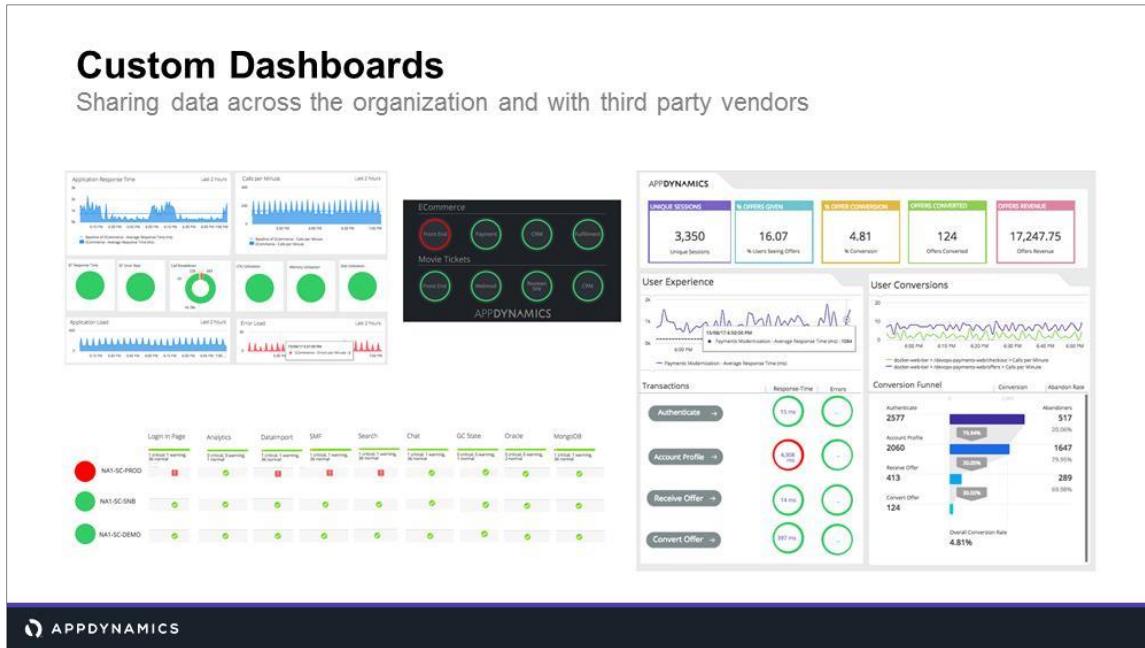
One customer we know used to have a service provided by a vendor that handled bill payments for them. There were regular issues with the performance dropping off, which the customer knew could be fixed (in the short-term at least) by simply restarting the service. However this required RDP-ing into the server to do it, and was a manual process.

With the adoption of AppDynamics, this customer realized they could set up Health Rules to monitor the performance of the bill payment service - then on top of the Health Rule they could define Policies and Actions and have an automated Runbook, to take care of the server restart for them, based on detecting a drop-off in the performance of the service.

But the customer went further still - they set up an additional Policy and Actions to detect when the service Health Rule continued to be Critical, at which point an Action to notify the vendor was sent (because the restart on the customer server hadn't fixed the problem) and a new Jira ticket created internally so the customer could track the issue closely, as well as the Ops team getting notified of the problem.

Once the issue had been fixed, and the service began running normally again, they used the fact that the Health Rule either stopped violating, or was cancelled, to trigger notifications to all the interested parties and automatic closing of the Jira ticket.

Custom Dashboards and Reports



Custom Dashboards provide a flexible, visual way to present the specific information you want to see at a glance.

You determine what data is displayed and how it is visualized.

Depending on your role, you may have a specific set of metrics you are interested in tracking. If you work in Operations, you are interested in metrics that deal with server performance; if your colleague is a developer, he or she may be more interested in comparing metrics that have to do with performance of the application code.

AppDynamics offers the ability to create custom dashboards, so everyone can have a dashboard that presents all the data that's relevant to their responsibilities.

Custom Dashboards Features

Aggregate key data in a single dashboard

- Create widgets from multiple AppDynamics products
 - Highlight alerts with status lights
 - Double-click on widget to link to Metric Browser OR other URL
 - Display browser content via iFrame widget
 - Use your own branding
 - Pure HTML 5
 - Share with others via a URL, including non-AppDynamics users
- Two layout options available:
 - **Grid (default):** flexible easy alignment, can specify titles for widgets, will scale on a mobile device, cannot overlap widgets
 - **Absolute:** Can specify exact widget dimensions, can overlay widgets, cannot specify titles for widgets, does not scale
 - Multiple Y-Axis support to enable comparing data of different units or scales
 - Drill down from the dashboard directly to the underlying objects and data



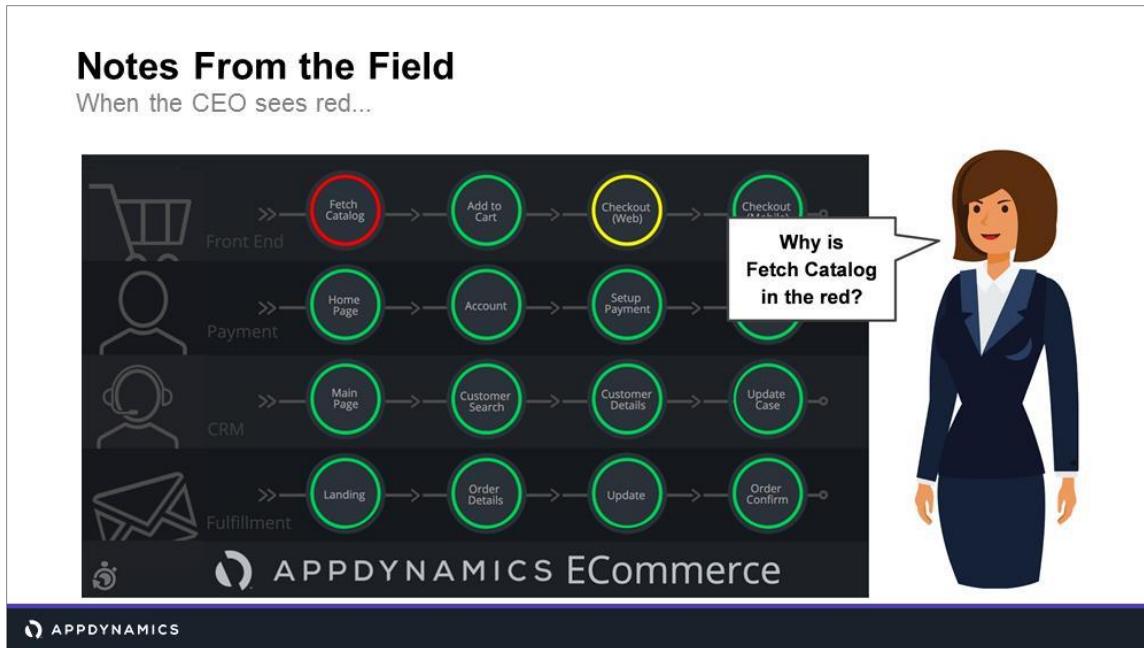
Custom dashboards have several features that enable you to do more with AppDynamics. You can brand the dashboard with your company logo and colors. They are built with HTML5 so you can check your application's health from anywhere, at any time, on any device.

By default only users logged into the controller with the necessary permissions can view a Custom Dashboard. However, if you share a dashboard, it becomes available to anyone possessing the shared URL, regardless of whether they can log into AppDynamics or not. You can also stop sharing a dashboard if you don't wish to make data available to a wider audience indefinitely.

Further, with the use of the iFrame widget, you can embed HTML content or insert data from other monitoring applications and make this custom dashboard the single pane of glass to view all available application performance information.

Custom dashboards can present baseline information along with actual metrics, so it's easy to compare and detect anomalies, and the status light feature presents health rule status visually so you know the health status right away. And if you detect anomalies, you can drill down to the root cause of the problem right from the dashboard.

Additionally, you can import & export dashboards, so you can use the same dashboard in Dev, QA, and Production environments without creating each one separately.



One AppDynamics user experienced the power of the Dashboard when their CEO wandered past their desk and spotted a widget showing red. The CEO asked what was wrong (of course) and the Ops member was able to quickly give the following answer:

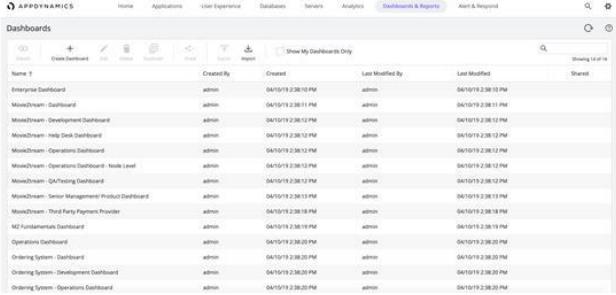
“The system is letting us know of some higher than average errors with the Fetch Catalog functionality, which we’ve already investigated and figured out is due to the third party look-up service. We’ve already switched to the backup service and contacted the third party provider to report the failure. We were only impacted for 2 minutes.”

The CEO was satisfied and impressed with the quick and thorough response, which justified the expense of purchasing and configuring AppDynamics the way they had.

