



Zenoss Service Impact Users Guide

Release 5.0.x

Zenoss, Inc.

www.zenoss.com

Zenoss Service Impact Users Guide

Copyright © 2015 Zenoss, Inc. All rights reserved.

Zenoss and the Zenoss logo are trademarks or registered trademarks of Zenoss, Inc., in the United States and other countries. All other trademarks, logos, and service marks are the property of Zenoss or other third parties. Use of these marks is prohibited without the express written consent of Zenoss, Inc., or the third-party owner.

Flash is a registered trademark of Adobe Systems Incorporated.

Oracle, the Oracle logo, Java, and MySQL are registered trademarks of the Oracle Corporation and/or its affiliates.

Linux is a registered trademark of Linus Torvalds.

RabbitMQ is a trademark of VMware, Inc.

SNMP Informant is a trademark of Garth K. Williams (Informant Systems, Inc.).

Sybase is a registered trademark of Sybase, Inc.

Tomcat is a trademark of the Apache Software Foundation.

VMware is a registered trademark or trademark of VMware, Inc. in the United States and/or other jurisdictions.

Windows is a registered trademark of Microsoft Corporation in the United States and other countries.

All other companies and products mentioned are trademarks and property of their respective owners.

Part Number: 1210.15.237

Zenoss, Inc.
11305 Four Points Drive
Bldg 1 - Suite 300
Austin, Texas 78726

Contents

Preface.....	4
Chapter 1: Introduction to Service Impact.....	5
Service models.....	6
State propagation policies.....	9
Service model definition process.....	13
Chapter 2: Tutorial.....	15
1. Define the service to model.....	15
2. Create logical node and subservice nodes for internet connections.....	21
3. Create service nodes for major network segments.....	25
4. Create summary nodes for critical paths.....	25
5. Create a node for the network service.....	26
6. Create a service node for the service model.....	26
7. Send events to fake devices.....	27
8. Remove tutorial service elements.....	29
Chapter 3: Exporting and importing service models.....	30
Exporting objects from a Service Impact environment.....	31
Importing a service graph.....	32
Reconciliation file.....	32
Reconciliation attempts.....	33
Performing the final commit.....	34
zenimpactimport.....	34
Chapter 4: Service Impact interface resources.....	36
Service Impact home page.....	36
Members.....	37
Add to Service dialog.....	38
Impact Events.....	38
Impact View.....	39
Impact Policies dialog.....	42
Logical node details view.....	44
Chapter 5: Configuring Service Impact.....	46
Production state propagation threshold.....	46
Service Impact server configuration files.....	47
Chapter 6: Service Impact Glossary.....	50

Preface

Zenoss Service Impact Users Guide provides information about using and administering Zenoss Service Impact (Service Impact).

Audience

This guide is designed for system administrators with Zenoss Resource Manager (Resource Manager) experience. In addition, administrators need working knowledge of Linux system administration, and their data center environment.

Related publications

Title	Description
<i>Zenoss Service Impact Installation Guide for Resource Manager 5.0.x</i>	Describes how to install Service Impact with a Resource Manager version 5.0.x deployment.
<i>Zenoss Service Impact Installation Guide for Resource Manager 4.2</i>	Describes how to install Service Impact with a Resource Manager version 4.2 deployment.
<i>Zenoss Service Impact Release Notes</i>	Describes known issues, fixed issues, and late-breaking information not already provided in the published documentation set.

Additional information and comments

If you have technical questions about this product that are not answered in this guide, visit the [Zenoss Support](#) site.

Zenoss welcomes your comments and suggestions regarding our documentation. To share your comments, please send an email to docs@zenoss.com. In the email, include the document title and part number. The part number appears at the end of the list of trademarks, at the front of this guide.

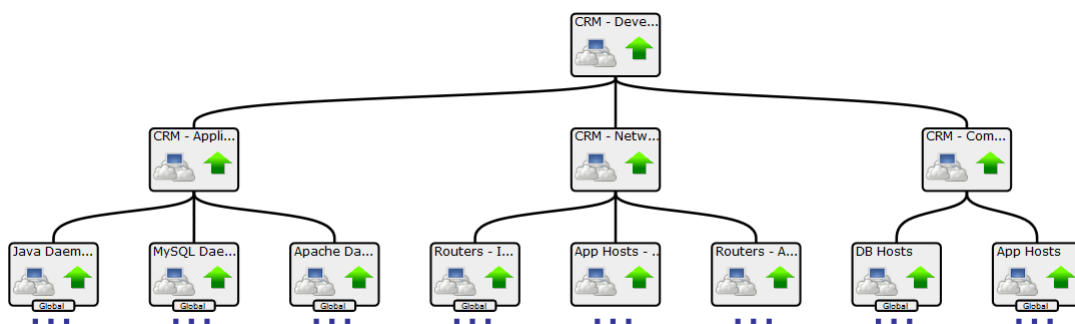
1

Introduction to Service Impact

Zenoss Service Impact (Service Impact) quickly reveals the affect of Resource Manager events on the *services* your company uses or provides. You can think of a service as all of the individual pieces of a system that allows internal or external users to accomplish something. For example, your environment might include several physical devices, database applications, and one or more web applications that support your company's website or online store. The individual pieces work together and rely on one other to provide the service as a whole--in this case, the company website.

To help you manage and maintain services, Service Impact performs near-real-time dependency tracking of the individual components of a service, and maintains the availability and performance state information of your services. When new events occur at any point in a service, Service Impact knows the origin of the event, and automatically performs a root-cause analysis (RCA), so you can quickly diagnose and triage problems—often, before end-users even notice or call to complain.

Service Impact enables you to quickly create rich models of all services in your environment and displays each one as an interactive graph, similar to the following example:



Service Impact overcomes the challenges of manual RCA for the following reasons:

- Network operations center personnel do not have uniform levels of domain knowledge.

In Service Impact, domain knowledge is captured in lightweight, context-sensitive policies.

- Events occur without context, eventually becoming a "sea of red" in the user interface.

