

An Oracle White Paper
June, 2013

Enterprise Manager 12c Cloud Control Service Management White Paper

Executive Overview	1
Introduction	1
Service Management	2
Service Definition (Modeling)	2
Service Availability	5
Service Test	7
To create an ATS service test within a Service, follow these steps	9
System	12
Metric Dependencies in Service Targets	13
Service Level Agreements	15
Services Dashboard	17
Services Reports	18
Conclusion	19

Executive Overview

Oracle Enterprise Manager is Oracle's integrated enterprise IT management product line and provides the industry's first complete cloud lifecycle management solution. Oracle Enterprise Manager's Business-Driven IT Management capabilities allow you to quickly set up, manage and support enterprise clouds and traditional Oracle IT environments from applications to disk. Enterprise Manager allows customers to achieve:

Best service levels for traditional and cloud applications through management from a business perspective for a number of Oracle Application Products, including Fusion Applications, E-Business Suite, Siebel, PeopleSoft, and JD Edwards EnterpriseOne (herein referred to as Oracle Application Products).

Maximum return on IT management investment through the best solutions for intelligent management of the Oracle stack and engineered systems

Unmatched customer support experience through real-time integration of Oracle's knowledgebase with each customer environment

Introduction

Oracle Enterprise Manager provides the industry's only comprehensive solution for managing the aforementioned Oracle Application Products – all from a business-perspective. In addition to Oracle Enterprise Manager's cross-product capabilities including user experience management, change and configuration management, patching, provisioning, testing, performance management, integrated diagnostics and automatic tuning, Oracle Application customers benefit from executive-level visibility into the business impact of these Applications and the ability to manage these applications in enterprise cloud environments. With Oracle Enterprise Manager, customers can manage their Oracle Applications environments from a single console.

Service Management

Oracle Enterprise Manager 12c Cloud Control provides a comprehensive monitoring solution that helps IT organizations achieve high availability and performance and optimized service levels for their business services. The Enterprise Manager Service Management solution allows you to actively monitor and report on the availability and performance of a wide variety of services, including end-user business functions (mail, infrastructure, network and database services), Web applications, Oracle Application Products. Using service tests or synthetic transactions executed from remote user locations, businesses are able to monitor services from the end user's perspective. It equips administrators with a proactive 24 X 7 approach to ensure that all critical services and applications are available and provide consistent high levels of service to end-users.

EM12c Cloud Control monitors not only individual components in the IT infrastructure, but also the applications hosted by those components, allowing you to model and monitor business functions from an end-user perspective. If modeled correctly, services can provide an accurate measure of the availability, performance and resource usage of the function or application they are modeling.

Service management involves:

1. Service Definition (Modeling)
2. Service Tests
3. Systems
4. Service Level Agreements

Service Definition (Modeling)

A **service** is defined as an entity that models one or more business functions in an enterprise and is useful for business users. Some examples of possible service descriptions include Siebel CRM applications, EBS Financial Applications, online banking portals, e-commerce application or an e-mail service – just to name a few.

EM12c Cloud Control allows you to define one or more services that represent the business functions or applications that run in your enterprise. Services can be defined by associating targets with entities like systems or service tests.

Using these service tests or system metrics you can measure the performance and availability of critical business functions, receive alerts when there is a problem, identify common issues and diagnose causes of failures.

In Enterprise Manager 12c Cloud Control, you can create the following service models:

1. Generic Service
2. Aggregate Service

A **Generic Service** is used to monitor a single system and/or combination of one or more service tests.

An **Aggregate Service** is combination of one or more services, called subservices. A subservice is any service created in Enterprise Manager. The availability, performance, business criteria and resource usage for the aggregate service depend on the availability, performance, business criteria and resource usage for the individual subservices comprising the service.