Custom Dashboards

View and Administer Existing Dashboards

- Go to **Dashboards & Reports** from the menu at the top of the screen.
- Click **Dashboards** to:
 - Review existing Dashboards
 - Select to modify, delete, copy, export
 - Import dashboard exported from another Controller



Name	Created By	Created	Last Modified By	Last Modified	Shared
Enterprise Dashboard	admin	04/19/19 2:38:10 PM	admin	04/19/19 2:38:10 PM	
MobileTeam - Dashboard	admin	04/19/19 2:38:11 PM	admin	04/19/19 2:38:11 PM	
MobileTeam - Development Dashboard	admin	04/19/19 2:38:12 PM	admin	04/19/19 2:38:12 PM	
MobileTeam - Help Desk Dashboard	admin	04/19/19 2:38:12 PM	admin	04/19/19 2:38:12 PM	
MobileTeam - Operations Dashboard	admin	04/19/19 2:38:12 PM	admin	04/19/19 2:38:12 PM	
MobileTeam - Operations Dashboard - Node Level	admin	04/19/19 2:38:12 PM	admin	04/19/19 2:38:12 PM	
MobileTeam - QA Testing Dashboard	admin	04/19/19 2:38:12 PM	admin	04/19/19 2:38:12 PM	
MobileTeam - Senior Management/ Product Dashboard	admin	04/19/19 2:38:13 PM	admin	04/19/19 2:38:13 PM	
MobileTeam - Third Party Payment Provider	admin	04/19/19 2:38:18 PM	admin	04/19/19 2:38:18 PM	
MQ Fundamentals Dashboard	admin	04/19/19 2:38:19 PM	admin	04/19/19 2:38:19 PM	
Operations Dashboard	admin	04/19/19 2:38:20 PM	admin	04/19/19 2:38:20 PM	
Ordering System - Dashboard	admin	04/19/19 2:38:20 PM	admin	04/19/19 2:38:20 PM	
Ordering System - Development Dashboard	admin	04/19/19 2:38:20 PM	admin	04/19/19 2:38:20 PM	
Ordering System - Operations Dashboard	admin	04/19/19 2:38:20 PM	admin	04/19/19 2:38:20 PM	

You can copy existing dashboards, so if you are creating a few variations of a similar dashboard, you do not have to start each from scratch.

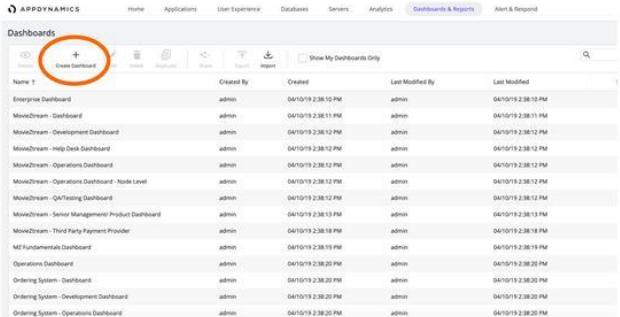
This is also a good way to make a backup copy when you are editing an existing dashboard.

It is a good idea to use a common naming convention for naming Dashboard to keep them organized and descriptive. E.g., [System Name] - [Role] (see screenshot).

Custom Dashboards

Creating a new Dashboard

Click **Create Dashboard** button, or select an existing Dashboard and click **Duplicate** to clone it.



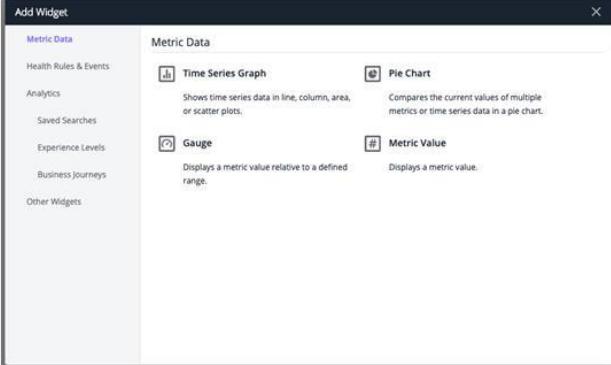
Name	Created By	Created	Last Modified By	Last Modified
Enterprise Dashboard	admin	04/10/19 2:38:10 PM	admin	04/10/19 2:38:15 PM
MovieDream - Dashboard	admin	04/10/19 2:38:11 PM	admin	04/10/19 2:38:11 PM
MovieDream - Development Dashboard	admin	04/10/19 2:38:12 PM	admin	04/10/19 2:38:12 PM
MovieDream - Help Desk Dashboard	admin	04/10/19 2:38:12 PM	admin	04/10/19 2:38:12 PM
MovieDream - Operations Dashboard	admin	04/10/19 2:38:12 PM	admin	04/10/19 2:38:12 PM
MovieDream - Operations Dashboard - Node Level	admin	04/10/19 2:38:12 PM	admin	04/10/19 2:38:12 PM
MovieDream - QA Testing Dashboard	admin	04/10/19 2:38:12 PM	admin	04/10/19 2:38:12 PM
MovieDream - Senior Management Product Dashboard	admin	04/10/19 2:38:12 PM	admin	04/10/19 2:38:12 PM
MovieDream - Third Party Payment Provider	admin	04/10/19 2:38:18 PM	admin	04/10/19 2:38:18 PM
ME Fundamentals Dashboard	admin	04/10/19 2:38:19 PM	admin	04/10/19 2:38:19 PM
Operations Dashboard	admin	04/10/19 2:38:20 PM	admin	04/10/19 2:38:20 PM
Ordering System - Dashboard	admin	04/10/19 2:38:20 PM	admin	04/10/19 2:38:20 PM
Ordering System - Development Dashboard	admin	04/10/19 2:38:20 PM	admin	04/10/19 2:38:20 PM
Ordering System - Operations Dashboard	admin	04/10/19 2:38:20 PM	admin	04/10/19 2:38:20 PM

APPDYNAMICS

Custom Dashboards

Selecting Widgets

- Define Dashboards by Adding Widgets
- Select Metric(s) or Event(s)
- Choose colors, styles, fonts etc.



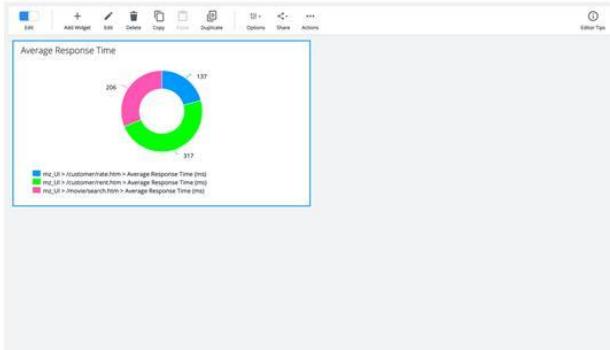
APPDYNAMICS

Building the Dashboard is an iterative process, where you select a widget, then configure what it should display and select the style of it (size, colors, font, etc), then select another widget and configure that widget as you like.

Custom Dashboards

Arranging Widgets

- Drag widget to desired location
- Resize widget by dragging corners



Tier & Node Dashboards

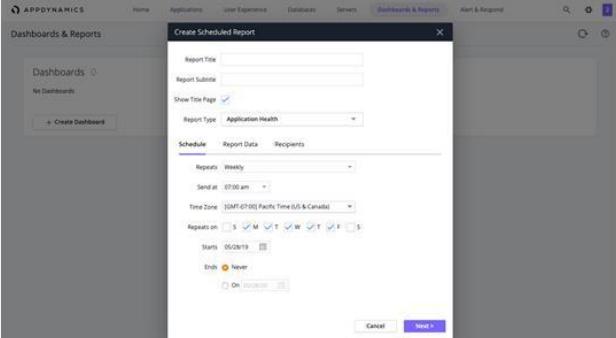
- Create Tier or Node level Dashboards
- Same principles and approach as Custom Dashboards
- Save Dashboard as Template for re-use on other Tiers or Nodes

You can create dashboards for Tiers or Nodes in pretty much the same way you build Custom Dashboards. In this case though, the saving of the dashboard automatically means that you have a template which can be used to set up a dashboard for the other (similar) Tiers or Nodes. Logically this makes most sense if you consider that you would often have several Nodes in a Tier, all configured the same and all needing similar dashboards: you only need to build the dashboard once and roll it out across the other Nodes.

Scheduled Reports

Dashboards & Reports > Reports > Scheduled Reports
tabs

- Enable/Disable reports
- Import/Export reports in JSON format
- Application Health, Custom Dashboard, Controller Audit, Home Screen, etc.
- Weekly, daily, hourly...
- Email to list of recipients



APPDYNAMICS

Reports can be scheduled and distributed automatically via email.

You can enable and disable reports as well as import and export reports. The exported report file type is in a JSON format.

Review Question - Answer

It is possible to share a custom dashboard with non-AppDynamics users.

True

False



One caveat: if you are on an on premises instance of AppDynamics, you may need to open up a port to share your dashboard(s) out.

