APPDYNAMICS

AppDynamics Essentials

AppDynamics Pro Documentation

Version 3.8.x

1. AppDynamics Essentials 3
1.1 Features Overview 3
1.2 Architecture 8
1.3 Logical Model 10
1.3.1 Hierarchical Configuration Model 12
1.3.2 Mapping Application Services to the AppDynamics Model 13
1.4 Getting Started 16
1.4.1 Get Started with AppDynamics SaaS 17
1.4.1.1 Use a SaaS Controller 20
1.4.1.2 SaaS Availability and Security 22
1.4.2 Get Started with AppDynamics On-Premise 23
1.4.3 Download AppDynamics Software 26
1.4.4 Quick Start for DevOps 29
1.4.5 Quick Start for Architects 30
1.4.6 Quick Start for Administrators 31
1.4.7 Quick Start for Operators 31
1.4.8 Set User Preferences 32
1.5 Use AppDynamics 34
1.6 Configure AppDynamics 34
1.7 Glossary 35
1.8 AppDynamics Support 44
1.8.1 Use the Documentation Wiki 44
1.8.2 Controller Dump Files 46
1.8.3 Controller Log Files for Troubleshooting 48
1.8.4 Download Doc PDFs 48
1.8.5 License Information 49
1.9 Documentation Map 50
1.10 AppDynamics Lite 53

AppDynamics Essentials

Expert Advice

Proactive Monitoring and Alerting with AppDynamics

by Ian Withrow

AppDynamics provides application performance management for modern application architectures. Designed for distributed SOA environments, AppDynamics helps you to manage service levels, reduce mean-time-to-repair for problems, plan for application efficiency, and automate typical life-cycle actions for distributed applications.

Basics

Features Overview
AppDynamics in Action Videos
Architecture
Logical Model
Glossary

How AppDynamics Works, from AppDynamics, 4:09 minutes

Getting Started

Get Started with AppDynamics SaaS
Get Started with AppDynamics On-Premise
Download AppDynamics Software
Set User Preferences

Quick Start for Operators
Quick Start for DevOps
Quick Start for Architects
Quick Start for Administrators

Support

Release Notes
Supported Environments

AppDynamics Support
Documentation Map

Features Overview

This topic describes high-level benefits and features of AppDynamics Pro.

Expert Advice

How monitoring analytics can make DevOps more agile

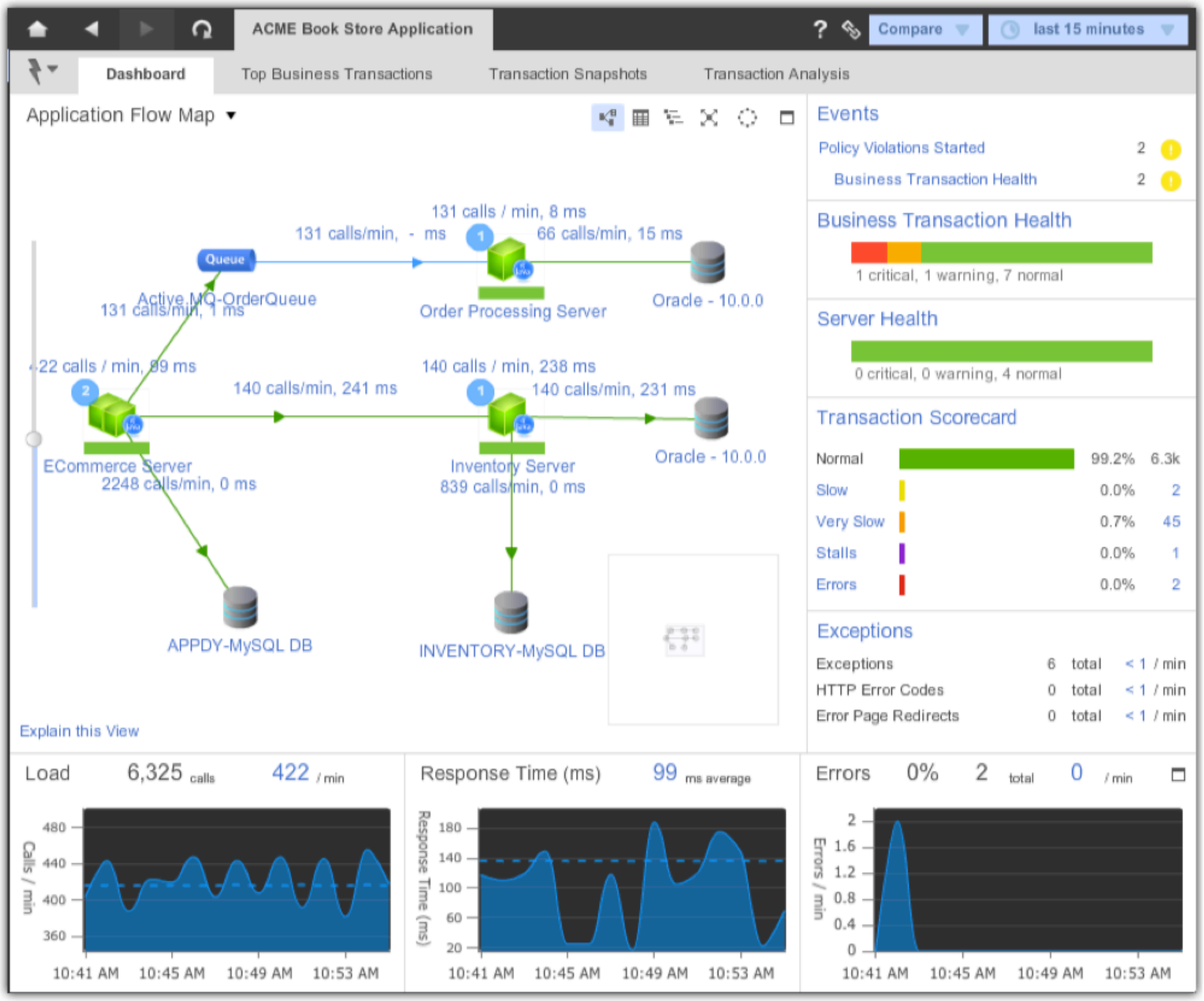
by Sandy Mappic

- Continuous Discovery, Visibility, and Problem Detection
- Real-Time Business Transaction Monitoring
- End User Monitoring
- Service Endpoint Monitoring
- Hardware and Server Monitoring
- Health Rules, Policies, and Actions
- Troubleshooting and Diagnostics
- Systems Integration
- Learn More

Continuous Discovery, Visibility, and Problem Detection

AppDynamics continuously discovers and monitors all processing in your application environment using advanced tag, trace, and learn technology across your distributed transactions. With this information, AppDynamics provides a simple intuitive view of live application traffic and you can see where bottlenecks exist.

Dashboards show the health of your entire business application. Health indicators are based on configurable thresholds and they update based on live traffic. When new services are added to the system AppDynamics discovers them and adds them to the dashboards and flow maps. See Visua lize App Performance.



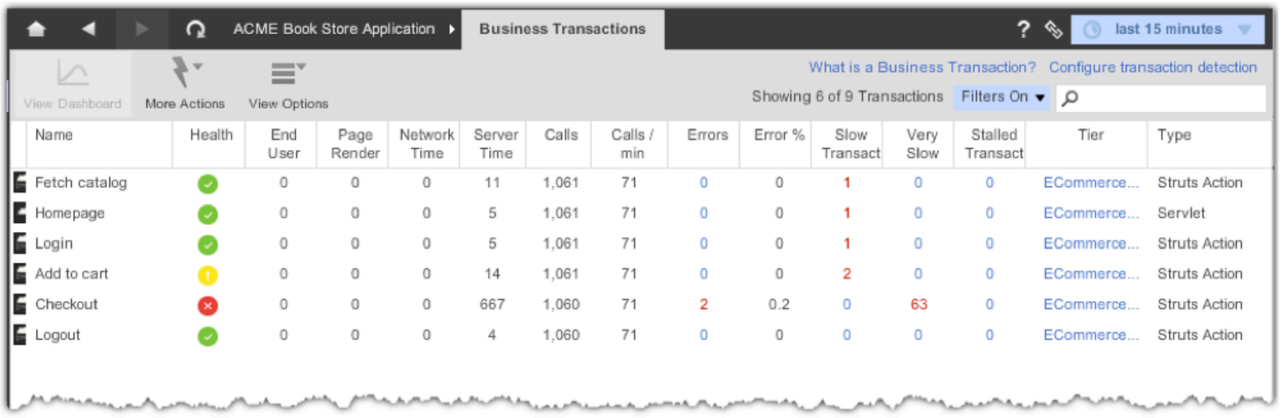
Transaction Scorecard		
Normal	99.2%	6.3k
Slow	0.0%	2
Very Slow	0.7%	45
Stalls	0.0%	1
Errors	0.0%	2

Exceptions			
Exceptions	6	total	< 1 / min
HTTP Error Codes	0	total	< 1 / min
Error Page Redirects	0	total	< 1 / min

AppDynamics observes normal performance patterns so that it knows when application performance becomes abnormal. It automatically identifies metrics whose current values are out of the normal range, based on dynamic baselines it has observed for these metrics. See Behavior Learning and Anomaly Detection.

Real-Time Business Transaction Monitoring

An AppDynamics business transaction represents a distinct logical user activity such as logging in, searching for items, buying an item, etc. Organizing application traffic into business transactions aligns the traffic with the primary functions of a web business. This approach focuses on how your users are experiencing the site and provides real-time performance monitoring.


ACME Book Store Application
Business Transactions
last 15 minutes
View Dashboard
More Actions
View Options
What is a Business Transaction?
Configure transaction detection
Filters On

Showing 6 of 9 Transactions

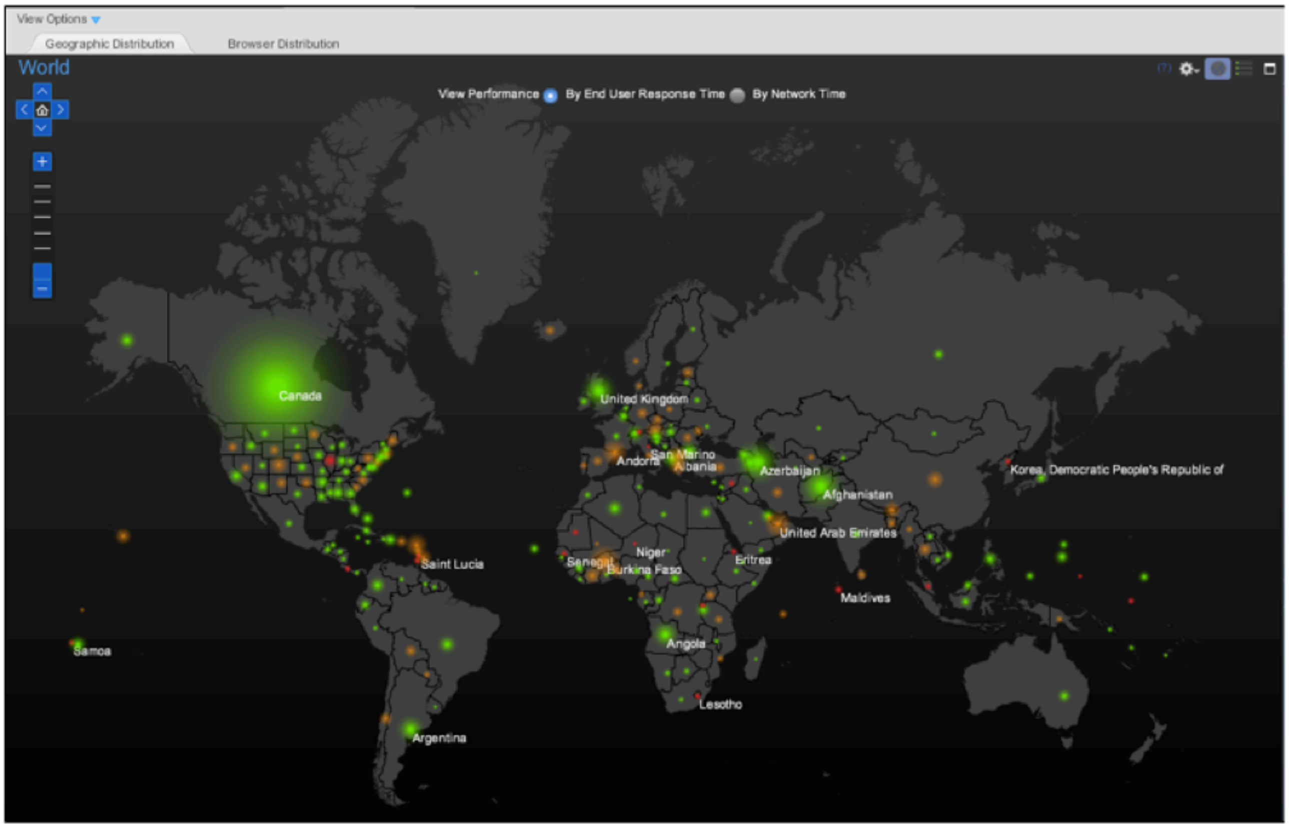
Name	Health	End User	Page Render	Network Time	Server Time	Calls	Calls / min	Errors	Error %	Slow Transact	Very Slow	Stalled Transact	Tier	Type
Fetch catalog	✓	0	0	0	11	1,061	71	0	0	1	0	0	ECommerce...	Struts Action
Homepage	✓	0	0	0	5	1,061	71	0	0	1	0	0	ECommerce...	Servlet
Login	✓	0	0	0	5	1,061	71	0	0	1	0	0	ECommerce...	Struts Action
Add to cart	!	0	0	0	14	1,061	71	0	0	2	0	0	ECommerce...	Struts Action
Checkout	✕	0	0	0	667	1,060	71	2	0.2	0	63	0	ECommerce...	Struts Action
Logout	✓	0	0	0	4	1,060	71	0	0	0	0	0	ECommerce...	Struts Action

See Business Transaction Monitoring and Background Task Monitoring.