Generic and aggregate services can be defined using Enterprise Manager Console, Cloud Control GUI, or the Enterprise Manager Command Line Interface (EMCLI). New wizards have been added to support creation of these services and following EMCLI commands have been added in EMCLI to support automatic integration of services:

1. create_service
2. enable_test
3. disable_test
4. delete_test
5. add_beacon
6. remove_beacon
7. assign_test_to_target
8. change_service_system_assoc
9. remove_service_system_assoc

10. set_availability
11. set_key_beacons_and tests
12. set_properties
13. sync_beacon

Based on these two service models many other types of services are supported in Enterprise Manager. For example:

1. Generic Service
2. Aggregate Service
3. Business Application
4. EM Service
5. Service Entry Point
6. Oracle Database Service
7. Oracle applications Service
8. Siebel Services
9. Beehive Services etc.

Note: Creation of new Web and Forms Applications is deprecated in EM 12c Cloud Control but existing service definitions for this type continue to be reported. The features for these services have now been rolled into the Generic Service type.

The key parameters to measure a service are:

- 1. Status:** This indicates whether a service is up or down at a given point in time. The status of a service is based on the status of its underlying system components and their metrics. The status of a service will be down if the metrics of the underlying system component have exceeded their thresholds.

The availability status of a service is based on the result of the service tests or on the availability of the associated system. System based availability can either be based on the 'System target itself' or the user can choose few components from the underlying system and designate them as critical components that determine the availability of a service.

- 2. Performance:** The performance of a service depends on the performance of the underlying metrics. Depending on the requirements, Administrators can choose to monitor their own desired aspects of performance. It can be a single

response time or an aggregation of multiple response time measurements if the service test involves multiple steps or the service provides multiple business functions. Examples include: Query response time, transaction creation time, etc.

3. Usage: Similarly, the usage of a service is dependent on the underlying metrics. For example: The CPU usage of a Hosting box in the service can be built on top of the host CPU metric. Or it can be a business activity measurement for a service. Examples include: Number of orders processed by an order entry service, number of users accessing a service, etc.

For service targets, the **metrics are computed** based on either the underlying service test or the underlying system components. The service target copies the values of the base underlying metric and stores the copied value in the repository. This ensures that even if the base metric is deleted the service metric will still retain its values. The user can define performance, usage and business metrics, any metric will be computed in the context of the Service once promoted. Metric promotion can be done for any system metric.

Note: System metrics can be used as both performance and usage metrics. However, service test metrics can only be used as performance metrics

Service Availability

The 'availability' of a service is a measure of the end user's ability to access the service at a given point in time. However, the rules of what constitutes availability may differ from one application to another. For example, for a Siebel Customer Relationship Management (CRM) application, availability may mean that a user can successfully log on to the application and access a sales report. For an online store, availability may be monitored based on whether the user can successfully log in, browse the store and make an online purchase.

Enterprise Manager 12c Cloud Control allows the user to define the availability of a service based on “Service Test(s)” or “System”.

1. Service Test(s) Based Availability

If the availability of the service is determined by the availability of a critical functionality to end users, you should select “Test Based Availability”. Examples of critical functions include accessing e-mail, generating a sales report, performing online banking transactions and so on. While defining a service test, you can choose the protocol that most closely matches the critical functionality of the monitored business process and beacon locations that match the locations of user communities.

One can define one or more service tests using standard protocols and designate one or more service tests as “Key Tests.” These key tests can be executed by one or more “Key Beacons” in different user communities. A service is considered available if one or all key tests can be executed successfully by at least one beacon, depending on one’s availability definition. Beacons are explained later in this doc.

2. System Based Availability

A Service’s availability can alternatively be based on the underlying system that hosts the service. One should select the components that are critical to running service and designate one or more components as ‘Key Components’, which are used to determine the availability of the service. Each system component (target type) will do its own calculation to determine status/response. If that calculation fails the component status fails.

The service is considered available as long as at least one or all key components are up and running, depending on one’s availability definition.

Service Test

A service test is a synthetic test that emulates the way clients access the service (a user session). A service test consists of two parts. A test script and a test beacon from which the script gets executed.

A script is the set of instructions that have to be executed to emulate a user session. This is a narrow description of the wide set of options Service Management options.

The following test (script) protocols are supported:

HTTP(s), DNS, FTP, Ping, ICMP Ping, IMAP, JDBC SQL Timing, LDAP, HTTP Ping, NNTP, Oracle SQL Timing, POP, Port Checker (includes SSL), SMTP, SOAP, TNSPing, Web Transaction, WEBDAV, ATS transaction, CalDAV and Custom Script.