What you learned

- The difference between Errors and Exceptions in AppDynamics
- How to configure AppDynamics to ignore specific Exceptions
- How to use baselines to evaluate application health over time
- How to configure Health Rules
- How to configure Policies and Actions to respond to Health Rule Violations
- How to create dashboards to share data
- How to create scheduled reports

Differentiate yourself and your company with AppDynamics Certification

APPDYNAMICS | Certification Program

General Certification Information

<https://learn.appdynamics.com/certifications>

AppDynamics Certified Associate Performance Analyst

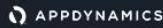
<https://learn.appdynamics.com/certifications/performance-analyst>

AppDynamics Certified Associate Administrator

<https://learn.appdynamics.com/certifications/administrator>

AppDynamics Certified Implementation Professional

<https://learn.appdynamics.com/certifications/implmenter>



To differentiate yourself and your company in an increasingly competitive market, please consider becoming AppDynamics certified.

For more information, please copy and paste this URL into a separate tab in your browser: <https://learn.appdynamics.com/certifications>

For information on specific certification tracks, please copy and paste the appropriate URLs below:

AppDynamics Certified Associate Performance Analyst:

<https://learn.appdynamics.com/certifications/performance-analyst>

AppDynamics Certified Associate Administrator:

<https://learn.appdynamics.com/certifications/administrator>

AppDynamics Certified Implementation Professional:

<https://learn.appdynamics.com/certifications/implmenter>

Advanced Troubleshooting and Monitoring (APM224)



University

APM224 - Advanced Troubleshooting and Monitoring

Core APM II: Advanced - Module 4

Objectives

Course

After completing this course, you will be able to:

- Explain the purpose and applications for Information Points
- Configure Information Points to collect Custom Metric data
- Explain the purpose and applications for Data Collectors
- Configure Data Collectors to add information to Snapshots
- Enable Development Level Monitoring to collect full call graph snapshots
- Explain when and when not to enable Development Level Monitoring
- Configure JMX Metrics to monitor Java Applications
- Use Windows Performance Counters to collect and display information
- Identify and troubleshoot memory management issues
- Use Automatic Leak Detection to find collection-based memory leaks
- Use Object Instance Tracking to find object-based memory Leaks

Labs

In this course's labs, you will:

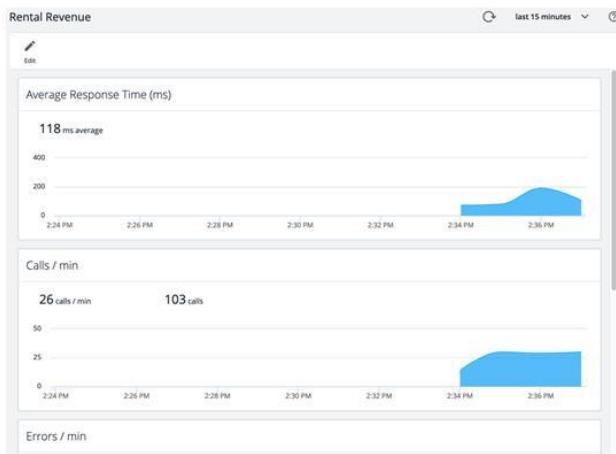
- Collect Data with Information Points
- Capture Business Data using Data Collectors
- Resolve Data-dependent Issues with Data Collectors
- Use Development Level Monitoring to capture more snapshots
- Define Custom JMX Metrics (Optional, Java only)
- Troubleshoot Memory Issues with ALD (Java)
- Troubleshoot Memory Issues with OIT (.NET)

Creating and Using Information Points

Information Points

What they do

- Information Points collect numeric data from methods. For example:
 - Average Response Time
 - Calls per Minute
 - Errors per Minute
 - Custom Metrics (optional)
- Configured from **More > Information Points**
- Not defined in relation to any Business Transactions



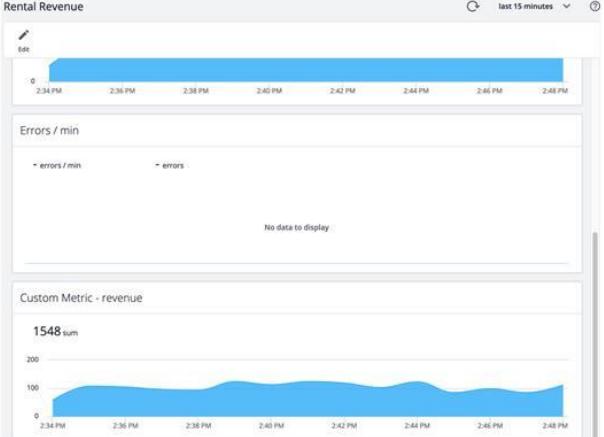
An information point is used to instrument a method. Information points capture numeric data that fall into two categories: Code Metrics and Custom Metrics.

Code metrics evaluate how a method is performing across the application.

Information Points

Custom Metrics

- Custom Metrics typically relate directly to critical business data so we sometimes call them Business Metrics.
- They are based on numerical values processed by an instrumented method
 - Method Parameters
 - Method Returns
- Configure new metric to be Average or Sum of values collected.



The screenshot shows the AppDynamics interface with two charts. The top chart is titled 'Rental Revenue' and shows a blue area chart with data points from 2:34 PM to 2:48 PM. The y-axis ranges from 0 to 200. The bottom chart is titled 'Custom Metric - revenue' and shows a blue area chart with a single data point labeled '1548 sum' at the top. The y-axis ranges from 0 to 200. Both charts have x-axis ticks at 2:34 PM, 2:36 PM, 2:38 PM, 2:40 PM, 2:42 PM, 2:44 PM, 2:46 PM, and 2:48 PM. A message 'No data to display' is visible in the middle chart area.

APPDYNAMICS

Custom Metrics, also called Business Metrics, evaluate data from a method's parameters or return values, or anything that could be called with a getter chain. Custom metrics are aggregated – either averaged or summed over time.

For example, if an information point is configured to get metrics about a particular method invocation, such as credit card authentication, any time that authentication method is invoked the information point is triggered.

Information Point Examples

1. How many times was the method executed?
2. How long did it take to execute on average?
3. Are there errors occurring when the method is being executed? If so, how often?
4. What is the average value of the credit card total?
5. How many credit cards did my application process in a certain time period, regardless of the Business Transaction?
6. What was the average time spent processing a credit card transaction?
7. What is the current rate of rejected credit cards?

Information points can capture how a method is performing, as well as data from a method's parameters or return values to report on the performance of the business.

Notes From the Field

Database recovery time

Business Problem

- PlaceOrder transaction makes calls to several services, including a DB.
- During HA/DR testing, discovered that the DB was not HA, manually would require over an hour to recover.
- Customer did not have budget to instrument the DB with HA, despite possible impact of over 20,000 customers.

Solution

- Created an Information Point to track revenue for the PlaceOrder transaction. This revealed that over \$225k would be lost in just 1 hour.
- They found the money to add HA to the database.



APPDYNAMICS

- Testing HA/DR (high availability/disaster recovery)
- PlaceOrder transaction would make calls to several services, including a database.
- During testing, the DB took over an hour to recover without HA.
- Upon informing the Product Manager of the long recovery time for this function, his response was that they did not budget for adding HA to the database and they would just live with it.
- Showed him the flow map with Timerange set to 1 hour. This showed that over 20,000 customers would be impacted in that time. Incredibly, this did not seem to change his mind about the cost to HA the database.
- Created an Information Point that tracked Total Order Revenue for this transaction.
- The Information Point revealed that over \$225,000 in revenue would be lost or delayed in just that one hour.
- After seeing the impact in lost dollars, they were quickly able to justify the cost of instrumenting the DB.

Review Question - Answer

You can collect string-based data using Information Points.

True

False



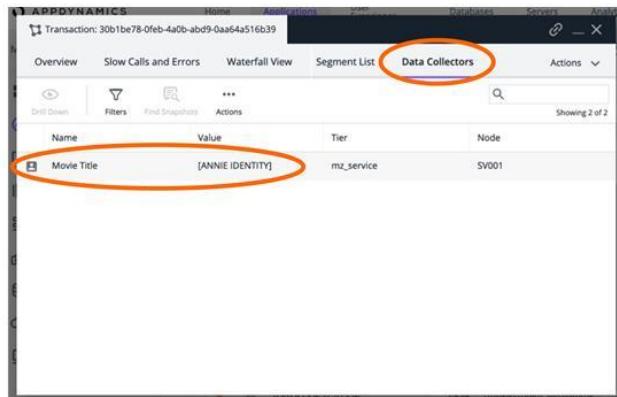
Information Points only capture numeric data.

Creating and Using Data Collectors

Data Collectors

What they do

- Collect user-specified data from transactions
- Used specifically to troubleshoot transactions with data-dependent problems
- Captured data appears in Snapshot



Name	Value	Tier	Node
Movie Title	[ANNIE (IDENTITY)]	mz_service	SV001

Data Collectors can collect any data passing through an application as long as it can be displayed as a string (opposed to Information Points that gather numeric data).

They can help determine whether data that a transaction passed into an application is causing problems.

When you configure Data Collectors, AppDynamics accesses information in application code arguments, return values, and getter chain and displays the information in the Call Drill Down panels.

Data Collectors are also used to collect data for the Analytics Engine.