End User Monitoring

End user monitoring (EUM) provides information about your end users' experience starting from the users' web browsers and their native mobile applications. It gives you visibility across geographies and browser types, answering questions such as:

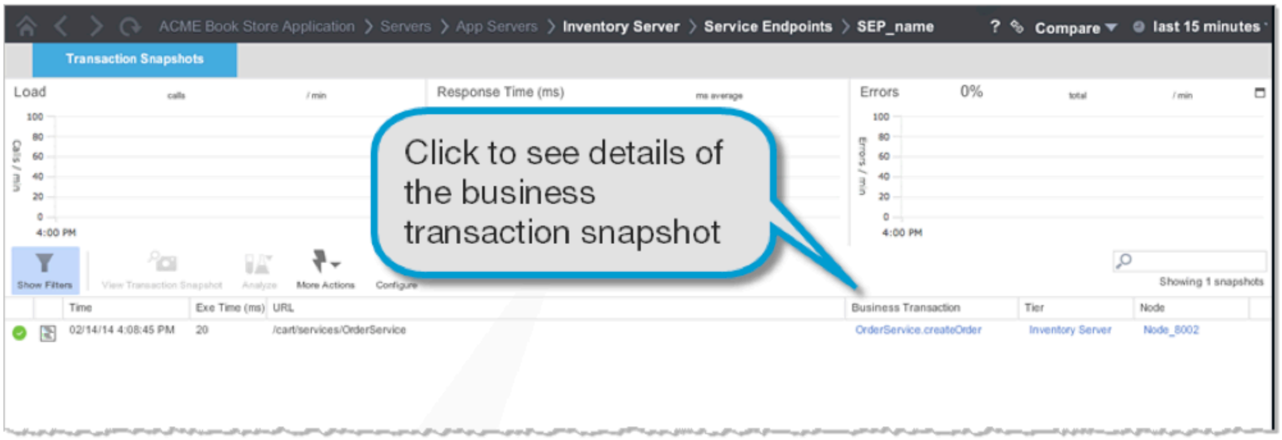
- Where are the heaviest loads?
- Where are the slowest end-user response times?
- How does end user performance vary by Web browser?
- How does end user performance vary by mobile application, carrier, or device?



See AppDynamics End User Experience.

Service Endpoint Monitoring

Service endpoints are helpful in complex, large-scale applications where an owner is assigned to one or more logical tiers and the standard representation does not correspond with real-life ownership of application components. Service endpoints allow you to see a subset of the metrics for the tier so you can focus on the key performance indicators and snapshots of entry points that are truly of interest to you. Service endpoints are similar to business transactions except that they only show metrics for the entry points and do not track metrics for any downstream segments.


ACME Book Store Application > Servers > App Servers > Inventory Server > Service Endpoints > SEP_name
? Compare last 15 minutes
Transaction Snapshots
Load calls / min
Response Time (ms) ms average
Errors 0% total / min
Calls / min
Errors / min
100 80 60 40 20 0
4:00 PM
Click to see details of the business transaction snapshot
Show Filters
View Transaction Snapshot
Analyze
More Actions
Configure
Showing 1 snapshots

Time	Exe Time (ms)	URL	Business Transaction	Tier	Node
02/14/14 4:08:45 PM	20	/cart/services/OrderService	OrderService.createOrder	Inventory Server	Node_8002

See Service Endpoint Monitoring.

Hardware and Server Monitoring

AppDynamics machine agents gather information about the operating systems and machines, such as CPU activity, memory usage, disk reads and writes, etc. AppDynamics agents monitor JVM and CLR metrics including heap usage and collections. See Infrastructure Monitoring.

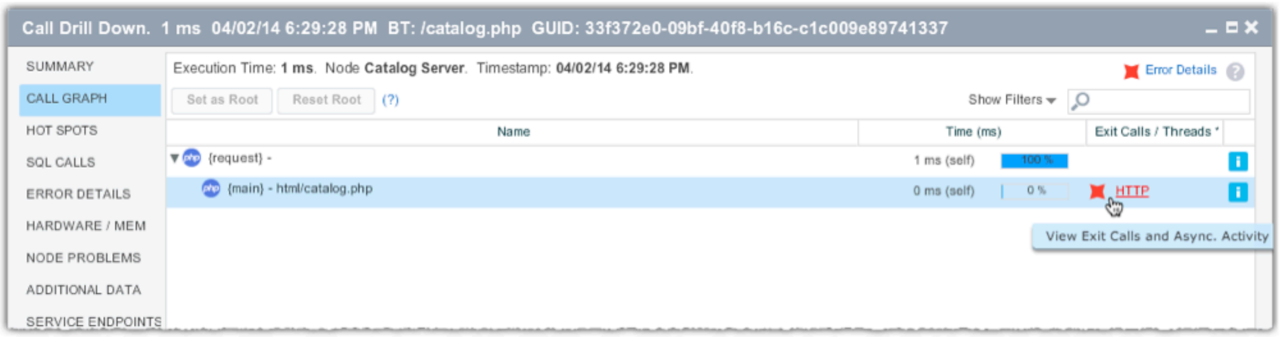
Health Rules, Policies, and Actions

Dynamic baselines combined with policies and health rules help you proactively detect and troubleshoot problems before customers are affected. Health rules define metric conditions to monitor, such as when the "average response time is four times slower than the baseline". AppDynamics supplies default health rules that you can customize, and you can create new ones.

You can configure policies to trigger automatic actions when a health rule is violated or when any event occurs. Actions include sending email, scaling-up capacity in a cloud or virtualized environment, taking a thread dump, or running a local script. See Alert and Respond.

Troubleshooting and Diagnostics

You can examine transaction snapshots for slow and error transactions and drill down into the snapshot with the slowest response time to begin deep diagnostics to discover the root cause of the problem.


Call Drill Down. 1 ms 04/02/14 6:29:28 PM BT: /catalog.php GUID: 33f372e0-09bf-40f8-b16c-c1c009e89741337
SUMMARY
CALL GRAPH
HOT SPOTS
SQL CALLS
ERROR DETAILS
HARDWARE / MEM
NODE PROBLEMS
ADDITIONAL DATA
SERVICE ENDPOINTS
Execution Time: 1 ms. Node Catalog Server. Timestamp: 04/02/14 6:29:28 PM.
Error Details
Set as Root
Reset Root
Show Filters
Name
Time (ms)
Exit Calls / Threads
{request} -
1 ms (self)
100 %
{main} - html/catalog.php
0 ms (self)
0 %
HTTP
View Exit Calls and Async. Activity

See Rapid Troubleshooting.

Systems Integration

AppDynamics is designed to interface with other systems in your organization. You can add data to AppDynamics, retrieve data from AppDynamics, and integrate AppDynamics actions into your alerting system. See AppDynamics Extensions and Integrations.

Learn More

- Product Features and Benefits

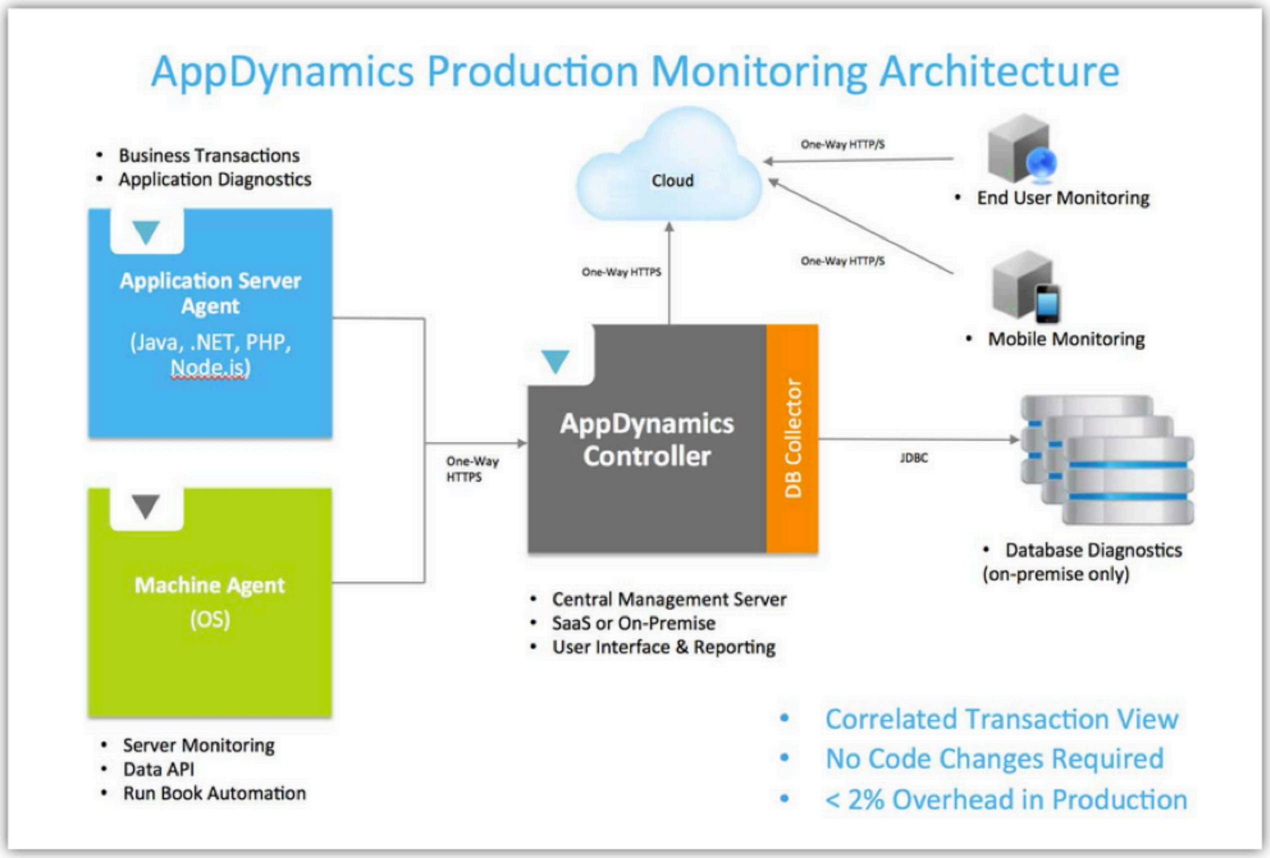
Architecture

- AppDynamics Pro Architecture
  - AppDynamics Controller and UI
  - AppDynamics App Agents
  - AppDynamics Machine Agents
  - AppDynamics Web EUM
  - AppDynamics Mobile APM
  - AppDynamics for Databases
- Learn More

This topic summarizes the components of AppDynamics and how they work together to monitor your application environment.

AppDynamics Pro Architecture

An AppDynamics deployment consists of a Controller (either on-premise or SaaS) and its UI, app agents, and machine agents. Additional components include Web End User Monitoring, Mobile APM, and AppDynamics for Databases.