The Oracle recommended method of recording a service test is **by creating an ATS transaction**, using the **Openscript** component of the Oracle Application Test Suite (OATS). You can use the **'ATS Transaction' Test Type** to record and play back various types of Web transactions, such as Web/HTTP, Oracle EBS/Forms, Oracle Fusion/ADF, Siebel, Adobe Flex etc.

How to create an Openscript Loadtest script

As a prerequisite to recording any ATS script that can be used for any Service test, you'll need to download the Openscript software to your Windows desktop to record a websession. [Links to download Openscript from OTN](#). By clicking link "Oracle Application Testing Suite - Openscript Only" you can install the Openscript component only, the only component needed to setup ATS Service Tests. The use of Openscript for recording ATS service test transactions is included in the Service Management license.

Once the Openscript component is installed on your Windows Desktop, please follow the following steps/guidelines:

1. In Openscript, select **File, New Script**
2. Select "Load Testing (Protocol Automation) -> Web/HTTP" and click "**Next**". (if you are recording against an Oracle application, please check if specific recording features are offered for your application, if so, pls select the appropriate protocol.)
3. Then name your script and click "**Finish**"
Note : you have to select a valid repository normally the folder named default.
4. Click the **Record** button [Red color] on top of the Openscript toolbar to start recording the script. Now Record your Session as if you would be browsing the application at any given time
5. Click the "**Stop**" button to stop recording the script
6. Click the "**Save**" button and save the newly recorded script.
7. Click the "**Play**" button to play the newly-recorded script and verify successful script execution in the results window.
8. Click "File" and "Export", provide a filename for your recording and select "ok". A zip file will be created that can be used as a source within a Service.

Note: A very useful feature of Openscript (amongst many) is the Databank option, this allows you to store script parameters in an easy to edit file (.csv) that is included with your script. The Databank can be used to change parameters like login name and password, hostname details, variables like product ID's, etc). A databank can be included in your script export and used within a Service to create different tests re-using the same script if required.

To create an ATS service test within a Service, follow these steps

1. From the **Targets** menu, select **Services**.
2. Select Generic Service from the **Type** drop down list and click **Add**.
3. Enter a name for the service and select a time zone. Click **Next**.
In the Create Generic Service: Availability page, select **Service Test**.
In the Create Generic Service: Service Test, select the **Test Type** as ATS Transaction.
4. Enter a name, description, and collection frequency for the service test.
5. In the ATS Zip Archive region, click **Upload**. Click **Browse** and upload the script bundle that has been exported from Openscript. Click **Continue**.

You will return to the ATS Service Test page where you can specify the following:

Usage Options : You can configure the script variable values by selecting the required usage option. You can either use the values recorded during the transaction or use the databank. A databank is an external CSV file that ATS scripts can refer to supply different

input values over multiple iterations of the same script. For example, a login script can use a databank file, named login_credential.csv, to supply different login credentials during iteration. You can select:

- **Use Recorded Values** : While playing back the transaction, the beacon uses the recorded values in the script.

- **Use Values From EM Test Property** : You can specify values in the databank columns. These values are used by the beacon while playing back the script. This is useful if the same value is to be used for each variable. If variables defined as test properties, the value can be easily modified without having to modify script bundle or databank files.

- **Loop Through All Databank Records** : While playing back the transaction, the beacon will go through each row in the script. For example, the first iteration will use the first rows of all the databanks. The second iteration will use the second rows of data and so on.

Encryption Password : If you have configured ATS Openscript to encrypt script data (using the **Openscript View File > Openscript Preferences > Generic > Encryption**) option, when you create the scripts, you need to enter the same encryption password as specified in the ATS openscript, so that beacon can play back the script properly.

For additional details on how to create ATS Load testing scripts, please refer to the [Oracle® Functional Testing OpenScript User's Guide](#).

The beacon functionality is built into Enterprise Manager Agents and is used to execute tests [they playback the recorded transactions] on behalf of the service and collect metrics on behalf of the service. Multiple beacons can be deployed at various locations to test for the same script from different geographic locations.