Notes From the Field

Sending unencrypted data

Business Problem

- Customer concerned about sending unencrypted data to partners, making the data vulnerable.

Solution

- Set up Data Collectors against key API methods for outbound calls to partner orgs.
- Review sample Data Collector outputs to see if data is properly encrypted.

Am I sending unencrypted client data to partners?

How can I check?



 APPDYNAMICS

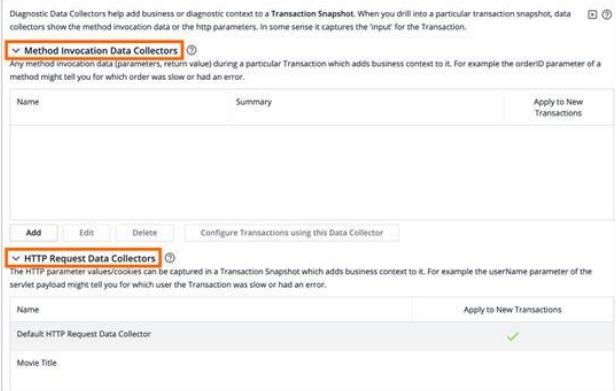
Data Collectors can be used in a number of contexts, but principally the benefit is the ability to expose data values hidden within the running programs - any data values!

One customer was concerned that they might be sending unencrypted data across to partner organizations, potentially meaning that internet snoopers could pick up customer details such as names, addresses and credit card numbers. To check if this was happening they configured Data Collectors against key API methods in their system which handled outbound calls to the partner orgs. Then all they needed to do was review sample Data Collector outputs in the Snapshots to see if the data was unencrypted or properly encrypted as they hoped it was.

Two Types of Data Collectors

Method Invocation Data Collectors
capture values from method invocations

HTTP Request Collectors
capture values within HTTP requests



Diagnostic Data Collectors help add business or diagnostic context to a Transaction Snapshot. When you drill into a particular transaction snapshot, data collectors show the method invocation data or the http parameters. In some sense it captures the 'input' for the transaction.

Method Invocation Data Collectors

Any method invocation data (parameters, return value) during a particular Transaction which adds business context to it. For example the orderID parameter of a method might tell you for which order was slow or had an error.

Name	Summary	Apply to New Transactions

Add Edit Delete Configure Transactions using this Data Collector

HTTP Request Data Collectors

The HTTP parameter values/cookies can be captured in a Transaction Snapshot which adds business context to it. For example the userName parameter of the servlet payload might tell you for which user the Transaction was slow or had an error.

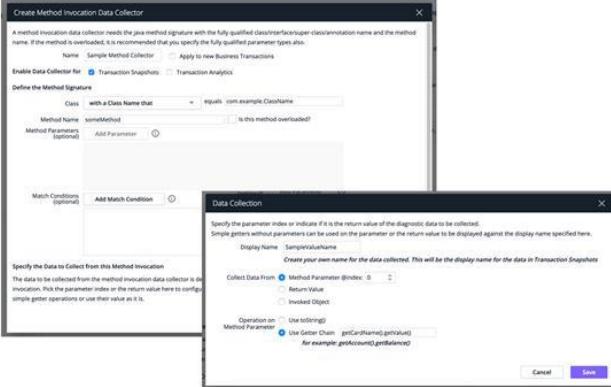
Name	Summary	Apply to New Transactions
Default HTTP Request Data Collector		✓
Movie Title		

Configuration > Instrumentation > Data Collectors

Method Invocation Data Collectors

Method Invocation Data Collectors

- Method arguments
- Variables
- Return values



The screenshot shows two overlapping windows. The top window is titled 'Create Method Invocation Data Collector' and contains fields for 'Name' (Sample Method Collector), 'Enable Data Collector For' (Transaction Snapshots), and 'Define the Method Signature' (Class: com.example.CoolName, Method Name: someMethod, Method Parameters: (None)). The bottom window is titled 'Data Collection' and contains fields for 'Display Name' (SampleParameterValue), 'Collect Data From' (Method Parameter (Index: 0)), 'Operation on Method Parameter' (Use Getter Chain: getCardName().getValues()), and a preview of the code 'for example: getCardName().getValues()'.

APPDYNAMICS

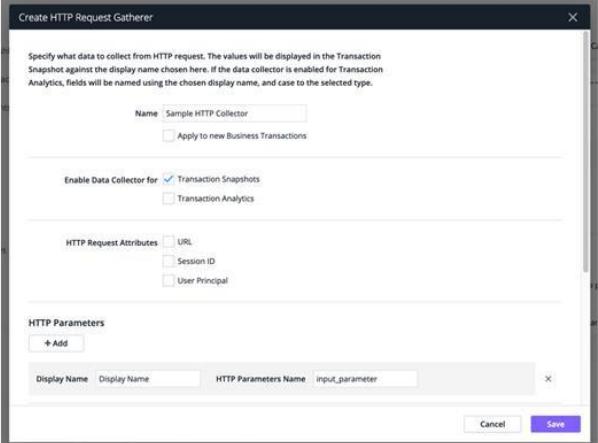
Method Invocation Data Collectors capture code data such as method arguments, variables, and return values.

Use them to create custom fields to collect business related data.

HTTP Data Collectors

HTTP Request Collectors

- URL
- Parameter values
- Headers
- Cookies
- Session objects

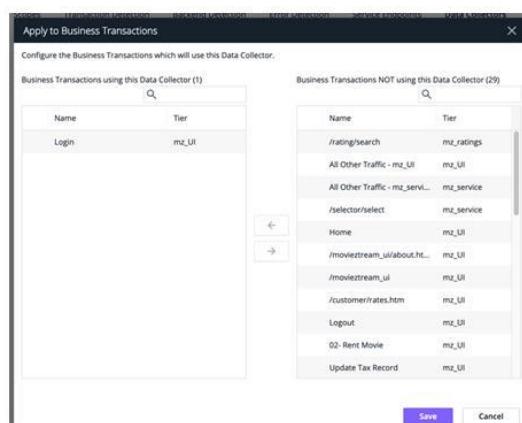


APPDYNAMICS

HTTP Data Collectors capture the URLs, parameter values, headers, and cookies of HTTP messages exchanged in a Business Transaction.

Configure BTs for Data Collector

Whether using an HTTP or Method Invocation Data Collector, the final step is to select which BT(s) it applies to.



APPDYNAMICS

While we recommend selecting only the particular BTs that may contain the data that you expect the Data Collector to collect, generally it is not harmful to select other BTs. If the BTs don't handle the data field that the Data Collector is trying to extract, the Data Collector will simply ignore them.

Data Collectors and Security

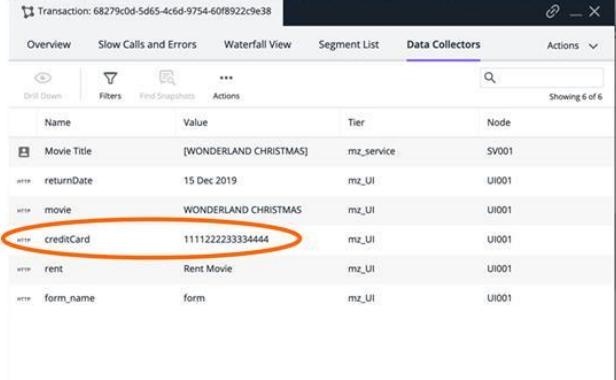
Take care who can configure Data Collectors.

Only grant authorization to trusted parties.

Either of the Data Collector types can be disabled:

- Java Agent: set the node property, in the Controller UI: disabled-features

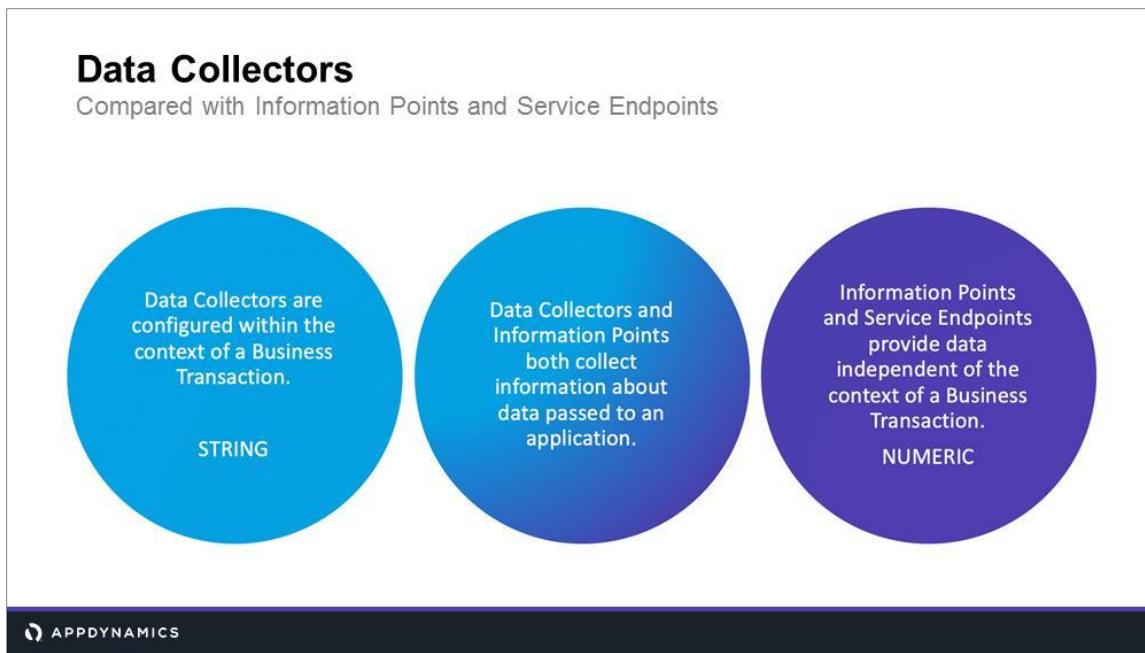

```
<property name="disabled-features" value="METHOD_INV_DATA_COLLECTOR, HTTP_DATA_COLLECTOR"/>
```
- .Net Agent: edit the config.xml, set the disabled-features property for each Data Collector type