AppDynamics Production Monitoring Architecture
• Business Transactions
• Application Diagnostics
Application Server Agent
(Java, .NET, PHP, Node.js)
Machine Agent
(OS)
• Server Monitoring
• Data API
• Run Book Automation
Cloud
One-Way HTTP/S
One-Way HTTP/S
One-Way HTTPS
• End User Monitoring
• Mobile Monitoring
One-Way HTTPS
AppDynamics Controller
DB Collector
JDBC
• Database Diagnostics (on-premise only)
• Central Management Server
• SaaS or On-Premise
• User Interface & Reporting
• Correlated Transaction View
• No Code Changes Required
• < 2% Overhead in Production

AppDynamics Controller and UI

The AppDynamics Controller is the central repository and analytics engine where all performance data is stored, baselined, and analyzed. The Controller is specially designed for large-scale production environments, and can scale to manage hundreds to thousands of application servers.

The AppDynamics Controller can be installed on-premise or it can be accessed as software as a service (SaaS). A SaaS Controller is managed at AppDynamics and you connect to it from a web browser using HTTP/HTTPS. An on-premise Controller is managed by you on your server in a data center or in the cloud.

You access performance data interactively using the Controller UI or programmatically using the AppDynamics REST API.

AppDynamics App Agents

AppDynamics app agents are installed on your JVM, .NET, or PHP application. They automatically inject instrumentation in application bytecode at runtime.

Patent-pending Dynamic Flow Mapping™ technology continuously discovers, maps, and tracks all business transactions, services, and backends in your web application architecture 24×7.

Patent-pending Deep-on-Demand Diagnostics™ technology learns code execution behavior for each business transaction. It automatically detects problems and collects deep diagnostics data to troubleshoot them.

AppDynamics Machine Agents

One or more machines (real or virtual) constitute the hardware and operating system on which your application runs. Machines can be instrumented by an AppDynamics machine agent, which collects data about machine performance and sends it to the Controller.

AppDynamics Web EUM

AppDynamics Web End User Experience Monitoring (Web EUM) allows you to see how your web application is performing from the point of view of your end user. You can answer questions like: Which 1st or 3rd party Ajax or iframe calls are slowing down page load time? How does server performance impact end user experience in aggregate or in individual cases? You can drill into the data to explore how users experience your application in their Web browsers.

AppDynamics Mobile APM

Mobile Application Performance Management (Mobile APM) provides visibility into the end-user experience of your mobile users. If you have also instrumented your application servers, you can get end-to-end visibility from the mobile device all the way to multiple tiers on the server-side.

AppDynamics for Databases

AppDynamics Pro along with AppDynamics for Databases gives you end-to-end visibility into the performance of your applications, helping you dramatically reduce the time it takes to find and fix database performance issues.

Learn More

- Logical Model
- AppDynamics Administration

Logical Model

- Business Application
- Tiers
- Nodes
- Learn More

This topic describes the basic elements of the AppDynamics model.

Before deploying AppDynamics, also see Mapping Application Services to the AppDynamics Model.

Business Application

An AppDynamics business application models all components or modules in an application environment that provide a complete set of functionality. Think of it as all the web applications, databases, and services that interact or "talk" to each other or to a shared component. When web applications, databases, and services interact, AppDynamics can correlate their activities to provide useful and interesting performance data.

AppDynamics lets you monitor multiple business applications, though it does not correlate events between them.

Because a single node belongs to a single business application, you can also think of a business application as a kind of namespace for all your nodes. See Nodes.

Business applications contain tiers, and tiers contain nodes.

Tiers

A tier represents a key module in an application environment, such as a website or processing application or a virtual machine. Tiers help you logically organize and manage your business application so that you can scale multiple nodes, partition metrics, define performance thresholds, and respond to anomalies. The metrics from one tier tell a different story than those from another tier; AppDynamics helps you define different policies and processes for each tier. A tier can belong to only one business application.

A tier is composed of one node or a group of nodes. For example, in the Acme sample application the Inventory tier has one node whereas the E-Commerce tier has 2 nodes.

Nodes grouped into a tier may have redundant functionality or may not. An example of a multi-node tier with redundant nodes is when you have a set of clustered application servers or services. An example of a multi-node tier with different nodes is when you have a set of services that do not interact with each other though you want to roll up their performance metrics together.

Keep in mind that an environment can have similar nodes that are used by different applications, so similar nodes should not always belong to the same tier. An example is a complex environment that has two HTTP web servers that serve two separate applications.

Business applications contain tiers. The traffic in a business application flows between tiers. This flow is represented in AppDynamics flow maps along with performance data for the traffic. There is always a tier that is the starting point for a Business Transaction, indicated by a Start label on the flow map.

Nodes

A node is the basic unit of processing that AppDynamics monitors. By definition a node is instrumented by an AppDynamics agent, either an app agent or machine agent or both. App agents are installed on:

- JVMs
- Windows .NET applications (IIS, executables, or services)
- PHP Runtime Instances
- Node.js processes

Machine agents are installed on virtual or physical machine operating systems.

Nodes belong to tiers. An app agent node cannot belong to more than one tier. A machine agent cannot belong to more than one tier; however you can install more than one machine agent on a machine.

Learn More

- Mapping Application Services to the AppDynamics Model

- Name Business Applications, Tiers, and Nodes
- Features Overview
- Glossary

Hierarchical Configuration Model

- Entry Point and Exit Point Inheritance
- Node Inheritance
- Switching Configuration Levels
- Learn More

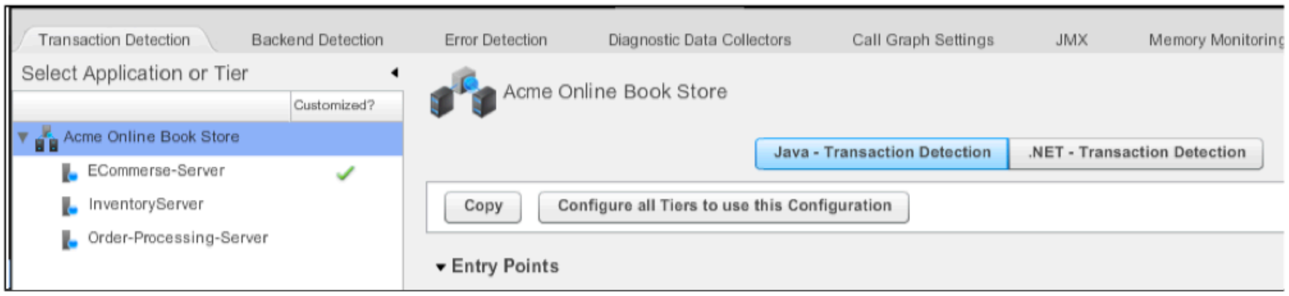
Transaction detection (entry point), backend detection (exit point), and node property configurations are applied on a hierarchical inheritance model. This model provides a default configuration for new tiers as well as the ability to re-use custom configurations in all tiers or tiers that you specify, eliminating the need to configure custom entry and exit points for all tiers.

A tier can inherit all its transaction detection and backend detection configuration from the application, or it can override the application configuration to use a custom configuration.

Similarly, a node can inherit its entire node property configuration from its parent, or it can override the parent configuration to use a custom configuration.

Entry Point and Exit Point Inheritance

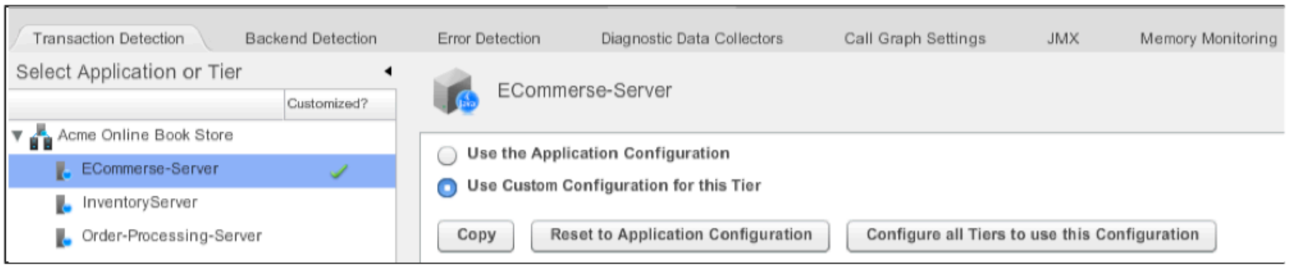
By default, tiers inherit the entry point and exit point configurations of the application. You can copy the application-level configuration to specific tiers or explicitly configure all tiers to use the application-level configuration.



At the tier level, you can specify that the tier should use the application-level configuration.

Or you can an override the application-level configuration by creating a custom configuration for the specific tier.

You can configure all tiers to use the custom configuration or copy the configuration for re-use in specific tiers. You can also reset a tier that is currently using a custom configuration to use the application-level configuration.

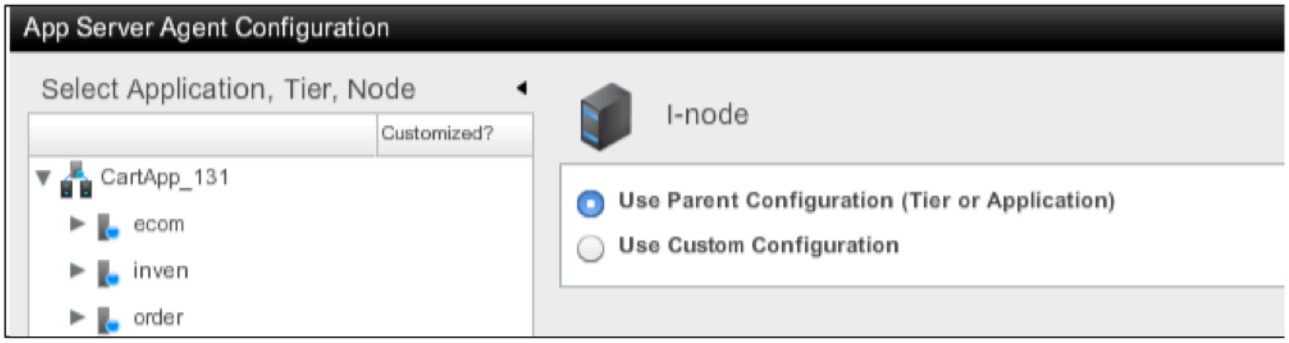


Transaction Detection
Backend Detection
Error Detection
Diagnostic Data Collectors
Call Graph Settings
JMX
Memory Monitoring
Select Application or Tier
Customized?
Acme Online Book Store
ECommerse-Server
InventoryServer
Order-Processing-Server
ECommerse-Server
Use the Application Configuration
Use Custom Configuration for this Tier
Copy
Reset to Application Configuration
Configure all Tiers to use this Configuration

Node Inheritance

By default a node inherits the node properties of its parent tier (or of the application).

When you configure node properties you can specify that all nodes in a tier inherit the node properties of the parent (tier or application) or that the node should use a custom configuration.


App Server Agent Configuration
Select Application, Tier, Node
Customized?
CartApp_131
ecom
inven
order
I-node

● Use Parent Configuration (Tier or Application)
○ Use Custom Configuration

If you create a custom configuration for a node, you can copy that configuration to the application, tier or to another node.

Switching Configuration Levels

If you customize configuration at the tier or node level and then switch back to the application-level configuration, you will not see the old configuration in the UI. However, the old tier or node level configuration is stored, and if you will see these old settings if you switch to the lower-level configuration again.

Learn More

- Configure Backend Detection (Java)
- Configure Backend Detection (.NET)
- Configure Business Transaction Detection
- App Agent Node Properties

Mapping Application Services to the AppDynamics Model

- Your Application and the AppDynamics Model
- How AppDynamics Represents Your Application
    - AppDynamics Business Applications
    - AppDynamics Tiers
- How to Map
- Learn More

AppDynamics and your team may use different terminology to describe your application environment. This topic discusses how to map the services in your application environment to the AppDynamics model, which uses the terms "business applications", "tiers", and "nodes". For an overview of these terms see Logical Model.

Your Application and the AppDynamics Model

Your distributed application environment most likely consists of various services, including:

- Web applications served from an application server (JVM, IIS, PHP Web server, etc.)

- Databases or other data stores
- Remote services such as message queues and caches

AppDynamics maps your application environment into a hierarchical system of business applications, tiers, nodes and backends.

The node represents the actual application server that is instrumented by an AppDynamics app agent.

Business applications and tiers are logical constructs used to represent your environment in the AppDynamics model.