The following diagram displays the dependency between Service and Service Tests.
Select **Service Home Page** and click **Test Performance**.

Name	Total Time (ms)	Connect Time (ms)	Dns-fetch Time (ms)	First Byte Time (ms)	HTML Time (ms)	Non-HTML Time (ms)	Transfer Rate (KB per second)	Status Description	Downloaded Total Time (ms)	DNS Time (ms)	Downloaded Slowest Page Time
HTTP Ping											
Expand All Collapse All											
HTTP Ping											
Google ping											
EMManagement1 Beacon	114	21	0	52	0	0		1	114	0	
OHS ping											
EMManagement1 Beacon	613	318	158	132	0	0		1	613	0	

This page shows the tests that are running as part of the service as well as the latest results of these tests from each of the beacons.

In the current service model all [key] service tests that are configured for a service will be executed by all the [key] beacons monitoring the tests. A service is considered available if one or all key tests can be executed successfully by at least one beacon, depending on one's availability definition.

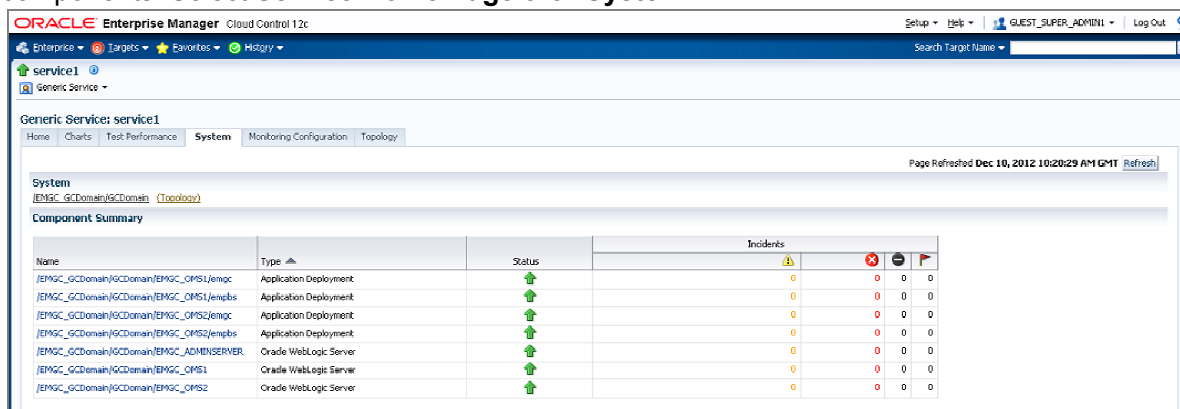
To run different kinds of service tests on the Internet and the Intranet, beacons can be configured for the following:

1. SSL Certificates for Web transaction and Port Checker service tests
2. Dedicated Beacons- If one wants to run a large number of tests (e.g., several hundred per minute) on a single beacon, one can configure dedicated beacons on an Agent to take full advantage of the dedicated hardware.
3. Web Proxy for a beacon, to run tests across firewalls.
4. Windows beacons for Web transaction (browser) playback features

System

A system is a set of infrastructure components that work together to host one or more business functions (e.g., such as a database and its associated listener). A service can be created on top of a system to expose the entry points of business functions provided by the system.

The following diagram shows the dependency between services, system and system components. Select **Service Home Page** click **System**



The screenshot shows the Oracle Enterprise Manager Cloud Control 12c interface. The main content area is titled "System" and displays a "Component Summary" table. The table lists several components, all with a status of "Up" (indicated by a green arrow) and zero incidents. The components are:

Name	Type	Status	Incidents
/EMGC_GCDomain/GCDomain/EMGC_OMS1/emgc	Application Deployment	Up	0
/EMGC_GCDomain/GCDomain/EMGC_OMS1/emgbs	Application Deployment	Up	0
/EMGC_GCDomain/GCDomain/EMGC_OMS2/emgc	Application Deployment	Up	0
/EMGC_GCDomain/GCDomain/EMGC_OMS2/emgbs	Application Deployment	Up	0
/EMGC_GCDomain/GCDomain/EMGC_ADMINSERVER	Oracle WebLogic Server	Up	0
/EMGC_GCDomain/GCDomain/EMGC_OMS1	Oracle WebLogic Server	Up	0
/EMGC_GCDomain/GCDomain/EMGC_OMS2	Oracle WebLogic Server	Up	0

The page shows the system that is being consumed by the service as well as the key components that are participating in the availability of the service.