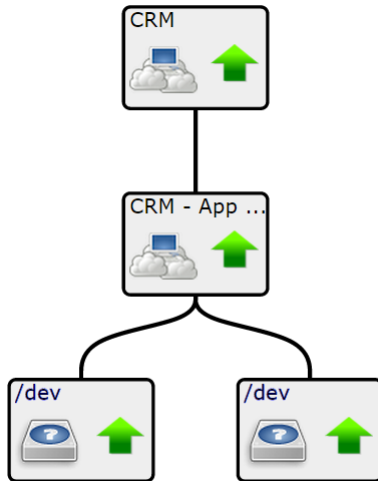
Because Service Impact understands the service's dependencies and their relationships, it shows you the origination point of an event and delivers notifications of significant events.

- The process of filtering out "noise" events, identifying impacted services, and starting recovery procedures is time-consuming and error-prone.

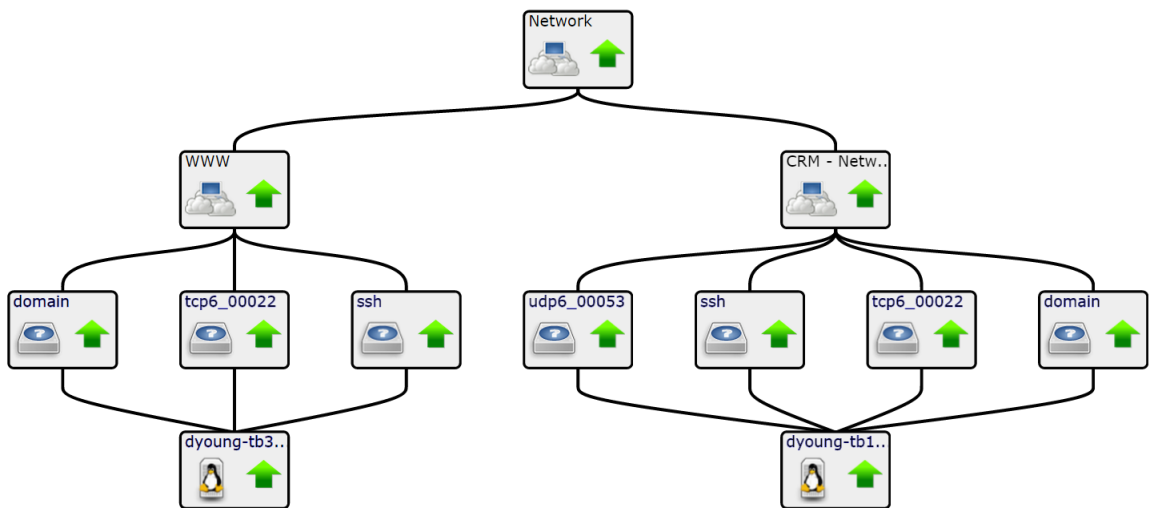
Service Impact maintains a near-real-time dependency map of each service and its dependencies, and can be used to alert on service-level performance metrics, before users notice or costly fines are levied.

Service models

Service Impact provides a graphical view of monitored resources and services in a structure called a *service model*. Service model graphs are built from model information stored in the Zope object database (ZODB), from other information available in Resource Manager, and from infrastructure objects that you create in Service Impact. A service model may consist of a single monitored resource and its child-dependencies, as shown in the following example:



A service model can also consist of multiple services models connected to one another in a hierarchy to show their inter-dependencies:



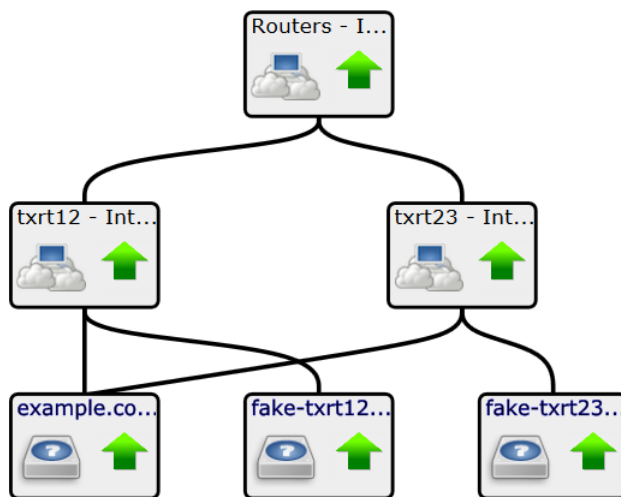
The resources or services that make up a service model are individual *nodes*. There are five different types of nodes:

- *Service nodes* represent either a service model as a whole, or as a single service (or subservice) that other service nodes rely on. Some examples of service nodes include a multi-node systems that represent specific functions

like Payroll processing, a Customer Relationship Management (CRM) tool; or departments in your company, such as Finance or Marketing.

- *Device nodes* represent resources that are monitored by Resource Manager, such as a Linux server. Additionally, ZenPacks that model a device node can extend the model to include other nodes. For more information about Service Impact ZenPacks, contact Zenoss Support.
- *Component nodes* also represent resources that are monitored by Resource Manager. Components are typically represent the elements of a device, such as an ethernet port.
- *Logical nodes* represent event states captured from arbitrary classes of resources or services that may or may not be monitored by Resource Manager. Logical nodes are customizable.
- *Organizing Group nodes* represent the current definition of an organizing group's devices and all child organizing group hierarchies. For more information about organizing groups, refer to [Dynamic Service Organizers and Organizing Groups](#) on page 8

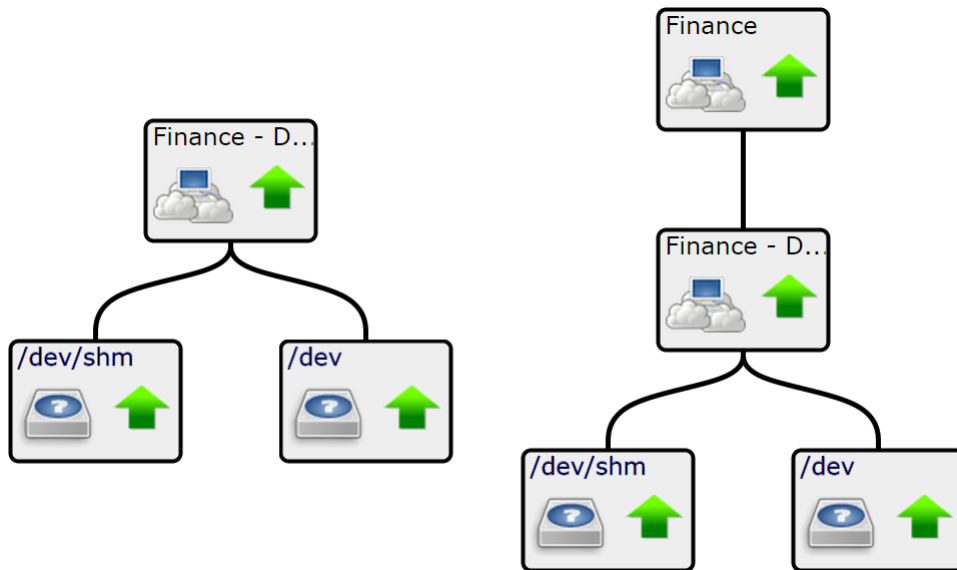
A node can be used in multiple service models, which greatly simplifies the process of creating new service models. For example, network-related nodes, such as switches or routers, are almost always shared and appear in multiple service models in the service model hierarchy. In the following example, in the following service model, the "example" node is shared with "txrt12" and "txrt23":