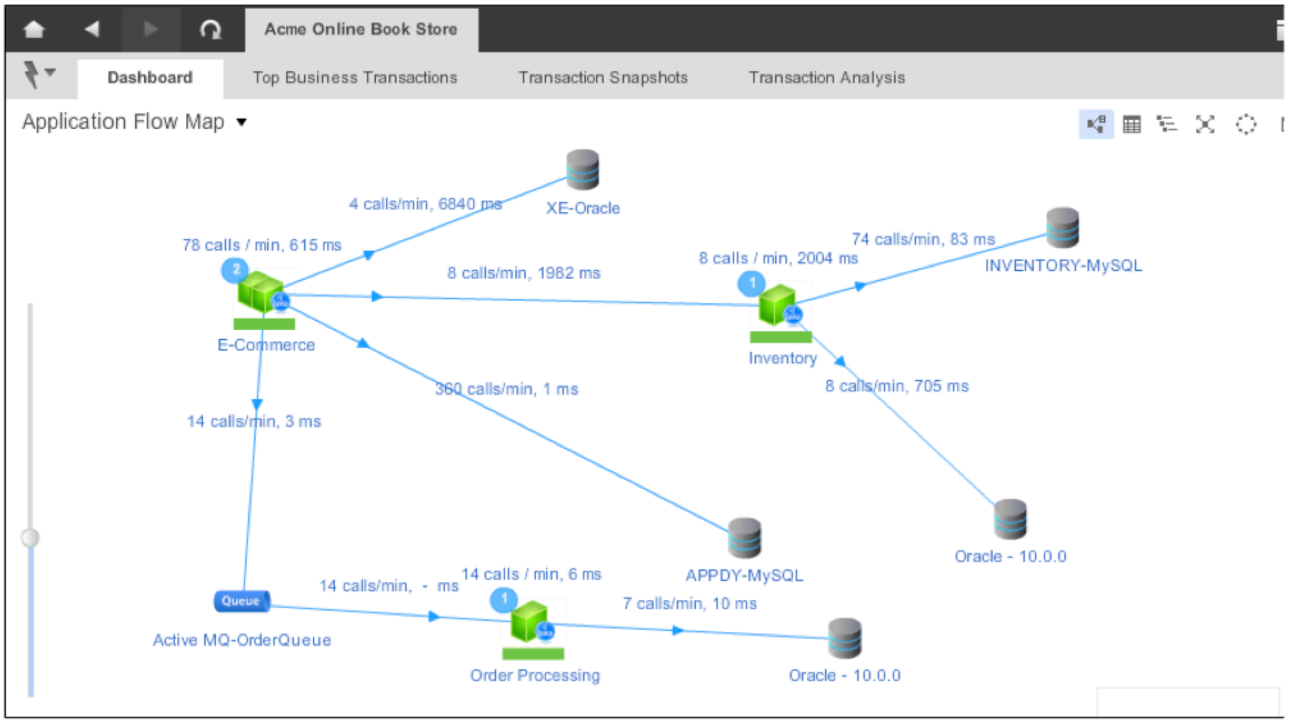
Business applications contain tiers and tiers contain nodes. A node cannot belong to more than one tier, and a tier cannot belong to more than one business application.

A backend is a component that is not instrumented by an AppDynamics app agent, but the model allows you to monitor the flows from the instrumented nodes to the backends. These flows often reveal the root cause of a problem that is first identified on an instrumented node.

How AppDynamics Represents Your Application

The flowmap below describes a single business application for the Acme Online Book Store.

- E-Commerce, Order Processing and Inventory are the tiers.
- The boxes inside the tiers represent instrumented nodes. The E-Commerce tier has two nodes, the Order Processing and Inventory tiers each has one node.
- The database backends are XE-Oracle, Inventory-MySQL, APPDY-MySQL and two Oracle 10.0.0s.
- The message queue backend is Active MQ--Order Queue.
- The blue lines represent the flow of traffic through the entire application.



Because the services provided by the E-Commerce, Inventory, and Order Processing interacting tiers are all modeled as part of the same business application, it is possible, for example, for AppDynamics to trace the root cause of poor performance of a node in the front-end E-Commerce tier to a slow SQL call from the downstream Inventory tier to the INVENTORY-MySQL database. Without this correlation among the services, this information would not be available.

AppDynamics Business Applications

You can use a single AppDynamics business application to model all of the application environment's services that provide a complete set of functionality. Think of the business application as all the services that interact to support the application's mission. When these services (web applications, databases, remote services, etc.) interact, they are modeled as part of the same business application, and AppDynamics can correlate performance metrics among them to provide a complete picture of the application's performance. A complete picture helps you identify the root cause of any problems that are detected. If any of the services upon which the application depends are missing from the model, you may miss information about a component that is causing problems to appear in a different component. AppDynamics cannot provide correlation between separate business applications.

For example, a single shopping business application may be composed of an inventory application, a e-commerce front-end application, and databases. The inventory application and e-commerce front-end application could be modeled as tiers in a single AppDynamics "business application".

On the other hand, if you do not care about correlation among these services and instead want to maintain separate access control to the various components, you could model the services as separate business applications.

It is also appropriate to have multiple business applications for sets of services that do not interact with each other. A typical example of using multiple business applications is when you have separate staging, testing, and production environments for the same website. In this case the three business applications are essentially copies of each other.

AppDynamics Tiers

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. These services may be thought of as "applications" in your application environment, but if they interact with one another AppDynamics usually models them as tiers in the same "business application".

A tier can be composed of one node or multiple redundant nodes. One example of a multi-node tier is a set of clustered application servers or services.

There is no interaction among nodes within a single tier. Interaction occurs between tiers in a business application, as illustrated in the flowmap.

How to Map

The mapping of tiers to business applications and of nodes to tiers occurs in the configuration of the app agent, either in the options to the app agent startup script or in the controller-info-xml file.

For example, in an all-Java environment, to map Node_8000 and Node_8003 to the E-Commerce tier in the AcmeOnLine business application, the startup options for Node_8000 would be

```
-Dappdynamics.agent.applicationName=AcmeOnLine
-Dappdynamics.agent.tierName=E-Commerce
-Dappdynamics.agent.nodeName=Node_8000
```

and for Node_8003

```
-Dappdynamics.agent.applicationName=AcmeOnLine
-Dappdynamics.agent.tierName=E-Commerce
-Dappdynamics.agent.nodeName=Node_8003
```

To map Node_8002 in the Inventory tier in the same business application, the configuration would be

```
-Dappdynamics.agent.applicationName=AcmeOnLine
-Dappdynamics.agent.tierName=Inventory
-Dappdynamics.agent.nodeName=Node_8002
```

and to map Node_8001 in the Order Processing tier in the same application

```
-Dappdynamics.agent.applicationName=AcmeOnLine
-Dappdynamics.agent.tierName=Order Processing
-Dappdynamics.agent.nodeName=Node_8001
```

Details vary depending on the platform of the agent. See the installation and configuration properties documentation for the particular app agent that you are configuring.

Learn More

- Logical Model
- Name Business Applications, Tiers, and Nodes
- App Agent for Java Configuration Properties
- App Agent for .NET Configuration Properties
- App Agent for PHP Proxy Configuration Properties
- Install the App Agent for PHP
- Install the App Agent for Node.js

Getting Started

- Initial Installation
    - Self-Service Trial or Standard?
    - On-premise or SaaS?
        - Get Started with AppDynamics SaaS
        - Get Started With AppDynamics On-Premise
- Monitoring, Troubleshooting, and Analyzing Application Performance

This section gives you a roadmap to using AppDynamics.

Initial Installation

Self-Service Trial or Standard?

If you are using the self-service trial see Quick Install.

If you are using a standard installation see Install and Upgrade AppDynamics.

On-premise or SaaS?

To get started with installing, configuring, and using AppDynamics, first determine whether you will use an on-premise or SaaS Controller.

For information about the different approaches see:

- SaaS Availability and Security
- Differences when using a SaaS Controller

Get Started with AppDynamics SaaS

If you are using or going to use the AppDynamics SaaS Controller, see Get Started with AppDynamics SaaS.

Get Started With AppDynamics On-Premise

If you are going to host your own Controller on premise, see Get Started With AppDynamics On-Premise.

Monitoring, Troubleshooting, and Analyzing Application Performance

To get started using AppDynamics after it is installed see:

- AppDynamics Essentials
- Quick Tour of the User Interface Video Tutorial

Get Started with AppDynamics SaaS

Follow these steps to get started with AppDynamics.

If you are reading a PDF of this document, use your Help Center login to access the documentation at http://docs.appdynamics.com.

- Get Your SaaS Account Information from AppDynamics
- Design Your AppDynamics Deployment
- Download and Install the AppDynamics App Agents
- Download and Install the AppDynamics Web and Mobile Agents
- SaaS Login Credentials

Expert Advice

Deploying APM in the Enterprise... the Path of the Rock Star
By Jim Hirschauer

- Connecting Agents to Your SaaS Controller Service
- Access the AppDynamics UI from a Browser
- Review the Dashboards and Flow Maps
- Review Defaults and Configure Business Transactions, if Needed
- Review Defaults and Configure Client-Side Monitoring, if Needed
- Review Defaults and Configure Databases and Remote Services, if Needed
- Review Default Heath Rules and Set Up Policies
- Review Default Error Detection
- Explore Additional Data and Metric Features
- Configure Advanced Features
- Start Monitoring and Troubleshooting
- Questions?

Get Your SaaS Account Information from AppDynamics

After signing up for AppDynamics SaaS, you receive a Welcome email containing important account information, including the Account Owner login. Save this information.

Design Your AppDynamics Deployment

- Learn about Business Transaction Monitoring and identify which critical business transactions you want to monitor.
- Learn about AppDynamics End User Experience and decide whether you want to use this feature.
- Learn about how to map your application components to the AppDynamics business application, tier, and node model. See Logical Model and Name Business Applications, Tiers, and Nodes.
- Based on the model, plan how you will specify AppDynamics application, tier, and node names during installation.
- Decide whether you want to monitor client-side usage with AppDynamics End User Experience.
- For Java environments, decide whether you want to use object instance tracking.

Download and Install the AppDynamics App Agents

Download the AppDynamics application agents from the Download Center. AppDynamics app agents collect data from your application servers and other monitored systems and report to the Controller. Select the agents that are appropriate for your environment:

- Java Agent
- .NET Agent
- PHP Agent
- Machine Agent
  For details see Download AppDynamics Software.

Follow the instructions to install the AppDynamics App Agents.

Download and Install the AppDynamics Web and Mobile Agents

Install the client-side agents in your mobile applications and web pages. See instructions for mobile and web.

SaaS Login Credentials

SaaS Controller login credentials are included in the welcome email from AppDynamics.

To add additional login accounts contact the AppDynamics Support Team.

The SaaS Controller login is an Account Administrator credential. The Account Administrator can create other users for the account. See Account Administrator.

Connecting Agents to Your SaaS Controller Service

For agents to successfully connect to the Controller, configure the Controller host and port information using either the controller-info.xml file or the system properties of your JVM startup script.

To use HTTPS communication, enable SSL by setting the <controller-ssl-enabled> agent configuration property to "True". For details see App Agent for Java Configuration Properties, App Agent for .NET Configuration Properties, App Agent for PHP Proxy Configuration Properties and Machine Agent Configuration Properties.

- The default ports for the SaaS Controller service are:
  - Port 80 for HTTP
  - Port 443 for HTTPS

If you need to specifically open up the communication ports (80 or 443) for the AppDynamics SaaS Controller IP address please request the IPs from the AppDynamics Support Team.

Access the AppDynamics UI from a Browser

Once you have installed the agents, launch your web browser and connect to the AppDynamics User Interface (UI). For SaaS, the URL includes the account name from the Welcome email:

```
http://<account-name>.saas.appdynamics.com/controller
```

When using SSL, use port 443 or https to access the Controller.

Review the Dashboards and Flow Maps

AppDynamics automatically discovers the Business Transactions in your application environment. Browse the Application Dashboard and see the Flow Maps to visualize your application. You can resize and move icons around on the flow maps.

Review Defaults and Configure Business Transactions, if Needed

The default configurations may need to be further customized for your environment. For example, AppDynamics may have discovered transactions that you want to group together or even exclude, because you want to concentrate on the most important transactions. There may be business transactions that are not yet discovered for which you need to configure detection rules. See:

- Monitor Business Transactions
- Configure Business Transaction Detection

Review Defaults and Configure Client-Side Monitoring, if Needed

You may want to refine the way AppDynamics names pages and mobile requests, for example, if the data for multiple web pages would be better understood under a single name. See:

- Configure Mobile Network Requests
- Set Up and Configure Web EUM

Review Defaults and Configure Databases and Remote Services, if Needed

AppDynamics automatically discovers "backends" such as databases, message queues, etc. by following calls in the Java or .NET code. Look at the databases and remote services dashboards to make sure all necessary backends are revealed. If needed, configure how backends are detected.