When a user creates a service based on a system, associations gets created between the service and the system and also between the service and the critical system components. The user can view the associations using the Topology tab.

In Enterprise Manager 12c Cloud Control, the Oracle Application Plug-ins deliver pre-defined systems. Alternatively, users can create their own custom systems For example, to monitor an e-mail application in Enterprise Manager, a user can first create a system, such as "Mail System" that consists of the database, listener, application server and host targets on which the e-mail application runs. The user can then create a service target that represents the e-mail application and specify that this application needs to run on the Mail System target.

Another example could be, a system like Siebel Enterprise that has certain web applications that are hosted on Application servers and use Oracle database. All targets hosting on one or more hosts. The user can create a “Siebel Enterprise” system that consists of the database, listener, application server and host targets on which the web applications runs.

To create a System:

1. Select the target members.
2. View the associations between the components of the system using the Topology Viewer.
3. Add charts that will appear in the System Charts page. The charts represent the overall performance of the system or the components of the system. Based on the target type of the components you select in the Components page, some charts are predefined. If there are Metric Extensions defined for any member target, one can select those custom metrics.
4. Customize the refresh frequency and specify the format for viewing component status, alerts and policy violations in the system dashboard.

The system target helps in monitoring the status of components, discovering dependencies/topology, and collecting relevant metrics to be used to determine the health of the overall service.

Metric Dependencies in Service Targets

Users can promote one or more metrics from the underlying system components to measure the performance of a service. These metrics can be added in any of the following categories ' performance' 'usage' or 'business'.

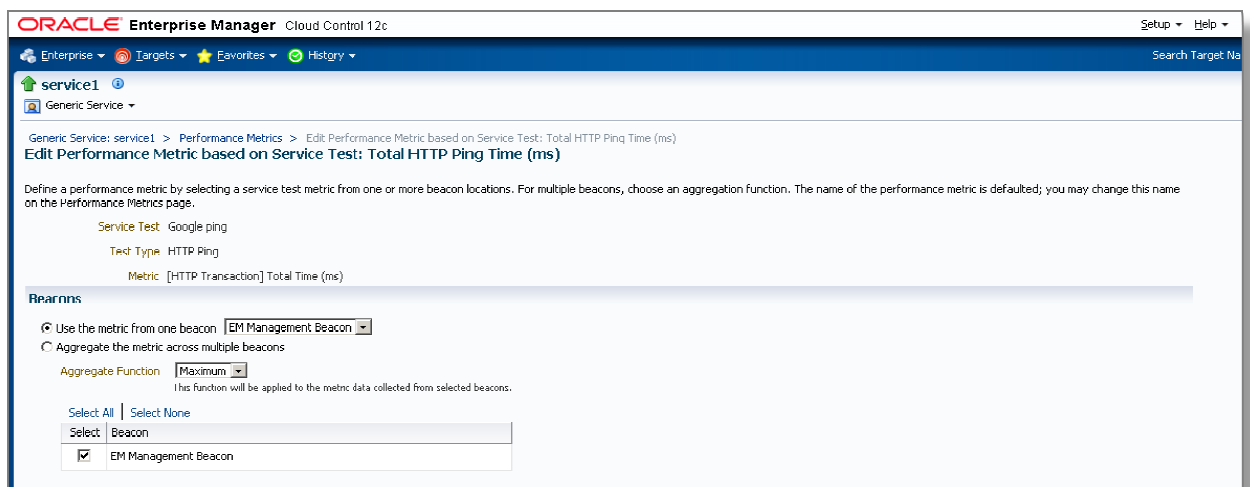
In service targets, metrics are computed based on either the underlying service test or the underlying system components. The service target copies the values of the base underlying metric and stores the copied value in the repository. This way ensures that even if the base metric is deleted the service metric will still retain its values.

Note: System member metrics can be used as both performance and usage metrics. However, test metrics can be used only as performance metrics.