Name	Value	Tier	Node
Movie Title	[WONDERLAND CHRISTMAS]	mz_service	SV001
returnDate	15 Dec 2019	mz_UI	UI001
movie	WONDERLAND CHRISTMAS	mz_UI	UI001
creditCard	1111222233334444	mz_UI	UI001
rent	Rent Movie	mz_UI	UI001
form_name	form	mz_UI	UI001

Creating a Data Collector opens up the ability to reveal values of ANY data being handled by your code, so don't grant the authority to define Data Collectors to all your staff, rather restrict it to a select few Administrators.

Note that AppDynamics does write to its own Audit log, capturing actions such as the creation or modification of Data Collectors, which can then be reviewed by generating the standard Audit Report within the Controller.



You may be wondering how the Data Collectors differ from Information Points. They are very similar as they collect information about data passed to an application, and both must be explicitly configured.

Data Collectors add STRING data to a Business Transaction Snapshot. Data from Data Collectors appear in the panels of the Transaction Snapshot call drill down - HTTP DATA, COOKIES, or USER DATA - depending on the type of the Data Collector. Data Collectors do not appear in the Metric Browser and cannot be used in Health Rules. You can filter Transaction Snapshots based on the value of a Data Collector in the Transaction Snapshot list or using the AppDynamics REST API.

Information Points aggregate numeric data about invocations of a method outside the context of any Business Transaction. Use Information Points to get metrics about method invocations across zero, one or multiple Business Transactions. Any time the method configured for the Information Point is invoked, no matter from where, the Information Point is triggered.

Like Business Transactions, Service Endpoints give you key performance indicators, metrics, and snapshots. However, like Information Points, they provide that information independent of the context of a Business Transaction or downstream performance data.

Notes From the Field

Data Collectors

Business Problem

- A client used a 3rd party vendor to manage credit card payments.
- Make Payments transaction suddenly started failing 10% of the time.
- Because a service endpoint was set up, we could see the issue was on the vendor side.

Solution

- The root cause was that the payment processor had re-written the way they did CC validation. The third party payment provider built and tested to the new interface and it was fine.
- The issue was the payment processor was using GMT for their deployments and the 3rd party payment provider was on CT.



© APPDYNAMICS

Review Question - Answer

Which would you use for troubleshooting?

- A) Data Collectors
- B) Information Points
- C) Both



Data Collectors are for troubleshooting. Information points (and Service Endpoints) are for monitoring.

Review Question - Answer

Data Collectors can only collect numeric information.

- A) True
- B) False**



Data Collectors can collect data of any primitive type, i.e. Strings, Numbers, Dates, Booleans.

You do not always have to explicitly configure which BTs the Data Collector applies to, since there is a checkbox in the Data Collector definition screen that says “Apply to new Business Transactions.” Of course, for existing BTs, yes you do need to manually select them to apply to the Data Collector.

Using Development Level Monitoring

Development Level Monitoring

Purpose and Scope

- Use Development Level Monitoring to test performance during non-production environments, without concern for the extra load generated from excessive snapshot collection.
- Diagnose Business Transaction specific problems by collecting detail-rich information for all requests monitored by a particular Business Transaction.
- Full call graph Snapshots are collected for every request.
- Backend Calls below 10ms are collected.
- Using Development Level Monitoring in your Production environment may result in a significant performance impact.



This is a useful feature where it is possible to gain development level monitoring for high value Business Transactions.

Using the Development Level Monitoring on a specific Business Transaction will provide more context.

Using Development Level Monitoring in production could in some cases, cause the system to crash. At the very least, depending on the frequency of the selected Business Transaction, Development Level Monitoring could cause a significant performance impact.

Development Level Monitoring

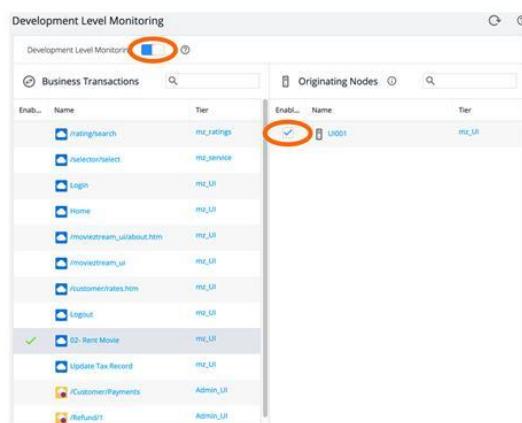
Selecting a Business Transaction to watch

To enable Development Level Monitoring, go to **Configuration > Development Level Monitoring**

Select the desired Business Transaction(s) to monitor from the list.

Check the checkbox to enable (or uncheck to disable).

Controller posts a message confirming enabling (or disabling) of monitoring level.



Enab...	Name	Tier
<input type="checkbox"/>	Rating/search	mtz_ratings
<input type="checkbox"/>	Selector/Select	mtz_service
<input type="checkbox"/>	Login	mtz_UI
<input type="checkbox"/>	Home	mtz_UI
<input type="checkbox"/>	movietracks_about.htm	mtz_UI
<input type="checkbox"/>	movietracks_ml	mtz_UI
<input type="checkbox"/>	ViewCustomerRates.htm	mtz_UI
<input type="checkbox"/>	LoginOut	mtz_UI
<input checked="" type="checkbox"/>	3D_Rent_Movie	mtz_UI
<input type="checkbox"/>	Update Tax Record	mtz_UI
<input type="checkbox"/>	/CustomerPayments	Admin_UI
<input type="checkbox"/>	/RefundIt	Admin_UI

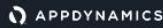
Enab...	Name	Tier
<input checked="" type="checkbox"/>	UI001	mtz_UI

Once Development Level Monitoring has been enabled for the desired Business Transaction(s), the Controller notifies the agents to begin capturing full call graphs for the selected Business Transaction(s), which usually takes a minute or less to take effect.

Development Level Monitoring

Safeguards

- Each node can have configured thresholds as a safeguard.
 - 500 calls/min. or 90% heap utilization are the default thresholds
 - Both thresholds are configurable
- Nodes will switch back to Production mode if thresholds are crossed.
- Mixed nodes (some in Production mode and some in Development Level Monitoring) in an application may result in snapshots with partial call graphs, or some requests without any snapshot collected.



As a safeguard, when the load reaches the maximum number of calls per minute you pre-defined or heap utilization reaches the maximum value you defined for Development, it turns Development Level Monitoring off on that specific node.

Review Question - Answer

Which of the following are true of Development Level Monitoring?

- A) Like Diagnostic Mode, Development Level Monitoring can be turned on for single transactions.
- B) Full call graph Snapshots are collected for every request on an enabled BT.
- C) It can be used in Production environment without risk of significant performance impact to the application.
- D) Depending on the frequency of the selected Business Transaction, Development Level Monitoring can cause significant performance issues.

Configuring JMX Metrics and Windows Performance Counters

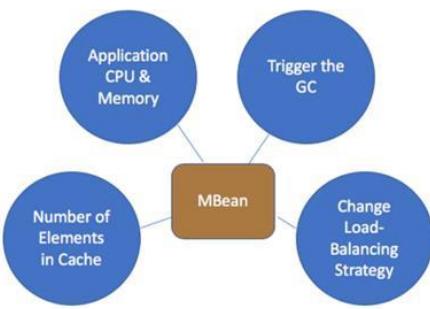
JMX

Introduction

JMX (Java Management Extensions) allows remote monitoring and administration of JAVA components.

These resources are represented by MBeans (Managed Bean) which contain Attributes and Operations.

Java Applications may expose values and methods within the JVM for other programs to read, update or execute.



JMX (Java Management Extensions) is a technology in Java to allow a program to publish state information and make functions available to other programs which connect to the JVM.