Review Default Heath Rules and Set Up Policies

AppDynamics provides default Health Rules that define performance parameters for business transactions, such as the conditions that indicate a slow transaction, or when too much memory is being used. You can adjust the thresholds that define when a health rule is violated, create new health rules, and set up policies to specify actions to automate when health rules are violated.

Review Default Error Detection

AppDynamics detects errors and exceptions. You can review and, if needed, modify the error detection rules. For example, some errors you may want to ignore.

Explore Additional Data and Metric Features

Explore these features to gain more insight into application performance:

- Data Collectors
- Business Metrics
- (for Java environments) JMX Metrics
- Machine Agent Custom Metrics

Configure Advanced Features

Additional features you may want to use include:

- Custom Dashboards
- Automation
- AppDynamics Extensions and Integrations

Start Monitoring and Troubleshooting

Start getting the benefits of AppDynamics! See:

- AppDynamics in Action Videos
- AppDynamics Features

Questions?

For questions about using AppDynamics contact the AppDynamics Support Team.

Use a SaaS Controller

- Your SaaS Controller URL

- Login Credentials
- Connecting Agents to Your SaaS Controller Service
- SMTP Service for SaaS
- Contact Support

If you are using the SaaS service for the AppDynamics Controller, simply open a web browser at the URL of the AppDynamics UI and log in with your AppDynamics credentials.

Your SaaS Controller URL

Your SaaS Controller URL is included in the welcome email from AppDynamics.

The URL is of the following form:

```
http(s)://<customer>.saas.appdynamics.com/controller
```

Login Credentials

Login credentials are included in the welcome email from AppDynamics.

To add additional login accounts contact the AppDynamics Support Team.

Connecting Agents to Your SaaS Controller Service

For agents to successfully connect to the Controller, configure the Controller host and port information using either the controller-info.xml file or the system properties of your JVM startup script.

To use HTTPS communication, enable SSL by setting the <controller-ssl-enabled> agent configuration property to "True". For details see App Agent for Java Configuration Properties, App Agent for .NET Configuration Properties, and Machine Agent Configuration Properties. See also I mplement SSL on SaaS.

- The default ports for the SaaS Controller service are:
  - Port 80 for HTTP
  - Port 443 for HTTPS

⚠ **Important** If you need to specifically open up the communication ports (80 or 443) for the AppDynamics SaaS Controller IP address the subnet range is: 69.27.44.0/24.

SMTP Service for SaaS

To enable email and SMS notifications you must configure SMTP. See Configure the SMTP Server.
For SaaS users, AppDynamics has an SMTP service running on every machine.

The configuration is:

SMTP Host: localhost
SMTP Port: 25

No authentication is needed.

Contact Support

For questions about the service contact the AppDynamics Support Team.

SaaS Availability and Security

- Service Availability
- Customer Account Login Security
- Hosting
- Data Access
- Data Collection
- Data Communication

This topic summarizes the service availability and security AppDynamics provides for customers who use the AppDynamics SaaS platform.

Service Availability

AppDynamics makes every best effort to operate and manage the AppDynamics SaaS platform with a goal of 99.5% uptime Service Level Agreement (SLA), excluding planned maintenance windows. AppDynamics actively monitors the latency of the SaaS platform 24/7 from different locations around the world to ensure AppDynamics delivers the best quality of service.

Customer Account Login Security

The AppDynamics user interface (UI) uses TLS 1.0 with AES 256 bit encryption terminated at the server to ensure end-to-end security over the wire.

For additional security, AppDynamics can restrict UI access to customer corporate networks. This is available for dedicated SaaS hosting plans only.

Hosting

The AppDynamics SaaS platform (servers, infrastructure and storage) is hosted in one of the largest Tier III data centers in North America. The data center is designed and constructed to deliver world-class physical security, power availability, infrastructure flexibility, and growth capacity. The data center provider is SSAE 16 SOC 1 Type II compliant, which means that it has been fully independently audited to verify the validity and functionality of its control activities and processes.

Every server is operated in a fully redundant fail-over pair to ensure high availability. Data is backed up nightly, stored redundantly and can be restored rapidly in case of failure. AppDynamics also provides an off-site backup service that is available at additional cost.

Security updates and patches are actively evaluated by engineers and are deployed based upon the security risks and stability benefits they offer to the AppDynamics SaaS platform and customers.

Data Access

Access to the AppDynamics SaaS platform infrastructure and data is secured by multiple authentication challenges including RSA and DSA key pairs, passwords, and network access control lists. Infrastructure and data access is restricted to AppDynamics employees and contractors, all of whom are under strict confidentiality agreements.

System and Network activity is actively monitored by a team of engineers 24/7. Failed authentication attempts are audited and engineers are paged immediately so that any possible intrusion or threat can be investigated promptly. Standard firewall policies are deployed to block all access except to ports required for AppDynamics SaaS platform and agent communication.

Data Collection

AppDynamics agents collect metrics that relate to the performance, health and resources of an application, its components (transactions, code libraries) and related infrastructure (nodes, tiers) that service those components.

Data Communication

AppDynamics agents typically push data using one-way HTTP or HTTPS connections to a single host (a Controller) which has been allocated to one or more customer accounts. AppDynamics offers dedicated Controllers for customers who require their data to be isolated.

For added security, agents can be configured to send data using encrypted transmission by simply selecting HTTPS port 443 and setting "controller-ssl-enabled" to true in the agent configuration. AppDynamics agents also have built-in support for outbound HTTP proxies for customers using these security mechanisms.

AppDynamics uses random staggering on agent data communication to the AppDynamics SaaS platform so traffic is spread evenly to minimize bursts and spikes of network traffic from your data center to the AppDynamics SaaS platform.

The following table shows typical bandwidth usage by number of agents, given the default agent configuration and typical application conditions:

Number of Agents	Typical Network Bandwidth Used (per minute)
1	300 Kbit to 500 Kbit
100	30 Mbit to 50 Mbit
1000	300 Mbit to 500 Mbit

These figures assume a 1:1 relationship between an agent and a JVM/CLR.

Get Started with AppDynamics On-Premise

Follow these steps to get started with AppDynamics.

If you are reading a PDF of this document, use your Help Center login to access additional documentation at http://docs.appdynamics.com.

Expert Advice

Deploying APM in the Enterprise... the Path of the Rock Star
By Jim Hirschauer

- Design Your AppDynamics Deployment
- Size and Verify the Controller Environment
- Download AppDynamics
- Install the AppDynamics Controller
- Install the AppDynamics App Agents

- Install the AppDynamics Web and Mobile Agents
- Review the Dashboards and Flow Maps
- Review Defaults and Configure Business Transactions, if Needed
- Review Defaults and Configure Client-Side Monitoring, if Needed
- Review Default Health Rules and Set Up Policies
- Review Default Error Detection
- Explore Additional Data and Metric Features
- Configure Advanced Features
- Start Monitoring and Troubleshooting

Design Your AppDynamics Deployment

- Learn about Business Transaction Monitoring and identify which critical business transactions you want to monitor.
- Learn about AppDynamics End User Experience and decide whether you want to use this feature.
- Learn about how to map your application components to the AppDynamics business application, tier, and node model. See Logical Model and Name Business Applications, Tiers, and Nodes.
- Based on the model, plan how you will specify AppDynamics application, tier, and node names during installation.
- Decide whether you want to monitor client-side usage with AppDynamics End User Experience.
- For Java environments, decide whether you want to use object instance tracking.

Size and Verify the Controller Environment

- Verify that you have the resources to support system requirements and the Controller performance profile. The profile reflects the number of nodes and AppDynamics applications that the Controller will monitor. For details see Controller System Requirements.

Download AppDynamics

- Download the AppDynamics software components from the Download Center. For details see Download AppDynamics Software.

Install the AppDynamics Controller

The AppDynamics Controller is the central management server where all data is stored and analyzed. All AppDynamics Agents connect to the Controller to report data, and the Controller provides a browser-based user interface for monitoring and troubleshooting application performance. A wizard installs the Controller in just a few minutes. Install the AppDynamics Controller only if you are using the on-premise Controller deployment option.

- Follow the instructions to install an on-premise Controller.
- Important installation and configuration considerations include:
  - High Availability
  - Backups
  - SSL and Certificates
  - User Authentication with LDAP or SAML

Install the AppDynamics App Agents

AppDynamics Application Agents collect data from your application servers and other monitored systems and report to the Controller. Install them on the application servers you want to instrument and any other machines you want to monitor. Follow the instructions to install the AppDynamics App Agents.

Install the AppDynamics Web and Mobile Agents

Install the client-side agents in your your mobile applications and web pages. See instructions for mobile and web.

Access the AppDynamics UI from a Browser

Once you have installed the Controller and agents, launch your web browser and connect to the AppDynamics User Interface (UI).

- For an on-premise Controller, the URL pattern is:

```
http://<controller-host>:<controller-port>/controller
```

  When using SSL, use port 443 or https to access the Controller.

Review the Dashboards and Flow Maps

AppDynamics automatically discovers the Business Transactions in your application environment. Browse the Application Dashboard and see the Flow Maps to visualize your application. You can resize and move icons around on the flow maps.

Review Defaults and Configure Business Transactions, if Needed

The default configurations may need to be further customized for your environment. For example, AppDynamics may have discovered transactions that you want to group together or even exclude, because you want to concentrate on the most important transactions. There may be business transactions that are not yet discovered for which you need to configure detection rules. See:

- Business Transaction Monitoring
- Configure Business Transaction Detection

Review Defaults and Configure Client-Side Monitoring, if Needed

You may want to refine the way AppDynamics names pages and mobile requests, for example, if the data for multiple web pages would be better understood under a single name. See:

- Configure Mobile Network Requests
- Set Up and Configure Web EUM

Review Defaults and Configure Databases and Remote Services, if Needed

AppDynamics automatically discovers "backends" such as databases, message queues, etc. by following calls in the application code. Look at the databases and remote services dashboards to make sure all necessary backends are revealed. If needed, change how backends are detected.

Review Default Health Rules and Set Up Policies

AppDynamics provides default Health Rules that define performance parameters for business transactions, such as the conditions that indicate a slow transaction, or when too much memory is being used. You can adjust the thresholds that define when a health rules is violated, create new health rules, and set up policies to specify actions to automate when health rules are violated.

Review Default Error Detection

AppDynamics detects errors and exceptions. You can review and, if needed, modify the error detection rules. For example, some errors you may want to ignore.

Explore Additional Data and Metric Features

Explore these features to gain more insight into application performance:

- Data Collectors
- Business Metrics
- (for Java environments) JMX Metrics
- Machine Agent Custom Metrics

Configure Advanced Features

Additional features you may want to use include:

- Custom Dashboards
- Automation
- AppDynamics Extensions and Integrations

Start Monitoring and Troubleshooting

Start getting the benefits of AppDynamics! See:

- AppDynamics in Action Videos
- AppDynamics Features

Download AppDynamics Software

- Accessing the AppDynamics Download Center
    - Download Tips
- AppDynamics Software Components
- Access to Older Versions
- Downloading from the Linux Shell
- Learn More

Accessing the AppDynamics Download Center

You should have received a Welcome email from AppDynamics. The Welcome email contains credentials for you to log in to the AppDynamics Support Center.

If you have not received this Welcome email, contact your AppDynamics Sales Representative or email support@appdynamics.com.

Access the AppDynamics Download Center (http://download.appdynamics.com) and browse to the appropriate section on the Download Center to download the relevant files.

Download Tips

Always copy or transfer the downloaded files in binary mode.

If you have downloaded a binary on Windows, and you are moving it to a Unix environment, the transfer program must use binary mode.

⚠ For each file you download, verify that the download is complete and that the file is not corrupted. Run a checksum tool and compare the results against the checksum information on the download site.