1. To view the metric source of an already computed metric

Select **Service Home Page**, then select **Monitoring configuration** and click on either the performance metric or the usage metric link. Alternatively, from the **Service** menu, select **Administration**, then select **Performance (or Usage) Metrics**.

Select the relevant metric and click **Edit** button



In the above image you can edit the **Total HTTP Ping Time (ms)** metric for the **Google ping** service test.

The image also shows that the metric is derived based on the values reported by the **Management Beacon**. If there were multiple beacons, you can either pick the metric from a specific beacon or from a group of beacons. If a metric is selected from a group of beacons, you can aggregate the metric based on statistical functions such as min, max, average and sum. -

2. To promote a new metric

Go to either the performance or usage metric page and select the source (system/test) and click the **Go**.

The rest of the steps are similar to the ones mentioned above.

Service Level Agreements

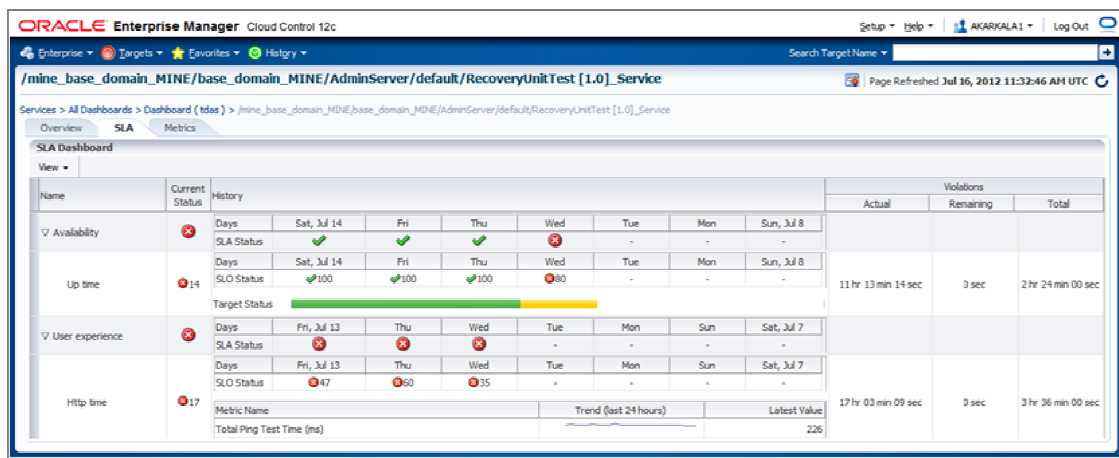
The results of metrics collected via systems and synthetic tests can be grouped into two categories, performance and availability. The result from both sets of metrics can be used for advanced status calculation of the entire service.

A Service Level Agreement (SLA) is a contract between the Business Service Provider and the customer defining the expected quality of service for a business period. An SLA can have one or more Service **Level Objectives (SLOs)** that define the service objectives to be provided. It's a logical grouping of individual measurable indicators.