For more detail, see:

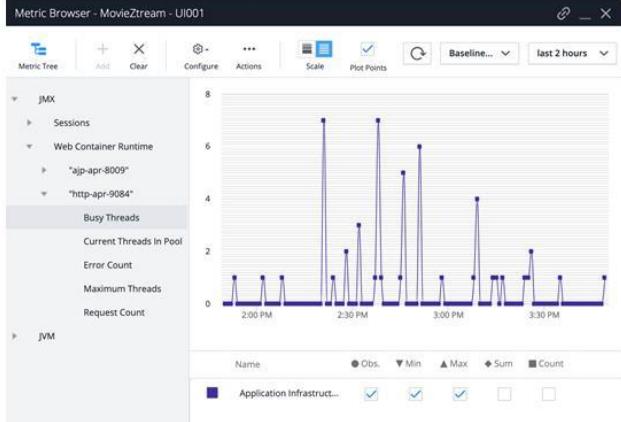
<https://docs.oracle.com/javase/8/docs/technotes/guides/jmx/overview/architecture.html>

JMX Visibility

AppDynamics uses JMX to monitor Java Applications

JMX uses MBeans to expose data via attributes. You can:

- Configure additional JMX Metrics based on these attributes
- Change the value of certain MBean attributes, where field 'Editable' = yes
- Execute JMX operations



APPDYNAMICS

AppDynamics uses JMX (Java Management Extensions) to monitor Java applications. JMX uses objects called MBeans (Managed Beans) to expose data and resources from your application. So to gain visibility into the JMX metrics, you can use one or more MBean attributes to create persistent JMX metrics in AppDynamics. In addition, you can import and export JMX metric configurations from one node, Application or version of AppDynamics to another.

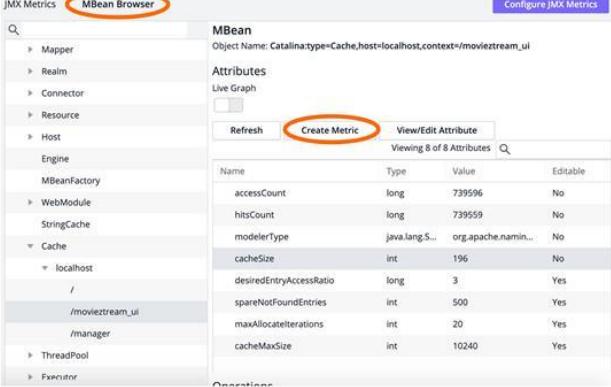
Monitoring JMX Metrics

JMX and MBean Browser tabs

To monitor the trend of an MBean, choose the **MBean Browser** tab.

Select a numerical Attribute and click **Create Metric**.

Editable MBean attributes can be modified from the MBean Browser.



Name	Type	Value	Editable
accessCount	long	739596	No
hitsCount	long	739559	No
modelerType	java.lang.S...	org.apache.naming...	No
cacheSize	int	196	No
desiredEntryAccessRatio	long	3	Yes
spareNotFoundEntries	int	500	Yes
maxAllocations	int	20	Yes
cacheMaxSize	int	10240	Yes

To view JMX metrics, navigate to the Metrics Browser by clicking **Tiers & Nodes > [Tier] > [Node] > JMX**. The Node Dashboard opens. Click the JMX Metrics tab and the JMX Metric Browser opens and displays the MBeans in the left navigation pane.

To monitor a particular metric, drag and drop the metric onto the graph panel.

You can monitor the trend of a particular MBean attribute over time using the Live Graph UI by switching to the MBean Browser sub tab. Select a particular MBean attribute, expand the Live Graph for Attribute section, and click on Start Live Graph. Also, the Operations section is now available to invoke methods.

When you modify an editable MBean attribute in the MBean Browser, the change appears in the Java component.

Configuring Additional JMX Metrics

Modifying JMX metrics

Configuration > Instrumentation > JMX > [choose the platform]

New JMX Configurations can be imported and exported between applications or Controllers.



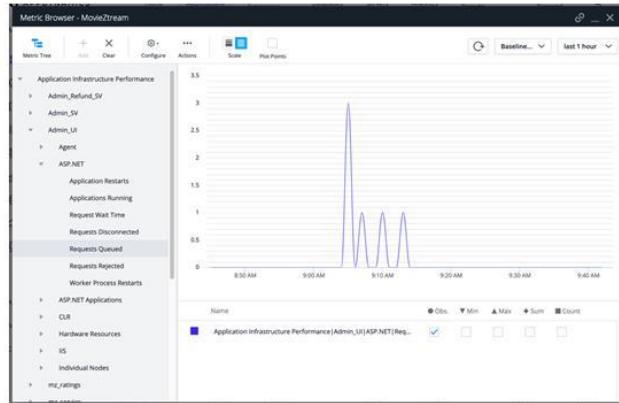
In addition to the out-of-the-box metrics, you can define a new persistent metric using a JMX Metric Rule that maps a set of attributes from one or more MBeans.

Navigate to **Configuration > Instrumentation > JMX**, then choose the platform for which you are configuring metrics. Click on the **+** icon to set up a custom rule.

Windows Performance Counters

Overview

- In Windows operating systems, Performance Counters provide information about:
 - Health of operating system
 - Applications
 - Services or any other components
- Some standard Windows Performance Counters are collected by default.



© APPDYNAMICS

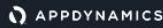
Windows Performance Counters - also called Performance Monitors or 'Perf Mons' - provide monitoring information on processors, memory, and network I/O, as well as performance information on applications, services and drivers. This data can help find system bottlenecks and fine-tune system and application performance.

Windows Performance Counters

Configure new Windows Performance Counter Metrics

- Update .NET Agent config.xml file (e.g., below)
- Restart AppDynamics.Agent.Coordinator Windows service

```
<perf-counters>
  <perf-counter cat="Memory" name="System Driver Total Bytes"
instance="" />
  <perf-counter cat=" ASP.NET Applications "name="Sessions Active"
instance="_LM_W3SVC_1_ROOT_movieztream_admin " />
</perf-counters>
```



AppDynamics reports certain performance counters out of the box. You can configure AppDynamics to capture additional counters as well.

Windows Performance Counters

Configure new Windows Performance Counter Metrics without touching XML

The AppDynamics Performance Counter Configuration tool provides an easy way to look-up available Windows Performance Counters with less risk of human error (e.g., mistyping) than working directly with XML.



The Windows Performance Counter Configuration Extension provides a GUI tool for configuring the monitoring of performance counters exposed by various applications or the operating system itself.

It is a standalone executable and requires Administrator privileges to operate.

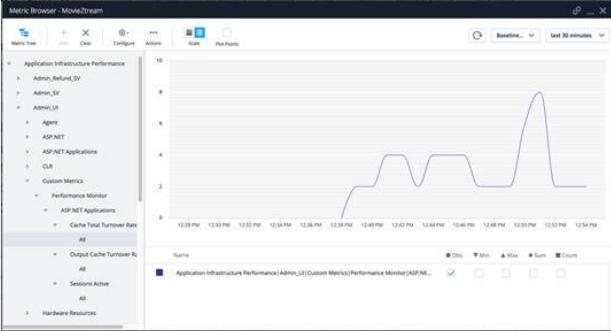
Windows Performance Counters

Customer story

DB Connection Unavailable causing drop in transactions.

New Windows Performance Counter metrics helped to understand Connection Pool state.

Pool sizing issues detected.



The screenshot shows the Metric Browser interface with a line graph titled 'Cache Total Turnover Rate'. The Y-axis ranges from 0 to 10, and the X-axis shows time from 12:28 PM to 12:54 PM. The graph shows a baseline around 2, a sharp rise to 5 at 12:38 PM, a peak of 7 at 12:40 PM, another rise to 5 at 12:42 PM, and then a dip back to 2. This visualizes the 'Pool sizing issues detected' mentioned in the story.

A customer was experiencing occasional timeouts of .NET transactions, which they narrowed down to be Database Connection Unavailable from looking in Snapshots.

It wasn't clear why this was happening, so they realized that they needed further information about what was happening with the Database Connection pool in IIS.

They configured new Windows Performance Counter metrics to start to see the size of the pool as well as the number of spare connections, and soon realised that the pool needed tuning - problem resolved!

Review Question - Answer

JMX Metrics, or Win Perf Counter Metrics, are recorded and sent to the Controller every 10 seconds.

True

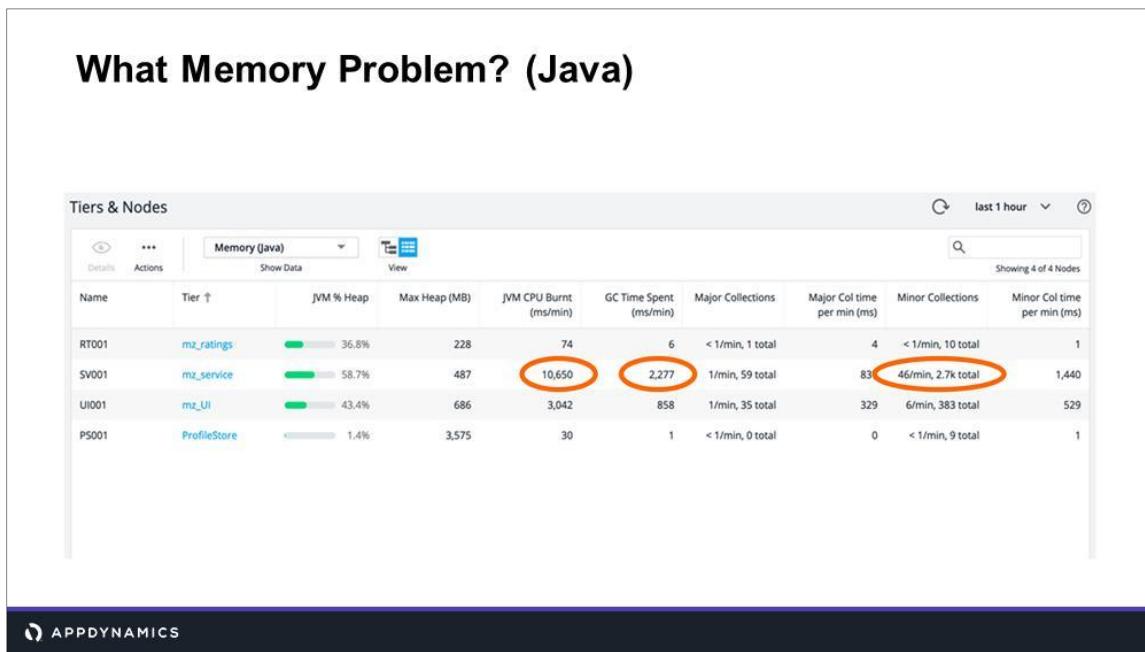
False



False - the agent reads and then sends JMX or Windows Performance Counter metrics to the Controller once every 60 seconds - this is not configurable.