Service context

Service models define a particular *service context*. The service context includes all the nodes in the service, how they are related to each other, and the policies for each node within that service context. An event that occurs on a device node participating in two different service contexts may change its state and/or propagate the event in one service model's context, but behave differently in the other service context. For more information, see [State propagation policies](#) on page 9.

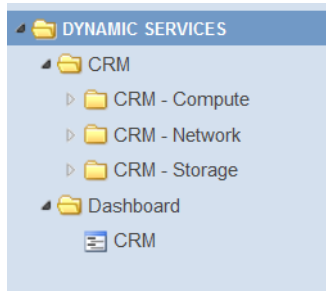
The active service context, the one you are viewing or managing, is always the highest-level resource in the service model. As you view or manage various nodes, it is important to know their current service context. In the following example, the service context of the service model on the left is "Finance - Databases," while the service context of the service model on the right is "Finance". The two models share common service nodes; however, they are in different service models, and thus different service contexts.



Dynamic Service Organizers and Organizing Groups

On the Resource Manager Services tab, Service Impact provides folders called Dynamic Service Organizers. You can use service organizers to group Dynamic Service Definitions. But note that you cannot add Service Organizers to a service model.

The following example shows organizers for Compute, Network, and Storage service models:



Dynamic Service Organizers provide valuable context in *service events* (state changes that affect service nodes) and in Zenoss Analytics reports.

On Resource Manager Infrastructure tab, you can use Organizing Groups, which consists of Groups, Locations, and Systems, to organize devices into different categories. For example you might organize devices into groups based on the organizational group they belong to, their physical location, or their system type to efficiently manage devices and use them in Service Impact. The following screen-shot shows an example of Organizing Groups:



Much like any other device, you can add organizing groups to a service model. In addition, as you add and remove devices and other organizing groups on the Infrastructure tab, the changes are automatically reflected in the related service models.

State propagation policies

State propagation policies determine how Availability and Performance state data is passed from the bottom to the top of a service model graph, and which node or nodes receive the state change data. When new events affecting a device, component, or logical node occur, Service Impact propagates that event through every service context that contains that node. Policy gateways identify how an event propagates to other entities in a service model.

State changes that affect service nodes result in Service Events, which are sent to Resource Manager, along with the event propagation path(s) from the originating event node to the service.

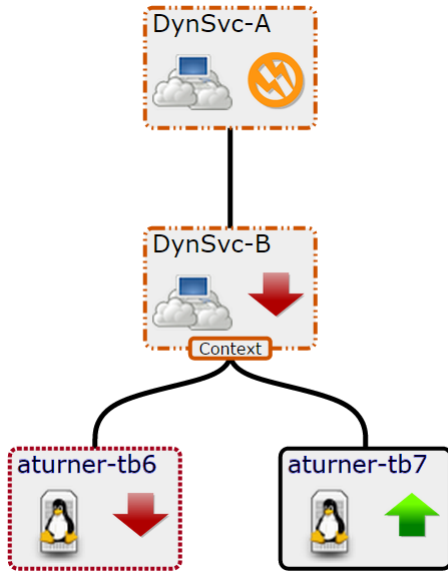
You can create and assign multiple policies to any node. In the Impact View, you can select the type of policy you want to create, edit or view by selecting the Availability Aspect or the Performance Aspect.

The policy type applied to the node determines which node or nodes receive the data. The three policy types, in precedence order are:

1 *Contextual policy*

Contextual policies propagate a node's state change to its immediate parent node (or nodes), only within the current service context.

In the following example, a Context policy is applied to Dynamic Service B (DynSvc-B). The policy states if one of the two child nodes is down, then the state of DynSvc-B should be degraded.

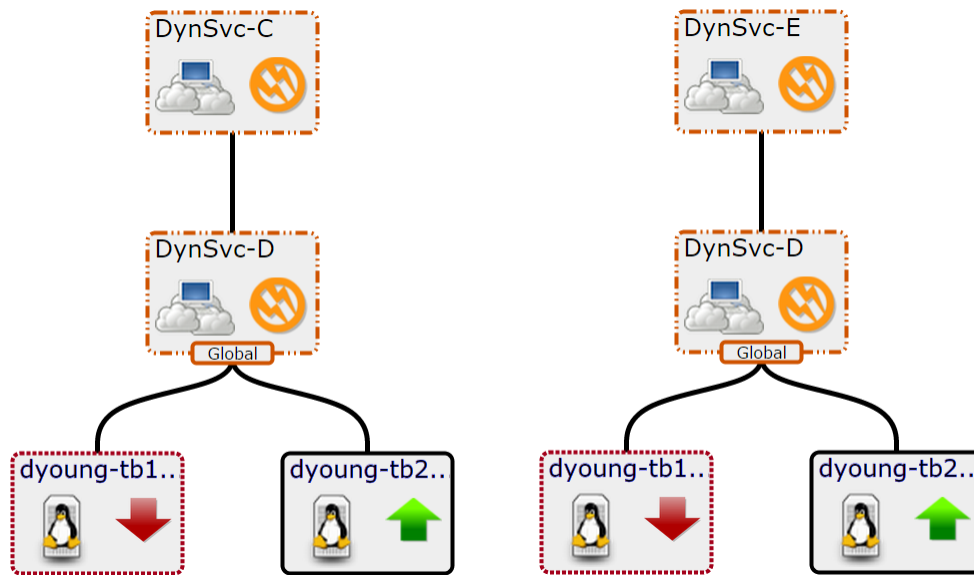


Note For more information about state symbols and borders, refer to *Actual and derived state* on page 11.