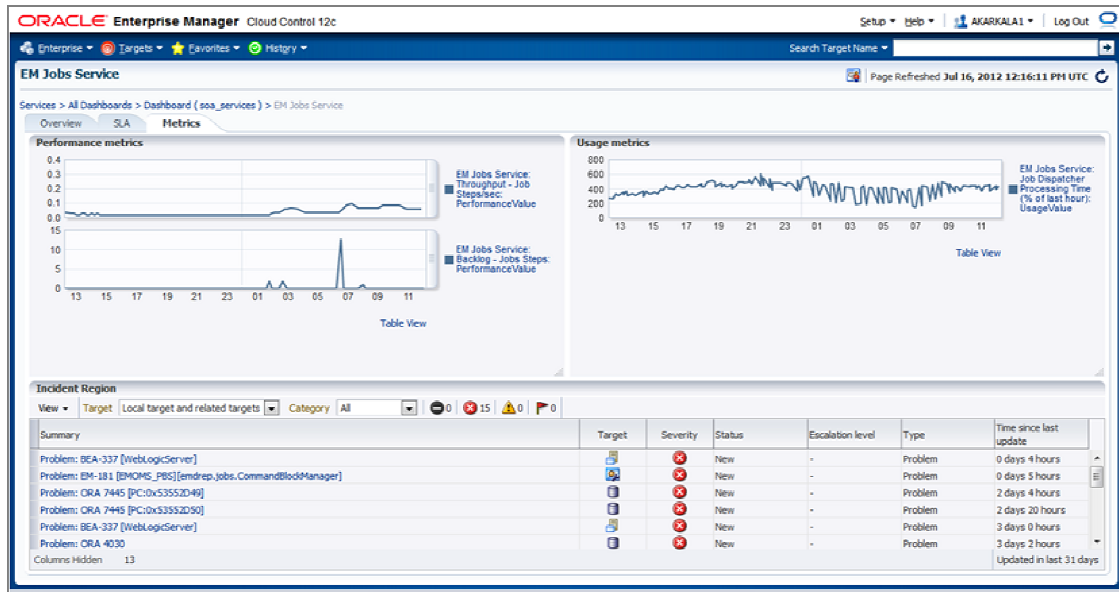
Service Level Indicators (SLIs) are measurable and quantifiable service metrics that can be used to evaluate the quality of a service..

By selecting metrics and setting objectives to them one can determine the health and status of the service. One can create agreement of how much time "SLO not met" is acceptable and when you run out of a service level agreement.

You can use the new service dashboard to monitor the status of a service, SLA and SLO. By looking at these screen details, one can correlate the SLA and its associated metrics.



The following screen correlates any performance or usage issue with an incident. Here you can investigate and find out which metric was responsible for an event or which metric might be impacting the SLA.



A **Service Level Rule** is defined as an assessment criteria used to determine service quality. It allows you to specify availability and performance criteria that your service must meet during business hours as defined in the Service Level Agreement. For example, e-mail service must be 99.99% available between 8am and 8pm, Monday through Friday.

A service level rule specifies the percentage of time a service meets the performance and availability criteria as defined in the Service Level Rule. By default, a service is expected to meet the specified criteria 85% of the time during defined business hours. One may raise or lower this percentage level according to the service expectations.

A **Service Level Rule** is based on the following:

Business Hours: Time range during which the service level should be calculated as specified in your Service Level Agreement.

Availability: Allows you to specify when the service should be considered available. This will only affect the service level calculations and not the actual availability state displayed in the console. You can choose a service to be considered up when it is one or more of the following states:

- Up: By default the service is considered to be Up or available.

– Under Blackout: This option allows you to specify service blackout time

(Planned activity that renders the service as technically unavailable) as available service time.

– Unknown: This option allows you to specify time that a service is unmonitored because the Management Agent is unavailable be counted as available service time.

Performance Criteria: You can optionally designate poor performance of a service as a Service Level violation. For example, if your Website is up, but it takes 10 seconds to load a single page, your service may be considered unavailable.

Business Criteria: Business criteria are useful in determining in the health of the business processes for a particular service. You can optionally define business metrics that can affect the Service Level. A Service Level violation occurs when a critical alert is generated for a specified business metric.

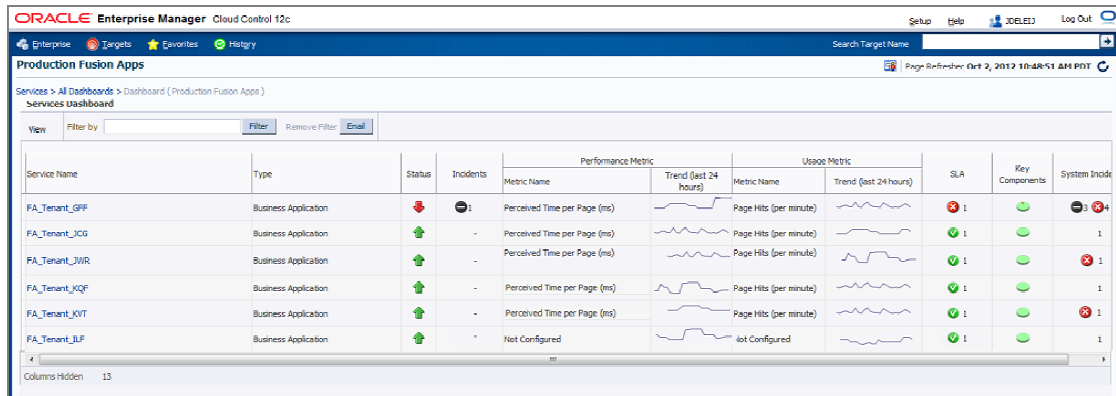
Actual Service Level: This is calculated as percentage of time during business hours that your service meets the specified availability, performance and business criteria.