AppDynamics Software Components

AppDynamics Software Component	Description	SaaS	On-Premise
Controller	Central management server where all data is stored and analyzed.	N/A	Required
Java App Server Agent	Instrumentation Agent for Java virtual machines. This component must be installed on each Java application server you want to instrument through AppDynamics.	Required for Java	Required for Java
.NET App Server Agent (includes a Machine Agent by default)	Instrumentation Agent for .NET Common Language Runtime (CLR). This component must be installed on those worker processes that you want to instrument through AppDynamics.	Required for .NET	Required for .NET
PHP Agent	App agent for PHP installations.	Required for PHP	Required for PHP

Machine Agent	Collects hardware performance metrics and can be installed on any machine in your environment. The Machine Agent can be extended to collect data from other subsystems.	Optional	Optional
GeoServer	For End User Management. See Customize Your Web EUM Deployment.	Optional	Optional

Access to Older Versions

The AppDynamics Download Center provides downloads of older versions of the products.

- For On-Premise installations: Go to "AD Pro-OnPremise".
- For SaaS installations: Go to "AD Pro-SaaS".

On the top-right corner, click on the drop-down list to select the version that you want to download.

Downloading from the Linux Shell

To download AppDynamics software from a Linux shell, you can use the wget utility or cURL.

When using these tools, you first need to authenticate to the AppDynamics domain and store the resulting session ID in a file. Next, send the request to download the software, passing the session information file as a cookie.

For example, on Fedora you can use the following wget commands:

```
wget --save-cookies cookies.txt  --post-data
'username=<USERNAME>&password=<PASSWORD>'
https://login.appdynamics.com/sso/login/
```

```
wget --content-disposition --load-cookies cookies.txt '<URL_TO_FILE>'
```

On the Windows platform add the --no-check-certificate option.

The equivalent cURL commands are:

```
curl -c cookies.txt -d 'username=<USERNAME>&password=<PASSWORD>'
https://login.appdynamics.com/sso/login/
```

```
curl -O -b cookies.txt <URL_TO_FILE>
```

You can discover the URL for the file to download at the AppDynamics Download Center.

Learn More

- Supported Environments and Versions
- Agent - Controller Compatibility Matrix

Quick Start for DevOps

Get Started

Get Started on SaaS

Get Started On-Premise

Features Overview

Tutorials for Java

Use AppDynamics for the First Time with Java
Understanding Events
Understanding Flow Maps
Understanding Slow Transactions
Understanding the Transaction Scorecard
Understanding Server Health
Understanding Exceptions

Learn More

Best Practices for Application Developers
Best Practices for Performance and Quality Assurance Engineers
Best Practices for Operations Professionals

Monitor Your Applications

Business Transaction Monitoring
Web EUM
Monitor Events
Monitor Application Change Events
Background Task Monitoring
Backend Monitoring
Infrastructure Monitoring
Monitor CLRs
Monitor Hardware

Troubleshoot Application Performance

Troubleshoot Slow Response Times
Troubleshoot Health Rule Violations

Transaction Snapshots
Troubleshoot Node Problems
Troubleshoot Errors
Diagnostic Sessions
Troubleshoot Java Memory Issues

Quick Start for Architects

Get Started

Supported Environments and Versions
Get Started on SaaS
Get Started On-Premise

Concepts

Features Overview
Architecture
Logical Model
Mapping Application Services to the AppDynamics Model
Behavior Learning and Anomaly Detection
Thresholds
Glossary

Basic Configuration

Configure Business Transaction Detection
Configure Policies
Configure Baselines
Configure Thresholds
Configure Error Detection
Set Up and Configure Web EUM
Getting Started Wizard for Alerts
Custom Dashboards

Analyze

Business Metrics
Infrastructure Metrics
Reports
Compare Releases

Learn More

Advanced Configuration

Hierarchical Configuration Model
Configure Data Collectors
Configure Code Metric Information Points
Configure Custom Exit Points
Configure Call Graphs
Configure Background Tasks
Remove Stale Backends

Configure Custom Memory Structures for Java
Configure JMX Metrics from MBeans
Configure Multi-Threaded Transactions (Java)
Configure Object Instance Tracking for Java
Configure Transaction Snapshots
Configure Memory Monitoring (Java)
Internationalization
Build an Alerting Extension
Export and Import Business Application Configurations

Integration

AppDynamics Extensions and Integrations
Use the AppDynamics REST API

Automation

Workflow Overview
Cloud Computing Workflows

Quick Start for Administrators

Get Started

Architecture
Get Started on SaaS
Get Started On-Premise
Logical Model

Basic Administration

Release Notes for AppDynamics Pro
Supported Environments and Versions
Install and Upgrade AppDynamics
Name Business Applications, Tiers, and Nodes

Learn More

Advanced Administration

Implement SSL
Best Practices for Failover Scenarios for Java
Administer Agents
Administer the Controller
User Authentication and Permissions
AppDynamics for Large Enterprises

Quick Start for Operators

Get Started

AppDynamics Essentials

Tutorials for Java

Use AppDynamics for the First Time with Java
Understanding Events
Understanding Flow Maps
Understanding Slow Transactions
Understanding the Transaction Scorecard
Understanding Server Health
Understanding Exceptions

Learn More

Best Practices for Operations Professionals
Set User Preferences
Custom Dashboards

Set User Preferences

- Change Account Settings
  - To change your password
  - To change your display name and contact email
- Configure View Preferences
  - To configure view preferences
- Advanced Features
- About Debug Mode

Users in the Controller UI can change their passwords, account settings, date and time format, and other user-specific settings in the User Preferences tab, as described by this topic.

Change Account Settings

The account settings for a Controller UI user include the user's password, display name, and contact email.

Passwords and account settings are attributes of local user accounts (that is, AppDynamics users). If your Controller is configured to use an external authentication mechanism to control access, such as SAML or LDAP, you need to change the equivalent settings in the external system instead.

To change your password

1. From the upper right menu bar of the Controller UI, click the **User** icon and then **My Preferences**.
2. Click the **Change Password** button.
3. Enter your current password in the **Current Password** field.
4. Type your new password in the **New Password** and **Confirm New Password** fields, and then click **Save**.

You will need to enter the new password the next time you log in.

To change your display name and contact email

The display name is the name that the Controller uses to identify you in certain screen text and messages. For example, it appears in notifications to other Controller users when you share a dashboard with them.

1. In the Controller UI, access your user preferences by clicking the **User** icon and then **My**

Preferences.
2. Click the **Edit Account** button.
   Note that your username cannot be changed. To effect a change of a username, you would need to have an administrator delete your account and create another one with the new name.
3. Enter new values for:
   - **Display Name**: Your new display name in the UI.
   - **Email**: The email address where you want to receive notifications from the Controller.
4. Enter your current password in the **Current Password** field. The Controller uses this field to ensure your identity before making changes to your account. If you do not provide the correct password, your changes will not be applied.
5. Click the **Save** button.

The change take effect immediately.

Configure View Preferences

The Controller UI allows individual users to customize certain view preferences in the UI, such as the time and date format and style elements of the UI.

To configure view preferences

1. In the Controller UI, access your user preferences by clicking the **User** icon and then **My Preferences**.
2. In the View Preferences of the page, configure any of the following settings as desired:
   - **Date Format**: By default, the format is MM/DD/YY (for example, 09/25/14). Choose an alternate format from the drop-down menu.
   - **Use 24 hour Time Format**: Enable this option if you want the UI to represent time in 24-hour time format instead of 12 hour clock format.
   - **Enable Help Pop-ups**: Help popups provide help text in context in the Controller UI. By default, they are enabled. To prevent help popups from appearing in the UI, clear this checkbox.
     Alternatively, you can prevent individual popups by selecting the **Don't Show Again** c heckbox when the popup appears. To clear the list of popups marked as "Don't Show Again", click the **Reset All** button.
   - **Graph Color Scheme for the Metric Browser**: Select either Light or Dark to change the metric browser color scheme.
   - **Graph Color Scheme for All Other Graphs**: Select either **Light** or **Dark** to change the navigation panel color scheme.
   - **Maximum number of Backends to display in graphical views**: This setting limits the number of backend systems that appear in flowcharts or other graphical depictions of your application environment. The default is 20.
   - **Font**: Determines the font type used in the UI. For screen text, the Controller UI uses a font set it embeds and manages by default. If the operating system of the computer on which you access the Controller UI uses a non-English language, you can configure the UI to use non-English languages by setting the font to use system fonts instead. For more information, see Internationalization.
   - **Mouse Wheel Legacy Mode**: If scrolling in the Controller UI using your mouse scroll doesn't work properly, you should try enabling the **Mouse Wheel Legacy Mode** optio n. This may be necessary if accessing the Controller UI with certain older browsers.

3. You may need to log out of the UI and log back in to see the effects of your changes.

Advanced Features

AppDynamics cloud automation features allow you to set up workflows that are triggered by policy conditions. By default, the features are hidden in the UI. You need to specifically enable the features to configure cloud auto-scaling features.

To enable cloud automation features in the UI, enable the **Show Cloud Auto-Scaling** option. Enabling this option displays the **Cloud Auto-Scaling** link at the bottom left side of the UI, under the Alert & Respond menu.

See Workflow Overview for information about using cloud scaling automation features. See Policie s for information about specifying policy conditions that trigger workflows.

About Debug Mode

The debug mode in the Controller UI is primarily intended for internal use by the AppDynamics development team.

In some cases, you may be asked to enable debug mode in consultation with AppDynamics Support, for example, when you are troubleshooting an issue. However, it is important to note that certain debug mode options can negatively impact Controller performance. For this reason, you should only enable debug mode when directly advised to do so by AppDynamics Support.

Use AppDynamics

Expert Advice

Deploying APM in the Enterprise

by Jim Hirschauer

Configure AppDynamics

Configure Discovery

Hierarchical Configuration Model
Organizing Traffic as Business Transactions
Configure Business Transaction Detection
Configure Transaction Snapshots
Configure Backend Detection (Java)
Configure Backend Detection

(.NET)
Configure Data Collectors
Configure Business Metric Information Points
Configure Code Metric Information Points

Configure Business Transaction Monitoring

Configure Thresholds
Configure Baselines
Configure Call Graphs
Configure Error Detection
Configure Background Tasks

Configure End User Experience

Set Up and Configure Web EUM

Configure Alerting and Automation

Configure Policies
Configure Alerts
Configure Custom Actions

Glossary

- Action
- Agent
- Alert
- Alert Digest
- Anomaly Detection
- Ajax Request
- Application Performance Management
- APM
- App Server
- App Server Agent
- Application
- Application Dashboard
- ART
- Average Response Time
- Backend
- Background Task
- Baseline
- Baseline Deviation
- Baseline Pattern
- BCI
- Browser Snapshot
- Business Application

- Business Metric
- Business Transaction
- Bytecode Injection
- Call Graph
- Compute Cloud
- Controller
- Detection
- Diagnostic Session
- Discovery
- Distributed Application
- End User Monitoring
- End User Response Time
- Entry Point
- Error
- Error Transaction
- Exception
- EUM
- Event
- Exit Point
- Flow Map
- Health
- Health Rule
- Health Rule Violation
- High Availability (HA) Cluster
- Histogram
- Home Page
- iframe
- Information Point
- Key Performance Indicator
- KPI
- Machine
- Machine Agent
- Managed Application
- Match Condition
- Node
- On-Premise
- Pageview
- Policy
- Real-time Business Metric
- Remote Service
- REST
- Request
- SaaS
- Scorecard
- Task
- Tier
- Tag, trace, and learn
- Threshold
- Trace, tracing

- Transaction Correlation
- Transaction Snapshot
- Transaction Splitting
- Workflow

Action

An action is an automatic response to an event, based on a policy. There are various types of actions including sending alerts, taking diagnostic snapshots, remediation through scripts, cloud auto-scaling, or custom.

Agent

In AppDynamics an agent collects data about an application (and optionally machine) performance or about web page performance. AppDynamics has application server agents (app agents), machine agents, mobile agents, and a JavaScript agent. For example, the App Agent for Java intercepts the bytecode loading at the classloader and enhances it before it is loaded in the JVM. See Bytecode Injection.

Alert

An alert notifies a recipient list of a problem or event; by email, SMS or customized to interface with external notification systems.

Alert Digest

An alert digest compiles alerts and sends the compilation sent by email or SMS to a recipient list at a configured time interval.

Anomaly Detection

Anomaly detection refers to the identification of metrics whose values are out of the normal range, where normal range is based on dynamic baselines that AppDynamics has created based on the previous performance of these metrics.

Ajax Request

An Ajax request is a request for data sent from a page using the XHR API. This API is used to send HTTP or HTTPS requests to a web server and to load the server response data back into the requesting page. The Ajax request is tracked as a child of the requesting page using EUM.

Application Performance Management

Application performance management monitors and manages the performance and availability of software applications in a production environment, focusing on relating IT metrics to business values.