2 *Global policy*

Global policies apply to all service model contexts that share a node with a changed state. For example, if a global policy is applied to a node in a service model, and a child node has change to its state data, the new state is propagated to the parent node(s) in all service models the node belongs to.

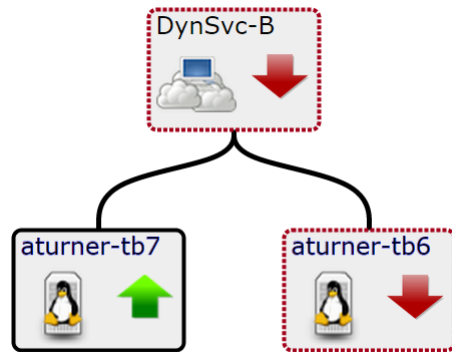
This example shows that DynSvc-D is used in two different service models: DynSvc-C and DynSvc-E. The global policy propagates the degraded state to all service contexts the DynSvc-D, of which DynSvc-D is a member.



3 *Default policy*

A node with the *default policy* (neither of the preceding policies) sends the state data of the worst condition affecting it to its parent node. The default policy is negated if you add either a contextual or global policy to a service model.

In this example, DynSvc-B does not have a policy applied. It's state is down because that is the worst case of its children.



If a node has multiple policies applied:

- A contextual policy overrides a global policy.
- Contextual and global policies override the default policy.

State triggers

A policy includes one or more user-defined *state triggers*. A state trigger defines a condition that the node's children must reach before the state change is propagated to the parent nodes of the current context (for a contextual policy) or to the parent nodes of all service models with the same child node (for a global policy). If a policy contains multiple triggers, each trigger is evaluated before the state data is sent.

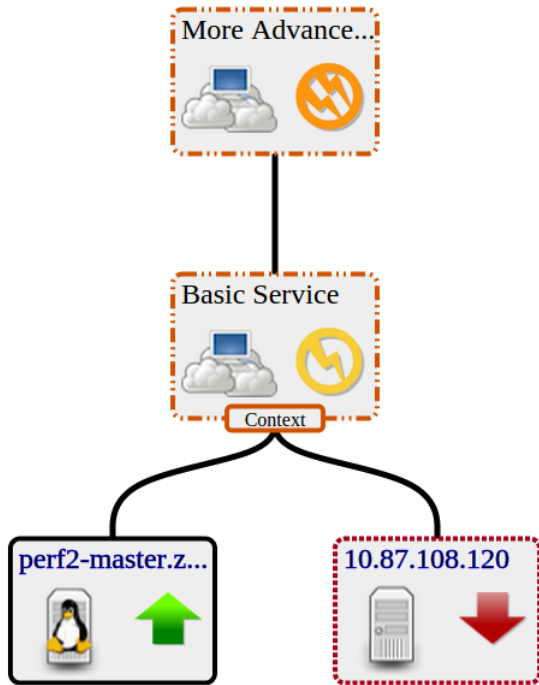
Nodes with *custom state providers* provide specialized options for defining state triggers. Service Impact always collects information from events belonging to most of the `/Status` subclasses, and all of the `/Perf` classes. You can customize Resource Manager to gather state data from events belonging to other classes. For example, you can customize state providers to define state triggers for device and component nodes monitored through customized classes provided by the ZenVMware and CiscoUCS ZenPacks.

Note If you create a policy with a state trigger that cannot be met, a state change will go undetected and unreported, leaving a service model in an inaccurate state. For example, suppose you create a policy and define the state trigger to be met when two children nodes go down. However if the service model this policy is applied to only contains a single child node, the trigger will never be met and the state data will always be shown as up. Because you applied a policy to the node, the default policy it also is negated.

Actual and derived state

Each node in a service model graph has two states: *Actual* and *Derived*. A device node's actual state is derived from events that occur on the device, irrespective of the different service models it may participate in. A service node's actual state is generated from the service model in its own service context. The node's derived state is the state that is derived from policy propagation within a given context. The symbol that appears inside the node's border reflects the actual state; the border that outlines the node reflects the node's derived state.

In the following example, the device node's actual and derived states are down. The parents of the device node have a degraded and at-risk actual state, yet the derived state based on the contextual policy, is down.





Visual state indicators

The Impact View provides two sets of states:

- *Availability*
- *Performance*

You can determine a node's Availability and Performance state based on the symbol that is displayed inside the node (this is also the node's Actual state) and the border surrounding the node (the node's Derived state based on policy propagation):

Availability	Performance	Symbol / Actual State	Border / Derived State
UP	ACCEPTABLE		
ATRISK	Not applicable.		
DEGRADED	DEGRADED		

Availability	Performance	Symbol / Actual State	Border / Derived State
DOWN	UNACCEPTABLE		

It is entirely possible, and normal, for a node's actual and derived states to be different. For example a service's actual state can be up, but an applied context policy has changed its derived state to down.



How policies work

The following list outlines the Service Impact policy information work-flow using a monitored switch as an example of a device node:

- 1 A switch that is monitored by Resource Manager goes down.
- 2 Resource Manager generates an event about the switch.
- 3 Service Impact reads the switch down event and changes the Actual state of the device node that represents the switch.
- 4 For each service context that the device node participates, Service Impact evaluates the state triggers to identify that node's Derived state within the service context.
- 5 Service Impact then propagates state data according to the context or global policies defined for the parent nodes of the device node.
- 6 If the device node's state changes the Derived state for the top-level node in the node's service context, Service Impact generates a Service Event and sends it to Resource Manager. (Like any other event, service event notifications can be customized.)
- 7 Finally, Service Impact performs a root cause analysis and generates confidence rankings of the impact of the switch down event for each service model graph the switch belongs to.

Service model definition process

The key to defining accurate service models is thorough dependency discovery. This section outlines a repeatable process for defining service models.

Note Defining service models is influenced by the following environmental factors:

- The level of organizational maturity (processes and tooling)
- The automation level of service life cycles (provisioning and management)
- The standardization level of applications and infrastructure

Nevertheless, by using the following process, you can create accurate service models.

- 1 Define the service to model.

A service is defined by its boundaries and type.