Expected Service Level: Denotes a minimum acceptable service level that your service must meet over any relevant evaluation period. You can define only one service level rule for each service. The service level rule will be used to evaluate the Actual Service Level over a time period and compare it against the Expected Service Level.

Services Dashboard

Monitoring a service helps you ensure that your operational and service-level goals are met. In Enterprise Manager 12c Cloud Control, service levels are defined as the percentage of time during business hours that a service meets the specified availability, performance, and business criteria. Using the Services Dashboard, administrators can determine whether the service levels are compliant with business expectations and goals.

The Services Dashboard enables administrators to browse through all service-level-related information from a central location. The Services Dashboard illustrates the availability status of each service, performance and usage data, as well as service-level statistics. You can easily drill down to the root cause of the problem or determine the impact of a failed component on the service itself.



The Services Dashboard is a one stop screen where an administrator or an executive can have a quick glance at the all business services, their status, performance, issues(if any), SLA compliance, any underlying issues etc.. Services dashboard also allows end user to look at trends of performance and usage metrics.

With EMCC, services dashboard offers all info needed by admin or execs in one single screen. If a user wants to drill down to details of any service, dashboard allows that. Failure of an underlying system's incident or services incident allow administrator to take quick action to fix the issue. Objective of providing trending of performance and usage metrics is to let end user be pro-active and take appropriate action well in time.

Services Reports

Enterprise Manager Cloud Control provides out-of-box reports that are useful for direct display in operations centers or to be send out to users. The out of box reports focus on Dashboards, Service Level Agreements and Service Tests.

Conclusion

Service Management is one of the key areas within Enterprise Manager that allows for modeling of your important applications and services, quickly review status of all its system components, pro-actively test if your Service is still performing and available.

It can be used for singular system environment to a combined target that consumes data from multiple Service definitions. It is easy to use and to understand from a Business perspective and allows for great Dashboard capabilities, in short the key to link your operational services to Business departments.

References

*Oracle Enterprise Manager Grid Control 11g R1: Business Service Management by
Ashwin Kumar Karkala, Govinda Raj Sambamurthy*



Oracle EM12g Cloud Control
Services White Paper

June 2013

Author: Neelima Bawa

Contributing Authors: Shanmuga priya
Ulaganathan, Jurgen de Leijer

Oracle Corporation
World Headquarters
500 Oracle Parkway
Redwood Shores, CA 94065
U.S.A.

Worldwide Inquiries:
Phone: +1.650.506.7000
Fax: +1.650.506.7200

oracle.com



Oracle is committed to developing practices and products that help protect the environment

Copyright © 2013, Oracle and/or its affiliates. All rights reserved.

This document is provided for information purposes only, and the contents hereof are subject to change without notice. This document is not warranted to be error-free, nor subject to any other warranties or conditions, whether expressed orally or implied in law, including implied warranties and conditions of merchantability or fitness for a particular purpose. We specifically disclaim any liability with respect to this document, and no contractual obligations are formed either directly or indirectly by this document. This document may not be reproduced or transmitted in any form or by any means, electronic or mechanical, for any purpose, without our prior written permission.

Oracle and Java are registered trademarks of Oracle and/or its affiliates. Other names may be trademarks of their respective owners.

Intel and Intel Xeon are trademarks or registered trademarks of Intel Corporation. All SPARC trademarks are used under license and are trademarks or registered trademarks of SPARC International, Inc. AMD, Opteron, the AMD logo, and the AMD Opteron logo are trademarks or registered trademarks of Advanced Micro Devices. UNIX is a registered trademark of The Open Group. 0113

Hardware and Software, Engineered to Work Together