According to Gartner research, application performance management includes: end user experience monitoring, application runtime architecture discovery and modeling, business transaction management, application component deep-dive monitoring, and application data analytics.

APM

See Application Performance Management.

App Server

An app server or application server provides software applications with services such as security, data services, transaction support, load balancing, and management of large distributed systems. Examples are a Java Virtual machine (JVM) or a Common Language Runtime (CLR).

App Server Agent

An app server agent or app agent monitors an application server. The App Agent for Java monitors a JVM. The App Agent for .NET monitors a CLR. See Node.

Application

See Business Application.

Application Dashboard

The application dashboard graphically represents high-level structural and status information for a single business application. The application dashboard includes a flow map, grid view, summary of key performance indicators.

ART

See Average Response Time.

Average Response Time

Average interval between the time the user request is received by the application server and the time that the response is returned to the application server. Does not include the network time for the request to reach the server or the time for the response bytes to reach the caller. Different from End User Response Time.

Backend

A backend or backend node is a software component that is not instrumented directly, but the flow of traffic to it can be monitored. Typical examples are a database or messaging service.

Background Task

A background task or a batch job is a scheduled program that runs without user intervention.

Baseline

A baseline provides a defined known point of reference for application performance. The baseline is established by configuration or by observing current performance. Baselines can be static or dynamic.

Baseline Deviation

A baseline deviation is the standard deviation from a baseline at a point in time. It is represented as an integer value. Baseline deviation can be used to configure health rule conditions based on the number of deviations. For example, you can configure a warning condition as 2 standard deviations from the baseline and a critical condition as 4 standard deviations from the baseline.

Baseline Pattern

A baseline pattern defines the base time period of data used to create baselines. It can be a fixed time range or rolling time range, in which the most recent x number of days is always used.

BCI

See Bytecode Injection.

Browser Snapshot

A browser snapshot presents a set of diagnostic data for an individual base page, iFrame or Ajax request. The data reports the end-user's experience starting with the Web browser. Browser snapshots are taken at periodic intervals and when certain performance thresholds are reached.

Business Application

An AppDynamics business application models all components or modules in an application environment that provide a complete set of functionality. A business application usually does not map directly to only one Java or .NET or PHP application, and often it maps to more than one.

Business Metric

See Real-time Business Metric and Information Point.

Business Transaction

A business transaction represents an aggregation of similar user requests to accomplish a logical user activity. Examples of these activities include: logging in, searching for items, adding items to the cart, checking out (e-commerce); content sections that users navigate such as sports, world news, entertainment (content portal); viewing a quote, buying and selling stocks, placing a watch (brokerage). A single request is a business transaction instance.

Bytecode Injection

Bytecode injection modifies a compiled class at runtime by injecting code into it immediately before it is loaded and run.

Call Graph

A call graph represents the calling relationships among subroutines in an application. It makes up a part of a transaction snapshot that is used to identify root cause of a performance problem.

Compute Cloud

A compute cloud delivers computing and storage capacity as a service. Examples are Amazon EC2, OpenStack, etc.

Controller

The Controller collects, stores, baselines, and analyzes performance data collected by app agents. A Controller can be installed On-Premise or you can use the AppDynamics SaaS model.

Detection

Detection is the process by which AppDynamics identifies a business transaction or backend in a managed application. Detection is also referred to as discovery.

Diagnostic Session

A diagnostic session is a session in which transaction snapshots are captured, with full call graphs. A diagnostic session can be started manually through the user interface or configured to start automatically when thresholds for slow, stalled, or error transactions are reached.

Discovery

See Detection.

Distributed Application

A distributed application runs on multiple computers in a network. Some of the computers can be virtual machines.

End User Monitoring

End User Experience Monitoring (EUM) provides performance information from the point of view of the client, whether that client is a web browser or a mobile native application. This is different from other types of AppDynamics monitoring, which typically begin at the application server. EUM collects a different set of metrics than the server-side app agents.

End User Response Time

Average interval between the time that an end-user initiates a request and the completion of the page load of the response in the user's browser. In the context of an Ajax request, the interval ends when the response has been completely processed. Not to be confused with Average Response Time.

Entry Point

An entry point begins or extends a business transaction. An entry point is usually a method or operation in your application code. AppDynamics automatically detects entry points for common frameworks, and you can configure entry points to customize how AppDynamics detects business transactions.

Error

An error in AppDynamics indicates an unhandled exception in the context of a business transaction, a logged error of the appropriate severity, or any exception called during an exit call, which prevents the transaction from working properly.

Error Transaction

An error transaction is an error that occurred during transaction execution. An error transaction can be caused by a logged error or a thrown exception.

Exception

An exception is a code-based anomalous or exceptional event, usually requiring special processing. It can occur in the context of a business transaction and outside of a business transaction

EUM

See End User Monitoring.

Event

An event represents an action or occurrence detected by the system that can be handled by the system. There are different event types.

Exit Point

An exit point is a call from an app server to a backend database, remote service or to another app server. AppDynamics automatically detects many exit points and you can configure custom exit points.

Flow Map

A flow map graphically represents the tiers, nodes, and backends and the process flows between them in a managed application.

Health

Health in AppDynamics refers to the extent to which the application being monitored operates within the acceptable performance limits defined by health rules. Health is indicated by a green/yellow/red color scheme.

Health Rule

Health rules allow you to select specific metrics as key to the overall health of an application and to define ranges for acceptable performance of those metrics. AppDynamics supplies default health rules that you can customize, and you can create new ones.

Health Rule Violation

A health rule violation exists if the conditions of a health rule are true.

High Availability (HA) Cluster

A cluster of computers that hosts duplicate server applications with the purpose of reducing down time. The HA cluster, also called a failover cluster, is enabled by redundant systems that guarantee continued delivery of service during system failure.

Histogram

A histogram is a graphical representation of the distribution of data, shown as adjacent rectangles, erected over discrete intervals (bins), with an area proportional to the frequency of the observations in the interval.

Home Page

The Web page you see when you first log into the AppDynamics Controller, before you have selected an application. See AppDynamics Home Page.

iframe

An iframe is an "inline frame", an HTML document that is embedded in another HTML document. It is tracked as a child page using EUM.

Information Point

An information point instruments a method in application code outside the context of any business transaction. It is used for monitoring the performance of the method itself or for capturing data from the method's input parameters or return value.

Key Performance Indicator

Key performance indicators are main metrics that an organization uses to measure its success. In AppDynamics, the key performance indicators are assumed to be load (number of calls and calls per minute), average response time, and errors (number of errors and errors per minute.)

KPI

See Key Performance Indicator.

Machine

A machine consists of hardware and an operating system. It hosts application services and it can be virtual.

Machine Agent

A machine agent instruments a machine to report data about hardware and the network to the Controller. AppDynamics provides both a Standalone Machine Agent and an embedded machine agent in the App Agent for .NET. The Standalone Machine Agent functionality can be extended to add additional metrics.

Managed Application

A managed application is an application with servers that are instrumented by AppDynamics.

Match Condition

A match condition frames a test consisting of a match criterion (such as a method name, servlet name, URI, parameter, hostname, etc.), a comparison operator typically selected from a drop-down list, and a value. Used in many types of AppDynamics configuration to specify entities to be included in or excluded from monitoring.

Node

A node is an instrumented Java application server or an instrumented Windows .NET application (IIS, executable, or service.) Instrumentation is accomplished by installing an AppDynamics App Agent. Nodes belong to tiers. See Tier.

On-Premise

On-Premise refers to an AppDynamics Pro installation where the controller is installed on machines at your site. Alternatively, AppDynamics offers AppDynamics Pro as an SaaS.

Pageview

A pageview is an instance of a web page being loaded into a Web browser.

Policy

A policy consists of a trigger based on events and an action respond to the event. A policy provides a mechanism for automating monitoring, alerting, and problem remediation.

Real-time Business Metric

Metric that measures items such as revenue per transaction, number of orders, number of credit card purchases and so on. Differ from a performance metric, which measure the performance of the application. Implemented through Information Pointss.

Remote Service

A remote service provides a service to a distributed application outside of the JVM or CLR. Examples are a Java Message Service or Web Service. See Backend.

REST

The AppDynamics REST API is implemented using Representational State Transfer (REST) Services. You use the REST API to retrieve information from AppDynamics programmatically.

Request

A request is a single instance of a business transaction; for example, 500 requests per minute for the "Checkout" business transaction.

SaaS

SaaS is an acronym for Software as a Service. AppDynamics provides AppDynamics Pro as an SaaS where the controller is hosted on AppDynamics in-house machines and monitors your applications, communicating over the internet with app server, Java, .NET, infrastructure, and database agents installed in your environments. You can, alternatively, install the AppDynamics Pro controller on your own equipment, an installation type referred to as On-Premise.

Scorecard

A visual summary of the performance of a business transaction within a specified time range, covering percentage of instances that are normal slow, very slow, stalled or errors.

Task

A task codifies a unit of work as a set of instructions and is a component of a step in a workflow.

Tier

A tier represents a key service in an application environment, such as a website or processing application. A tier is composed of one or more nodes or one or more backends. See Node and Ba ckend. An "originating tier" is the tier that receives the first request of a business transaction. A "downstream tier" is a tier that is called from another tier.

Tag, trace, and learn

Tag, trace, and learn is a methodology used for tracing code execution and discovering the business transaction context.

Threshold

A threshold is a configurable boundary of acceptable or normal business transaction or background task performance.

Trace, tracing

Tracing is following the execution of software code and recording information about the execution, usually for debugging or performance monitoring purposes.

Transaction Correlation

Transaction correlation is the internal mechanism that allows AppDynamics to do transaction tracing in modern web applications, tying together distributed components into a single entity, the business transaction, for monitoring purposes.

Transaction Snapshot

A transaction snapshot depicts a set of diagnostic data for an individual business transaction across all app servers through which the business transaction has passed. The data reports the user's experience starting with the application server. A transaction snapshot is taken at a specific point in time.

Transaction Splitting

The process of using a dynamic value to customize how Business Transactions are identified.

Workflow

A workflow builds a sequence of steps in which each step consists of one or more tasks that are executed on a machine instrumented by a Standalone Machine Agent.

AppDynamics Support

If you have questions about using AppDynamics please file a support ticket (http://help.appdynamics.com/tickets/new) and attach any logs that are related to your issue.

To get log files from the Controller, see Controller Log Files for Troubleshooting.

To get heap, histogram, and thread dumps see Controller Dump Files.

Use the Documentation Wiki

- Locate Information Using Search
- Locate Information Using Labels
- Use Comments for Feedback
- Generate a PDF or Word File from a Page
- Download a PDF of the Documentation
- Generate PDF or HTML Versions of Pages
- Known Issues
    - "Duplicate headers received from server" Error on PDF Export

Locate Information Using Search

1. Type a word or phrase in the Search box located at the upper right of the wiki pages.

2. Scroll through the list of results.

Locate Information Using Labels

Labels are keywords or tags that are added to pages. To view the labels defined in the system,

1. Click **Browse -> Labels**. The wiki displays a list of popular labels.

2. Click on any label to find pages tagged with that label.

Note: Labels are handy but may not return all instances of a topic. Use Search for more fine-grained results.

Use Comments for Feedback

AppDynamics is happy to receive your feedback! We strongly encourage you to leave comments on pages. Scroll to the bottom of a page and click **Add Comment** to add your comment. Comments are regularly monitored.

Generate a PDF or Word File from a Page

1. Navigate to the page.

2. Click **Tools -> Export to PDF** or **Tools -> Export to Word**.

Download a PDF of the Documentation

The entire documentation is available to download in PDF format as a ZIP file.

1. In the left navigation bar, scroll to the bottom.

2. Click the most recent ZIP file.

Generate PDF or HTML Versions of Pages

You can generate PDF or HTML versions of part or all of the documentation.