- A web service (for example, a human resources portal) or an IaaS platform (an Amazon EC2 instance) depends on specific resources, which may be internal or external.
- Software as a service (for example, *ServiceNow*) is defined by its deployment architecture.

In Service Impact, a service is represented as a *service node*.

2 Define services nodes for subservices.

A subservice has a direct relationship with one or more services: If the subservice fails or degrades, the service fails or degrades based on the propagation rules for that service context. Often, a subservice represents an infrastructure tier, such as a gateway or database service. At a lower level, a tier may be configured with redundant elements for high availability. However, Zenoss recommends modeling subservices as single points of failure.

In Service Impact, subservices are represented as *service nodes*. (Service node roles are contextual.)

3 Add device nodes, component nodes, logical nodes, and organizing groups to service and (or) subservice nodes.

Once subservices are defined as service nodes, the infrastructure resources that make up the subservices can be added.

4 Define global policies on subservice nodes.

Policies capture domain knowledge about services, and enable the automated dependency tracking and RCA computation that make Service Impact so valuable. Zenoss recommends using global policies instead of contextual policies wherever possible, to facilitate subservice re-use. Contextual policies are valuable, but their use cases are relatively rare.

The primary reason is that most deployment scenarios utilize shared resources. For example, a hypervisor hosts virtual machines that belong to separate services. One service node with a global policy can apply to all virtual machines, across all service models. The service relationship, not service ownership, is the key to determining the relevance of events and sending state data to parent nodes.

Note Steps 2, 3, and 4 may be performed iteratively.

5 Perform gap analysis.

Identify gaps in monitoring processes by analyzing failure scenarios. Are the key measurement points of each node in the service model properly monitored? For example, are synthetic transactions in place for web servers? Are ping checks being performed against host operating systems? Service Impact can only act on events that flow through Resource Manager.

6 Test failure scenarios.

The `zensendevent` command generates synthetic events, which are invaluable for validating service relationships, policies, and even monitoring functions, before real events or event storms occur. For more information about `zensendevent`, refer to the *Zenoss Resource Manager Administration Guide* Resource Manager Administration Guide.

7 Refine the service model.

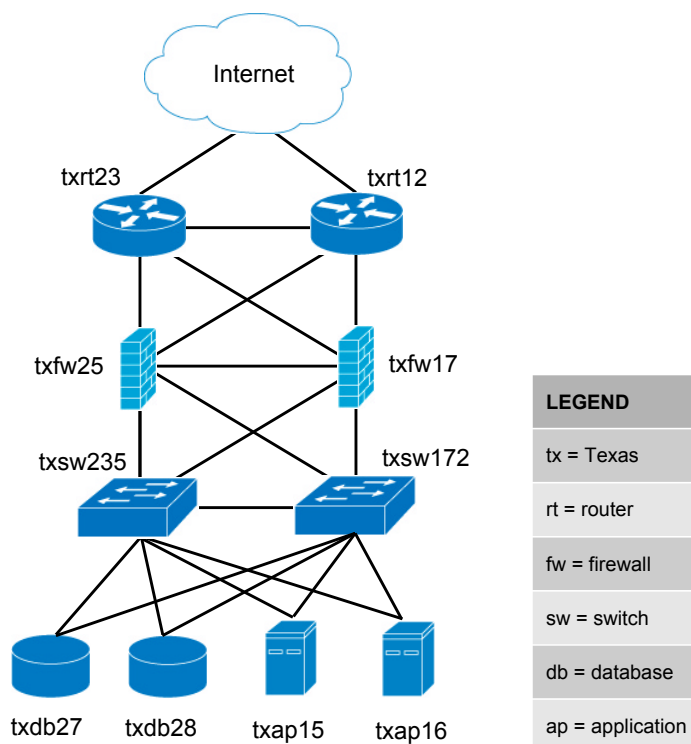
If a test or gap analysis reveals missing policies or service nodes, add them and test again.

Tutorial

2

This tutorial demonstrates how to create a service model for a simplified CRM application, how to view the model, and how to view and interpret the root cause analyses that Service Impact creates when events affect the application's availability state.

The following diagram shows the network topology of the CRM application featured in this tutorial.



Note The exercises in this tutorial are based on fake devices, which do not affect production environments. The final task erases the fake devices and the nodes created during the tutorial.

1. Define the service to model

The service to model is a CRM application, and it is defined by the resources on which it relies.

The CRM application relies on application and database hosts and processes, and on the network infrastructure that connects the service with its users. This tutorial creates a model of the development deployment of the CRM application, not the production or quality assurance deployments.

For this tutorial, all device and component resources are fake devices. [1.1. Fake device descriptions and roles](#) on page 17 describes the fake devices. The following procedure loads the fake devices into Resource Manager, and sets up the tutorial environment.

-
- 1 **Note** Skip this step if Service Impact is deployed with Resource Manager 4.2.x.
-

Gain access to the Resource Manager CLI environment in a Zope container.

- a Log in to the Control Center master host as a user with `serviced` CLI privileges.
- b Start an interactive session in a Zope container.

```
serviced service attach zope/0
```

- c In the new session, switch user to `zenoss`.

```
su - zenoss
```

- 2 **Note** Skip this step if Service Impact is deployed with Resource Manager 5.0.x.
-

Log in to the Resource Manager master host as `zenoss`.

- 3 Create a soft link to the Service Impact scripts directory.

Replace `VERSION` with the version number of the `ZenPacks.zenoss.Impact` ZenPack in your environment.

```
ln -s ${ZENHOME}/ZenPacks/ZenPacks.zenoss.Impact-VERSION-py2.7.egg\
/ZenPacks/zenoss/Impact/scripts/tutorial ${HOME}/impact_scripts
```

- 4 Change directory to the Service Impact scripts directory.

```
cd ${HOME}/impact_scripts
```

- 5 Load fake devices and components into Resource Manager.

```
zenbatchload --nomodel ./devices.txt
```

- 6 Check the values of the `ZENOSS_USERNAME` and `ZENOSS_PASSWORD` variables in `tutorial.sh`, and change them, if necessary.

The values must be the username and password of a Resource Manager user account.

- 7 Add execute permission to the `tutorial.sh` script.

```
chmod +x ./tutorial.sh
```

- 8 Set up the tutorial environment.

```
./tutorial.sh
```

- 9 Log in to the Resource Manager browser interface as a user with `ZenManager` or `Manager` privileges.