Using Memory Management Tools

What Memory Problem? (Java)



Name	Tier	JVM % Heap	Max Heap (MB)	JVM CPU Burnt (ms/min)	GC Time Spent (ms/min)	Major Collections	Major Col time per min (ms)	Minor Collections	Minor Col time per min (ms)
RT001	mz_ratings	36.8%	228	74	6	< 1/min, 1 total	4	< 1/min, 10 total	1
SV001	mz_service	58.3%	487	10,650	2,277	1/min, 59 total	83	46/min, 2.7k total	1,440
UI001	mz_UI	43.4%	686	3,042	858	1/min, 35 total	329	6/min, 383 total	529
PS001	ProfileStore	1.4%	3,575	30	1	< 1/min, 0 total	0	< 1/min, 9 total	1

It is not necessarily easy to spot when you have a memory problem with your Runtime, in either Java or .NET.

But AppDynamics presents some key metrics on the Nodes & Tiers summary page, that allow you to get a general understanding of whether or not there may be an issue.

A Node that is consuming a rather high number of CPU ms per minute, or spending some time to do Garbage Collection, or the number of GCs looks to be high - all these are indicators of a possible problem that could warrant more in-depth investigation.

What Memory Problem? (.NET)

The screenshot shows a table with the following data:

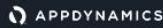
Name	Tier	Current Heap Utilization (MB)	Average Heap Utilization (MB)	Committed Heap (MB)	CPU Burnt (% proc. time)	Induced Collections	Time Spent On Collections (%)
WIN-OC30P1096EK-Adm...	Admin_Refund_SV	88.5	85.6	205.6	2	0	0
WIN-OC30P1096EK-Adm...	Admin_SV	98.5	95.6	505.6	92	3	96
WIN-OC30P1096EK-Adm...	Admin_UI	88.5	85.6	205.6	2	0	0

Likewise, with .NET a Node with high Heap Utilization or CPU Burn, or one that is spending a lot of time doing Garbage Collection may warrant more in-depth investigation.

JVM GC Metrics (Java)

These metrics are available in the Metric Browser, and support both the CMS Actually Parallel/Throughput default collector and the G1 (garbage first) collector that is the default as of Java 8:

- Object Allocation Rate
- Object Promotion Rate
- Live Data Size
- Object Free Rate



***CMS = 'Concurrent-Mark-Sweep'

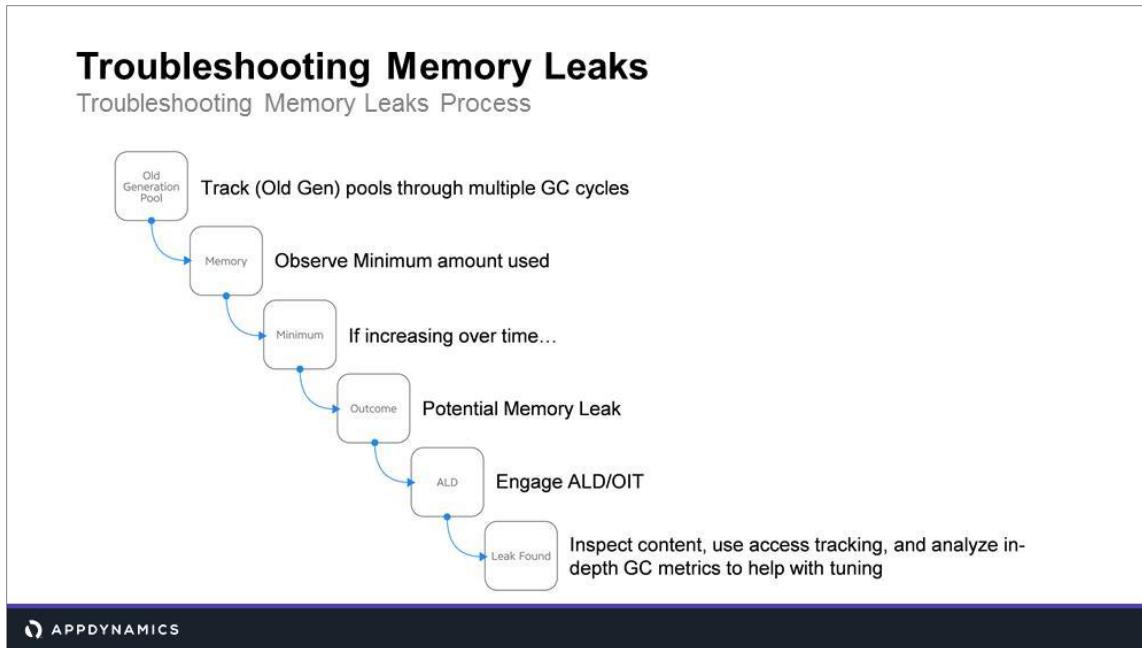
Object Allocation Rate Tracks the megabytes created per minute. Object creation almost always happens in the Eden space. This metric is important for tuning Young Gen size, as a very high allocation rate means Eden space fills up quicker, which leads to more frequent minor garbage collection (a Stop-The-World event). By understanding the relationship between the allocation rate and two other metrics: free rate and promotion rate, the young heap can be sized to fill up less frequently.

Object Promotion Rate Tracks the megabytes promoted from young to old gen per minute. This metric is important for tuning both young and old size, as a very high promotion rate means Eden (or Survivor) space may need to be increased, and leads to quicker filling of the old gen, which leads to more frequent major garbage collection (a Stop-The-World event if using default Parallel/Throughput collector memory collection and might for the others).

Live Data Size Tracks the memory footprint of the application. This metric is useful for understanding the effect of application changes or changes in load characteristics on the amount of memory the application needs to work efficiently. Any change that results in a higher memory footprint needs to be analyzed – having this metric allows analysis of memory footprint regression

Object Free Rate Tracks the megabytes freed from young gen per minute. This metric is important for tuning the size of the young generation and can give some clues to

possible coding issues. It can be used to analyze the effect of young gen tuning, and can be compared to Object Allocation Rate to anticipate and detect memory leaks.



If you track the long-lived (old generation) memory pools over multiple major garbage collection cycles, and observe that the minimum amount of memory used (the valleys in the graph) is steadily increasing over time, the application may be suffering from a memory leak.

The Java Agent offers an automatic leak detection feature to help identify the source of the memory leak. This feature tracks collections over time to identify growing collections. The minimum collection age and size is configurable. If a potential memory leak is found, you can inspect the collection content or use access tracking to determine which code paths are accessing the growing collection.

Follow this process for identifying memory issues.

1. In Grid View, in the **Tiers & Nodes List > Memory (Java) | Memory (.NET) dropdown**
2. The Virtual Machines running in your environment are displayed.
3. Sort the nodes by GC Time Spent in descending order.
4. Identify where garbage collection is taking too long and possibly hindering app performance.

Object Instance Tracking (OIT)

JAVA and .NET

- Use **Object Instance Tracking > Identify Business Transactions** and code paths responsible for excessive object allocations
- Tracking Tab
 - Shows the allocation trend for:
 - Top Application Classes
 - Top System Classes
 - Custom Classes
- Allows a drill-down into how objects are being instantiated and which part of your application code is creating each object type
- Useful for diagnosing memory leak / memory thrash situations



OIT tracks objects that are created, but never released. Thus helps pinpoint a type of memory leak.

Once the object types and memory allocation patterns have been identified, you can then start an allocation tracking session to identify the Business Transactions and code paths that are responsible for the excessive object allocations.

Object Instance Tracking (OIT)

Java

- Enable Object Instance Tracking to track top classes in the Heap by instance count
 - Tracks Top 20 Application classes
 - Specific classes to track can be configured
- OIT increases the amount of information captured by the agents,
 - Results in additional overhead
 - Best to use only while troubleshooting potential memory leaks

Class	Current Instance Count	Shallow Size (bytes)	Status	Instance Count Trend
com.eceleria.appdynamics.mou...	149,171	5,966,840		
com.sun.org.apache.xerces.int...	81,856	3,274,240		
org.apache.xmlbeans.AttributeImpl	55,978	1,223,472		
org.apache.velocity.runtime.par...	9,872	318,104		
com.sun.org.apache.xml.interna...	9,267	446,736		
org.apache.click.util.HtmlString	6,154	147,696		
com.sun.org.apache.xerces.int...	5,883	94,129		

Type	Summary	Time
Memory	Instance Tracking is [enabled]	09/15/19 7:08:25 PM

For java we capture this 1x per minute.