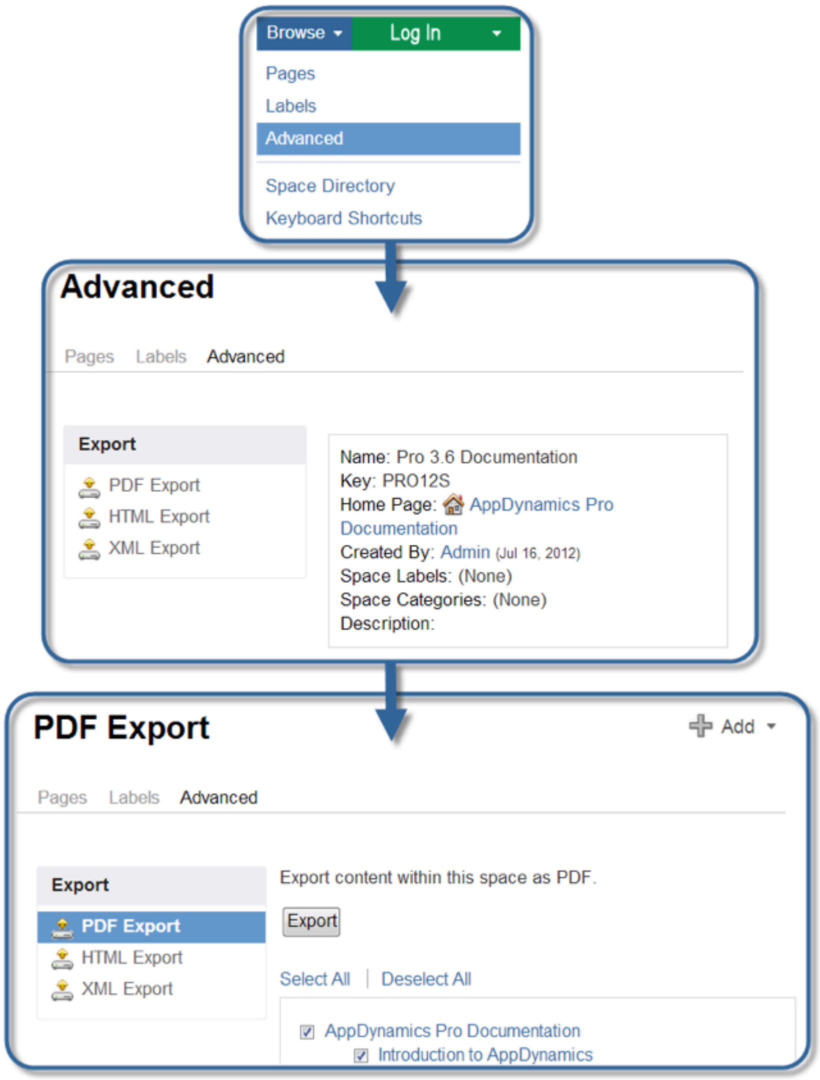
1. Go to the "Browse" menu.

2. Click **Browse -> Advanced**.

3. Click **PDF/HTML Export**.

4. Select those pages that you want to be available offline.

5. Click on "Export" button.



Name: Pro 3.6 Documentation
Key: PRO12S
Home Page: AppDynamics Pro Documentation
Created By: Admin (Jul 16, 2012)
Space Labels: (None)
Space Categories: (None)
Description:

Export content within this space as PDF.

[✓] AppDynamics Pro Documentation
[✓] Introduction to AppDynamics

Known Issues

"Duplicate headers received from server" Error on PDF Export

When exporting a single page as PDF the follow error occurs:

```
Duplicate headers received from server
The response from the server contained duplicate headers. This problem is
generally the result of a misconfigured website or proxy. Only the website or
proxy administrator can fix this issue.
Error 349 (net::ERR_RESPONSE_HEADERS_MULTIPLE_CONTENT_DISPOSITION): Multiple
distinct Content-Disposition headers received. This is disallowed to protect
against HTTP response splitting attacks.
```

This is most likely due to a problem in the Chrome browser. In Chrome, files with non-standard alpha-numeric characters in the file name, such as , $ % & * characters, can cause this issue. Try another browser.

Controller Dump Files

- Controller Troubleshooting Information Needed by the AppDynamics Support Team
  - To get the heap and histogram dump files
  - To take four thread dumps at three second intervals
  - Provide the AppDynamics Support Team with the files

Controller Troubleshooting Information Needed by the AppDynamics Support Team

If requested, collect the following files to send to the AppDynamics Support Team.

To get the heap and histogram dump files

- Get the **process id of the Controller** to use in subsequent commands.

```
ps -ef | grep java
```

- Get the **heap dump before garbage collection** using following command:

```
<java-jdk-install-dir>/bin/jmap
-dump:format=b,file=heap_before_live.bin <Controller_pid>
```

- Get the **histogram before garbage collection** using following command:

```
<java-jdk-install-dir>/bin/jmap -histo <Controller_pid> | head
-200 > histo_before_live.txt
```

- Get the **histogram before garbage collection** using following command:

```
<java-jdk-install-dir>/bin/jmap -histo:live <Controller_pid> |
head -200 > histo_after_live.txt
```

2. Save the generated files:

- heap_before_live.bin
- histo_before_live.txt
- histo_after_live.txt

To take four thread dumps at three second intervals

- Using the Controller process ID, execute following command:

```
kill -3 <Controller_pid>
```

- Save the

<Controller_Installation_Directory>/appserver/glassfish/domains/domain1/logs/jvm.log file.

Provide the AppDynamics Support Team with the files

- heap_before_live.bin
- histo_before_live.txt
- histo_after_live.txt
- jvm.log

Controller Log Files for Troubleshooting

- Log Files that Record Possible Problems
- Learn More

Use install.log and the files in the Controller logs directory to troubleshoot Controller errors.

Log Files that Record Possible Problems

The following log files may reveal errors:

- **install.log:**
  This file contains information about events of the install process such as extraction, preparation and other post-processing tasks. It is located at <Controller_Installation_directory>.
- **server.log:**
  This file contains log information for the embedded Glassfish application server used by the Controller. It is located at <Controller_Installation_directory>/logs.
- **database.log:**
  This file contains log information for the MySQL database that is used by the Controller. It is located at <Controller_Installation_directory>/logs.
- **installation.log:**
  This file contains log information about the installation specific to Install4j. It is located at <Controller_Installation_directory>/.install4j.
  Also include the entire .install4j directory and the latest <OS_Temporary_Directory>/i4j_log*, for example /tmp/i4j_log*.

If after analyzing the logs you have questions, please file a support ticket and attach any logs that are related to your issue. See AppDynamics Support for suggestions.

Learn More

- Controller Logs

Download Doc PDFs

Be aware: AppDynamics does not regenerate the PDF documentation every time that the documentation is updated. If having the most up-to-date documentation is essential to you please see the 3.8 wiki docs.

- Release Notes
- All Docs
- Doc Subsets

- To make your own PDFs

Release Notes

Release notes are inclusive. All updates since 3.8.0 are noted in the 3.8.x Release Notes.

3.8.2 Release Notes for AppDynamics Pro (309kB) - Updated May 21, 2014

All Docs

Complete Documentation Suite (64.22 MB) - Updated April 2, 2014

Doc Subsets

AppDynamics Essentials (1.53 MB) - Updated April 2, 2014

AppDynamics Features (15.06 MB) - Updated April 2, 2014

Getting Started (281 kB) - Updated April 2, 2014

- Getting Started with AppDynamics SaaS (242 kB) - Updated April 2, 2014
- Getting Started with AppDynamics On-Premise (233 kB) - Updated April 2, 2104

Introduction and Tutorials (5.96 MB) - Updated April 2, 2014

Use AppDynamics (30.81 MB) - Updated April 2, 2014

AppDynamics for Java (21.05 MB) - Updated April 2, 2014

AppDynamics for .NET (5.30 MB) - Updated April 2, 2014

AppDynamics for PHP (4.10 MB) - Updated April 2, 2014

AppDynamics for Node.js (1.03 MB) - Updated April 2, 2014

End User Experience (6.98 MB) - Updated April 2, 2014

Configure AppDynamics (15.91 MB) - Updated April 2, 2014

AppDynamics Extensions and Integrations, include REST and 3rd Parties (1.32 MB) - Updated April 2, 2014

Monitoring (16.04 MB) - Updated April 2, 2014

Alert and Respond (2.76 MB) - Updated April 2, 2014

Alert and Automate (3.58 MB) - Updated April 2, 2014

AppDynamics Administration (5.11 MB) - Updated April 2, 2014

Troubleshooting (7.75 MB) - Updated April 2, 2014

To make your own PDFs

See Use the Documentation Wiki.

License Information

From the top menu bar click **Settings** ⚙ **-> License** to see information about the

AppDynamics licenses installed on your system.

The License window shows the name of your customer account and the type of license (Trial, etc.) used for the account. It lists the number of agent licenses and expiration dates.

Contact salesops@appdynamics.com with any license questions or issues.

- Controller Licenses
- Web EUM License
- Mobile APM License

Documentation Map

Get Started

Get Started on SaaS

Get Started On-Premise

Features Overview
Architecture
Logical Model

Quick Start for DevOps
Quick Start for Operators
Quick Start for Administrators
Quick Start for Architects

Tutorials for Java

Use AppDynamics for the First Time with Java
Understanding Events
Understanding Flow Maps
Understanding Slow Transactions
Understanding the Transaction Scorecard
Understanding Server Health
Understanding Exceptions

Monitor and Troubleshoot

Best Practices for Application Developers

Monitor Your Applications

Business Transaction Monitoring
Web End User Experience Monitoring
Mobile Application Performance Monitoring
Monitor Events
Monitor Application Change Events
Background Task Monitoring
Health Rules

Troubleshoot Application Performance

Troubleshoot Slow Response Times
Troubleshoot Health Rule Violations
Transaction Snapshots
Troubleshoot Node Problems
Troubleshoot Errors
Diagnostic Sessions
Troubleshoot Java Memory Issues

Analysis

Business Metrics
Infrastructure Metrics
Reports
Compare Releases

Administration

Basic Administration

Release Notes for AppDynamics Pro
Supported Environments and Versions
Install and Upgrade AppDynamics
Name Business Applications, Tiers, and Nodes

Advanced Administration

AppDynamics for Large Enterprises
Implement SSL
Best Practices for Failover Scenarios for Java
Administer Agents
Administer the Controller
User Authentication and Permissions

Concepts

Architecture
Logical Model
Mapping Application Services to the
AppDynamics Model
Behavior Learning and Anomaly Detection
AppDynamics for Large Enterprises
Glossary

Configuration

Basic Configuration

Hierarchical Configuration Model
Configure Business Transaction Detection
Configure Policies

Configure Baselines
Configure Thresholds
Configure Error Detection
Set Up and Configure Web EUM
Getting Started Wizard for Alerts
Custom Dashboards

Advanced Configuration

Configure Data Collectors
Configure Business Metric Information Points
Configure Code Metric Information Points
Configure Custom Exit Points
Configure Call Graphs
Configure Background Tasks
Remove Stale Backends
Configure Custom Memory Structures for Java
Configure JMX Metrics from MBeans
Configure Multi-Threaded Transactions for Java
Configure Object Instance Tracking for Java
Configure Transaction Snapshots
Configure Memory Monitoring for Java
Internationalization
Build an Alerting Extension
Export and Import Business Application Configurations

Integration

AppDynamics Extensions and Integrations
Use the AppDynamics REST API

Automation

Workflow Overview
Cloud Computing Workflows

Reference

User Interface Reference

Dashboards
Transaction Analysis Tab
Flow Maps
Time Ranges
Call Graphs
Scorecards
KPI Graphs

FAQs

Controller Install and Admin FAQ

Controller High Availability FAQ
Machine Agent Install and Admin FAQ

AppDynamics Support

Controller Dump Files
Controller Log Files for Troubleshooting
Download Doc PDFs
Use the Documentation Wiki

AppDynamics Lite

- AppDynamics Lite Features
- Transition from Pro Trial to AppDynamics Lite
- Learn More

At the end of the trial period, AppDynamics Self-Service Pro Trial transitions to the free product edition, AppDynamics Lite. AppDynamics has retired previous versions of AppDynamics Lite.

AppDynamics Lite Features

AppDynamics Lite provides the same application monitoring and troubleshooting features as AppDynamics Pro. The free version includes the following agents:

1 App Agent for Java
1 App Agent for .NET
1 App Agent for PHP
3 Standalone Machine Agents

AppDynamics Lite restricts queries to the last 24 hours both from the user interface and via API. To retain history greater than 24 hours, upgrade to AppDynamics Pro.

Transition from Pro Trial to AppDynamics Lite

When the trial period ends and AppDynamics detects an expired AppDynamics Pro Trial license, the Controller resets all agents. For details about agent reset, see Resetting App Agents.

After the reset, the Controller limits agent registration to one each of the App Agent for Java, the App Agent for .NET, and the Agent for PHP. The Controller permits the first agent of each type to register and use a license.

AppDynamics recommends you uninstall any agents you don't use:

- Uninstall the App Agent for Java
- Uninstall the App Agent for .NET
- Uninstall the App Agent for PHP

Learn More

AppDynamics Essentials