- 10 Click **SERVICES**.

The Service Impact feature of Resource Manager adds a tab named **SERVICES** to the Resource Manager menu bar.

- 11 In the tree view, open the **CRM - Development** organizer, and then open the **Application** and **Compute** organizers.

The tree view displays the service nodes created by the `tutorial.sh` script, as shown in the following example.

The screenshot shows the Zenoss Service Dynamics interface for the 'CRM - Development' service. The left sidebar shows a tree view with 'CRM - Development' selected. The main area displays two large green circles: one labeled 'UP' and one labeled 'ACCEPTABLE'. Below these are two tables showing service components and their status.

Service	Availability	Service	Performance
Apache Daemons	UP	Apache Daemons	ACCEPTABLE
App Hosts	UP	App Hosts	ACCEPTABLE
CRM - Application Service	UP	CRM - Application Service	ACCEPTABLE
CRM - Compute Service	UP	CRM - Compute Service	ACCEPTABLE
DB Hosts	UP	DB Hosts	ACCEPTABLE
Java Daemons	UP	Java Daemons	ACCEPTABLE
MySQL Daemons	UP	MySQL Daemons	ACCEPTABLE
txfw17 - Routers	UP	txfw17 - Routers	ACCEPTABLE
txfw25 - Routers	UP	txfw25 - Routers	ACCEPTABLE
txsw172 - App Hosts	UP	txsw172 - App Hosts	ACCEPTABLE
txsw172 - DB Hosts	UP	txsw172 - DB Hosts	ACCEPTABLE
txsw172 - Firewalls	UP	txsw172 - Firewalls	ACCEPTABLE
txsw235 - App Hosts	UP	txsw235 - App Hosts	ACCEPTABLE
txsw235 - DB Hosts	UP	txsw235 - DB Hosts	ACCEPTABLE
txsw235 - Firewalls	UP	txsw235 - Firewalls	ACCEPTABLE
zfake txfw17 - bxt12	UP	zfake txfw17 - bxt12	ACCEPTABLE
zfake txfw17 - bxt23	UP	zfake txfw17 - bxt23	ACCEPTABLE
zfake txfw25 - bxt12	UP	zfake txfw25 - bxt12	ACCEPTABLE

Related Links

[Service Impact home page](#) on page 36

The home page of the Service Impact feature of Zenoss Service Dynamics displays the health summaries of all services.

[Members](#) on page 37

The **Members** view provides details about a service node, including the list of nodes that are associated with it.

1.1. Fake device descriptions and roles

A description of each fake device used in the tutorial, and the role (Resource Manager device or component) each fake device plays.

Application host 15

Fake device name	Description	Role
fake-txap15	Application host 15	Device
fake-txap15-httpd	Apache daemon	Component
fake-txap15-java	Java/JRE daemon	Component
fake-txap15-nic-0	Network interface card 0	Component
fake-txap15-nic-1	Network interface card 1	Component

Application host 16

Fake device name	Description	Role
fake-txap16	Application host 16	Device
fake-txap16-httpd	Apache daemon	Component
fake-txap16-java	Java/JRE daemon	Component
fake-txap16-nic-0	Network interface card 0	Component
fake-txap16-nic-1	Network interface card 1	Component

Database host 27

Fake device name	Description	Role
fake-txdb27	Database host 27	Device
fake-txdb27-mysqld	MySQL daemon	Component
fake-txdb27-nic-0	Network interface card 0	Component
fake-txdb27-nic-1	Network interface card 1	Component

Database host 28

Fake device name	Description	Role
fake-txdb28	Database host 28	Device
fake-txdb28-mysqld	MySQL daemon	Component
fake-txdb28-nic-0	Network interface card 0	Component
fake-txdb28-nic-1	Network interface card 1	Component

Firewall 17

Fake device name	Description	Role
fake-txfw17	Firewall 17	Device
fake-txfw17-10g-0	Port 0 (10GB capacity)	Component
fake-txfw17-10g-1	Port 1 (10GB capacity)	Component
fake-txfw17-10g-2	Port 2 (10GB capacity)	Component
fake-txfw17-10g-3	Port 3 (10GB capacity)	Component
fake-txfw17-1g-0	Port 0 (1 GB capacity)	Component

Firewall 25

Fake device name	Description	Role
fake-txfw25	Firewall 25	Device
fake-txfw25-10g-0	Port 0 (10GB capacity)	Component
fake-txfw25-10g-1	Port 1 (10GB capacity)	Component

Fake device name	Description	Role
fake-txfw25-10g-2	Port 2 (10GB capacity)	Component
fake-txfw25-10g-3	Port 3 (10GB capacity)	Component
fake-txfw25-1g-0	Port 0 (1 GB capacity)	Component

Router 12

Fake device name	Description	Role
fake-txrt12	Router 12	Device
fake-txrt12-100g-0	Port 0 (100GB capacity)	Component
fake-txrt12-10g-0	Port 0 (10GB capacity)	Component
fake-txrt12-10g-1	Port 1 (10GB capacity)	Component
fake-txrt12-1g-0	Port 0 (1GB capacity)	Component

Router 23

Fake device name	Description	Role
fake-txrt23	Router 23	Device
fake-txrt23-100g-0	Port 0 (100GB capacity)	Component
fake-txrt23-10g-0	Port 0 (10GB capacity)	Component
fake-txrt23-10g-1	Port 1 (10GB capacity)	Component
fake-txrt23-1g-0	Port 0 (1GB capacity)	Component

Switch 172

Fake device name	Description	Role
fake-txsw172	Switch 172	Device
fake-txsw172-10g-0	Port 0 (10GB capacity)	Component
fake-txsw172-10g-1	Port 1 (10GB capacity)	Component
fake-txsw172-1g-0	Port 0 (1GB capacity)	Component
fake-txsw172-1g-1	Port 1 (1GB capacity)	Component
fake-txsw172-1g-2	Port 2 (1GB capacity)	Component
fake-txsw172-1g-3	Port 3 (1GB capacity)	Component
fake-txsw172-1g-4	Port 4 (1GB capacity)	Component

Switch 235

Fake device name	Description	Role
fake-txsw235	Switch 235	Device
fake-txsw235-10g-0	Port 0 (10GB capacity)	Component

Fake device name	Description	Role
fake-txsw235-10g-1	Port 1 (10GB capacity)	Component
fake-txsw235-1g-0	Port 0 (1GB capacity)	Component
fake-txsw235-1g-1	Port 1 (1GB capacity)	Component
fake-txsw235-1g-2	Port 2 (1GB capacity)	Component
fake-txsw235-1g-3	Port 3 (1GB capacity)	Component
fake-txsw235-1g-4	Port 4 (1GB capacity)	Component

1.2. Introduction to the tutorial environment

The `tutorial-setup.sh` script creates the following service nodes and organizers to initialize the CRM service model graph in Service Impact.