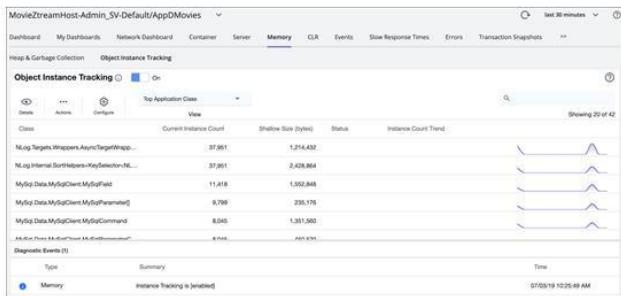
Object Instance Tracking (OIT)

.NET

Enable Object Instance Tracking to track top classes in the Heap by instance count.

Tracks Top 20 .NET framework and Top 20 App classes.

Specific classes to track can be configured.



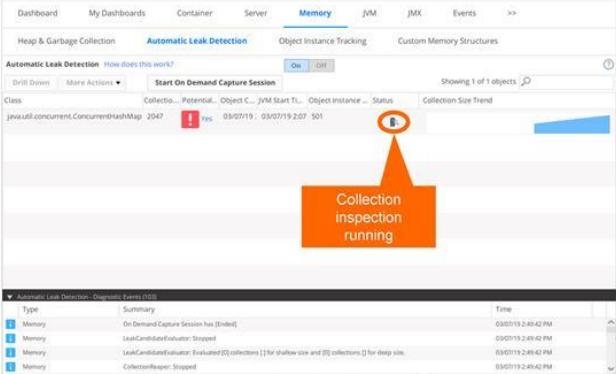
APPDYNAMICS

For .Net, we capture this once every 10 minutes.

Automatic Leak Detection (ALD)

JAVA only

- Monitors common collection classes in your application.
- If the collections grow over time beyond normal fluctuations, a memory leak may exist.
- Useful for detecting collection-based memory leaks.
- Can cause significant overhead:
 - Best used in dev | test
 - If used in production, enable on a single node rather than the whole tier



Type	Summary	Time
Memory	On Demand Capture Session has [Ended]	03/07/19 2:49:42 PM
Memory	LeakCandidateEvolution: Stopped	03/07/19 2:49:42 PM
Memory	LeakCandidateEvolution: Evaluated [0] collections [1] for shallow size and [0] collections [0] for deep size	03/07/19 2:49:42 PM
Memory	CollectorReaper: Stopped	03/07/19 2:49:42 PM

After enabling ALD in the page, you also have to start an On Demand Capture Session, specifying the Session Duration and Minimum Collection Age, as ALD is not something that you keep running indefinitely, it is intended to be used for a relatively short period of time, in no small part due to the overhead it has on the JVM in question.

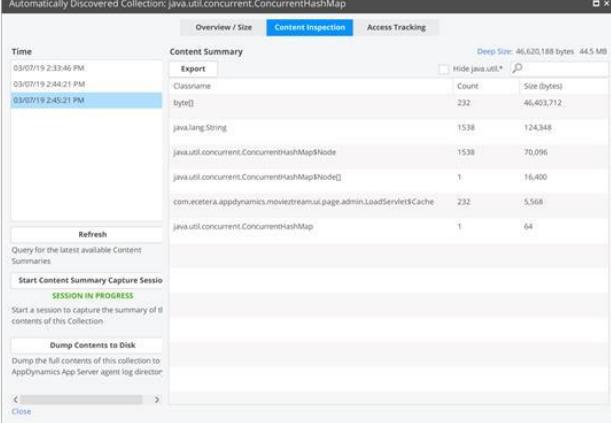
ALD Content Inspection

JAVA only

Drill down on suspect collection to run Content Inspection Session.

Interim results show:

- Class types
- Instance Count
- Size (bytes)



Content Summary	Count	Size (bytes)
by[]	232	46,403,712
java.lang.String	1538	124,348
java.util.concurrent.ConcurrentHashMap\$Node	1538	70,096
java.util.concurrent.ConcurrentHashMap\$Node[]	1	16,400
com.eclipsesource.appdynamics.nativestream.ui.page.admin.LoadServlet\$Cache	232	5,568
java.util.concurrent.ConcurrentHashMap	1	64

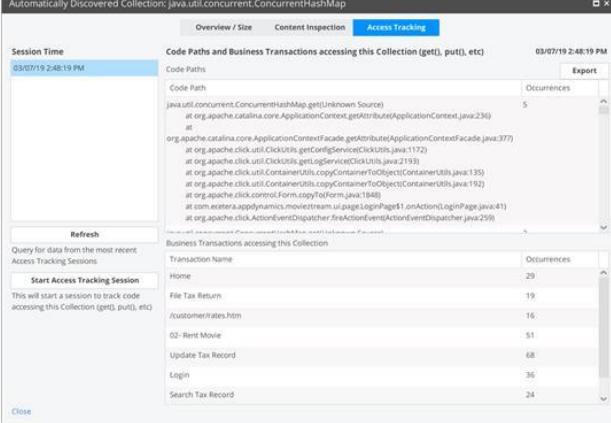
ALD Access Tracking

JAVA only

Drill down on suspect collection to run Access Tracking Session.

Results show:

- Code Paths
- Occurrences
- BTs accessing the collection



Automatically Discovered Collection: java.util.concurrent.ConcurrentHashMap

Session Time: 03/07/19 2:48:19 PM Last Update: 03/07/19 2:48:19 PM

Code Paths and Business Transactions accessing this Collection (get(), put(), etc)

Code Path	Occurrences
java.util.concurrent.ConcurrentHashMap.get(Unknown Source)	5
at org.apache.catalina.core.ApplicationContext.getAttribute(ApplicationContextFacade.java:377)	
org.apache.catalina.core.ApplicationContextFacade.getAttribute(ApplicationContextFacade.java:377)	
at org.apache.click.util.ClickUtils.getConfigServiceClickUtils.java:1172)	
at org.apache.click.util.ClickUtils.getLogServiceClickUtils.java:2193)	
at org.apache.click.util.ContainerUtils.copyContainerObjectToContainerObject(ContainerUtils.java:135)	
at org.apache.click.util.ContainerUtils.copyContainerObject(ContainerUtils.java:192)	
at org.apache.click.util.ContainerUtils.copyContainerObject(ContainerUtils.java:148)	
at com.everera.appdynamics.moviestream.ui.page.LoginPage\$1.onAction(LoginPage.java:41)	
at org.apache.click.ActionListenerEventDispatcher.fireActionEventActionEventDispatcher.java:259)	

Business Transactions accessing this Collection

Transaction Name	Occurrences
Home	29
File Tax Return	19
/customer/rates.htm	16
02- Rent Movie	51
Update Tax Record	68
Login	36
Search Tax Record	24

Start Access Tracking Session

This will start a session to track code accessing this Collection (get(), put(), etc)

Close

Review Question - Answer

Which of the following are true of Automatic Leak Detection (ALD)?

- A) It is useful for detecting collection-based memory leaks.
- B) There is no significant resource overhead with use of ALD.
- C) Allows you to see collections that grow over time.

What you learned

- The purpose and applications for Information Points
- How to configure Information Points to collect Custom Metric data
- The purpose and applications for Data Collectors
- How to configure Data Collectors to add information to Snapshots
- How to enable Development Level Monitoring to collect full call graph snapshots
- How to explain when and when not to enable Development Level Monitoring
- How to configure JMX Metrics to monitor Java Applications
- How to use Windows Performance Counters to collect and display information
- How to identify and troubleshoot memory management issues
- How to use Automatic Leak Detection to find collection-based memory leaks
- How to use Object Instance Tracking to find object-based memory Leaks

Differentiate yourself and your company with AppDynamics Certification

APPDYNAMICS | Certification Program

General Certification Information

<https://learn.appdynamics.com/certifications>

AppDynamics Certified Associate Performance Analyst

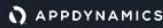
<https://learn.appdynamics.com/certifications/performance-analyst>

AppDynamics Certified Associate Administrator

<https://learn.appdynamics.com/certifications/administrator>

AppDynamics Certified Implementation Professional

<https://learn.appdynamics.com/certifications/implmenter>



To differentiate yourself and your company in an increasingly competitive market, please consider becoming AppDynamics certified.

For more information, please copy and paste this URL into a separate tab in your browser: <https://learn.appdynamics.com/certifications>

For information on specific certification tracks, please copy and paste the appropriate URLs below:

AppDynamics Certified Associate Performance Analyst:

<https://learn.appdynamics.com/certifications/performance-analyst>

AppDynamics Certified Associate Administrator:

<https://learn.appdynamics.com/certifications/administrator>

AppDynamics Certified Implementation Professional:

<https://learn.appdynamics.com/certifications/implmenter>