- The **Dashboard** organizer.

This root-level organizer is for service nodes that represent service models as a whole, a best practice. Initially, the organizer is empty.

- The root-level **CRM - Development** organizer, containing additional organizers.

Zenoss recommends using a single root-level organizer to contain the subservices of each service model, and using standardized names (and contents) for sub-organizers.

- The **CRM - Application Service** and **CRM - Compute Service** service nodes, children of the **CRM - Development** organizer.

These service nodes summarize the application and compute services associated with the CRM application, and are easily located without having to open the organizers in which their constituent subservices are located. This is a best practice.

- The service nodes in the **Network** organizer that start with **zfake** represent the network connections between fake devices.

Because these fake devices are not modeled, Resource Manager cannot discern their relationships, and Service Impact cannot create device or component nodes for us. So the setup script creates service nodes to represent the connections.

None of these nodes have either contextual or global policies, so the default policy applies: The state of the worst condition affecting child nodes becomes the state of the **zfake** service nodes, which is the correct policy for these connections.

- All of the DNS and interface names follow a naming convention, a best practice.
- The subservice nodes in the **Network** organizer that start with **tx** contain redundant resources, and have standardized, global availability policies defined.

These subservice nodes embody several best practices.

- Each subservice node contains homogeneous child nodes. Global policies work best when child nodes are homogeneous.
- Each subservice uses global policies. Global policies can be re-used across service model boundaries. Contextual policies are restricted to specific service models.
- Each global policy contains the following, standardized state triggers: The availability state is **ATRISK** if 50% or more child nodes are down, and **DOWN** if 100% of child nodes are down. By using percentage thresholds, the policies do not need adjustment if additional resources are deployed at a later date.

Note The standardized state triggers used in this case are not intrinsically best practices. Rather, they are examples of thinking about and using global policies systematically. For example, if a resource pool contains more than two members, additional state triggers may be defined.

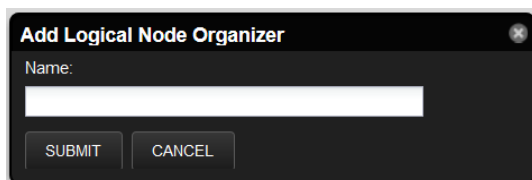
The remaining procedures in this tutorial demonstrate how to complete the CRM service model, and how to test it.

2. Create logical node and subservice nodes for internet connections

All of the network connections are modeled, except the connection to the internet. Since the internet is not modeled in Resource Manager, we create a logical node to represent it. For this tutorial, the logical node is simply an object that reacts to fake events. A "real" logical node could be configured to react to events from a `zencommand` that pings an internet resource.

2.1. Create logical node

- 1 In the Resource Manager browser interface, select **SERVICES > Logical Nodes**.
- 2 From the **Add** menu at the bottom of the tree view, select **Add Logical Node Organizer**.



- 3 In the **Add Logical Node Organizer** dialog, enter `CRM - Development`, and then click **SUBMIT**.
- 4 From the **Add** menu at the bottom of the tree view, select **Add Logical Node**.
- 5 In the **Add Logical Node** dialog, enter `example.com`, and then click **SUBMIT**.
- 6 In the `example.com` details view, enter values to match the following table, and then click **Save**.

Field	Description
Description	A node to help represent a route to the Internet.
Criteria	all, Summary, contains, fakeInternet
Events for this node in this event class	/Status/Ping
will result in these availability states	Critical: DOWN, Error: DOWN, Warning: ATRISK, Clear: UP

Related Links

[Service Impact home page](#) on page 36

The home page of the Service Impact feature of Zenoss Service Dynamics displays the health summaries of all services.

[Logical node details view](#) on page 44

The logical node details view allows you to create and edit logical nodes.

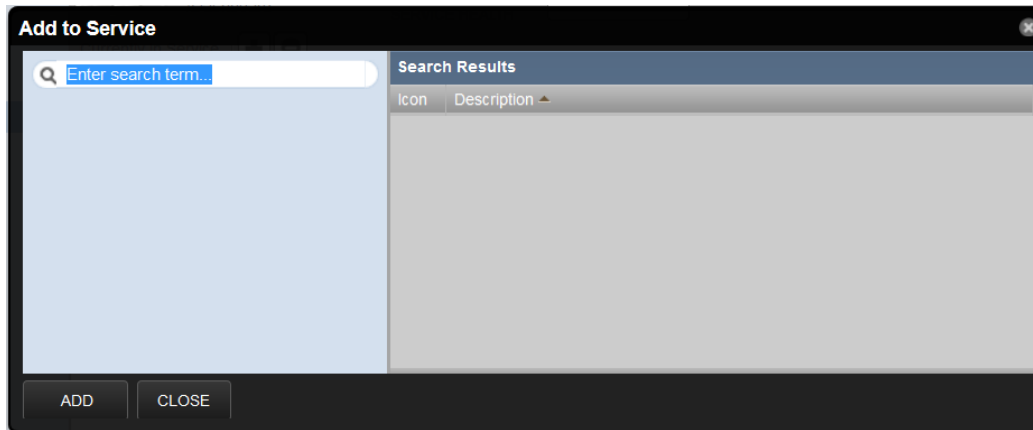
2.2. Create a subservice node for one internet connection

Create a service node for the connection from router 12 to the `zenoss.com` logical node.

- 1 In the Resource Manager browser interface, select **SERVICES > Dynamic Services**.
- 2 In the tree view, open the **CRM - Development** organizer, and then open the **Network** organizer.
- 3 In the tree view, select the **Network** organizer.

When an organizer is not selected, new service nodes are created at the root level. If that happens, simply drag the new service node into the correct organizer.

- 4 From the **Add** menu at the bottom of the tree view, select **Add Dynamic Service**.
- 5 In the **Add Dynamic Service** dialog, enter `txrt12 - Internet`, and then click **SUBMIT**.
- 6 In the **Overview** view, click the **Add** button.
- 7 In the **Add to Service** dialog, enter `fake-txrt12` in the search field.



Service Impact begins searching for matches after the first 3 characters are entered.

- 8 Select **Device** in the left column, and `fake-txrt12-100g-0` in the results list, then click **ADD**.
- 9 In the search field, enter `example.com`.
- 10 From the search results list, select the `example.com` logical node.
- 11 Click **ADD**, and then **CLOSE**.

Related Links

[Service Impact home page](#) on page 36

The home page of the Service Impact feature of Zenoss Service Dynamics displays the health summaries of all services.

[Add to Service dialog](#) on page 38

The interface for finding and adding nodes to a service node.

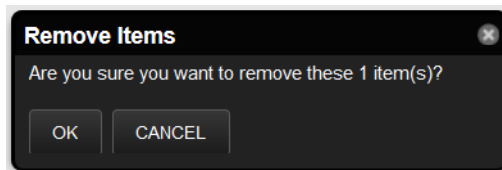
[Members](#) on page 37

The **Members** view provides details about a service node, including the list of nodes that are associated with it.

2.3. Create a subservice node for the other internet connection

Clone the service node for the connection through router 12 to create the service node for the connection through router 23.

- 1 In the tree view, select service node `txrt12 - Internet`.
- 2 From the **Action** menu at the bottom of the tree view, select **Clone Service....**
- 3 In the **Clone Service** dialog, enter `txrt23 - Internet`, and then click **SUBMIT**.
The new service is created and its contents are displayed in the **Overview** view.
- 4 From the list of nodes in the **Overview** view, select `fake-txrt12-100g-0`, and then click the remove button.
If you click on the name of the node, Resource Manager displays the device overview page. Click the browser's back button to return to the correct page.



- 5 In the **Remove Items** dialog, click **OK**.
- 6 In the **Overview** view, click the **Add** button.
- 7 In the **Add to Service** dialog, enter 100g in the search field.
- 8 In the left column, select **Device**. In the results list, select **fake-txrt23-100g-0**. Then click **ADD**, and **CLOSE**.

Note You may double-click an entry in the results list to add it to a service node.

Related Links

[Service Impact home page](#) on page 36

The home page of the Service Impact feature of Zenoss Service Dynamics displays the health summaries of all services.

[Add to Service dialog](#) on page 38

The interface for finding and adding nodes to a service node.

[Members](#) on page 37

The **Members** view provides details about a service node, including the list of nodes that are associated with it.

2.4. Create service node to represent redundant paths to the internet

The previous tasks created subservices to represent the two WAN connections to the internet. This task creates a subservice node to represent the redundant WAN tier.

- 1 In the tree view, select the **Network** organizer.
- 2 From the **Add** menu at the bottom of the tree view, select **Add Dynamic Service**.
- 3 In the **Add Dynamic Service** dialog, enter **Routers - Internet**, and then click **SUBMIT**.
- 4 In the **Overview** view, click the **Add** button.
- 5 In the **Add to Service** dialog, enter **Internet** in the search field.
- 6 In the left column, select **DynamicService**.
- 7 In the results list, select **txrt12-Internet** and **txrt23-Internet**, and then click **ADD** and **CLOSE**.

In the **Overview** view, the device names include their organizer paths relative to the root of the tree.

Related Links

[Service Impact home page](#) on page 36

The home page of the Service Impact feature of Zenoss Service Dynamics displays the health summaries of all services.

[Add to Service dialog](#) on page 38

The interface for finding and adding nodes to a service node.

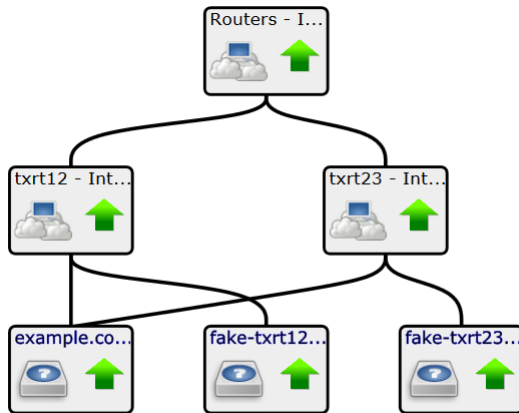
[Members](#) on page 37

The **Members** view provides details about a service node, including the list of nodes that are associated with it.

2.5. Add a policy to the internet connection node

Add a global availability policy to the internet connection subservice node.

- 1 In the tree view, select **Routers - Internet**.
- 2 In the **Overview** view, select **Impact View**.



- 3 In the service element widget of the **Routers - Internet** service (the widget at the top of the hierarchy), right-click to display the pop-up menu, and then select **Edit Impact Policies**.
- 4 In the **Impact Policies** dialog, click the **Add** button directly to the right of the **Global Policy** entry.



- 5 In the lower-left corner of the **Edit...Policy** tab, click the **Add** button, and then add the following triggers.
 - ATRISK if \geq 50% DynamicService is DOWN
 - DOWN if \geq 100% DynamicService is DOWN

Note When you select a type for a state trigger, the selection excludes other types. In this case, the only options are **Any** and **DynamicService**, and both child nodes are service nodes, so exclusivity is not a concern.

Related Links

[Service Impact home page](#) on page 36

The home page of the Service Impact feature of Zenoss Service Dynamics displays the health summaries of all services.

[Impact View](#) on page 39

The **Impact View** displays the interactive graph of a service node.

[Impact Policies dialog](#) on page 42

The **Impact Policies** dialog provides options for defining global and contextual policies, by creating or editing state triggers.

[Edit...Policy tab](#) on page 43

The **Edit...Policy** tab of the **Impact Policies** dialog is the interface for adding or editing state triggers for contextual or global policies.

3. Create service nodes for major network segments

All of the connections between devices are modeled, and all of the redundant resources are modeled. This procedure creates service nodes for the major network segments.

- 1 In the Resource Manager browser interface, select **SERVICES > Dynamic Services**.
- 2 In the tree view, open the **CRM - Development** organizer, and then open and select the **Network** organizer.
- 3 Create new service nodes for the major network segments.

The following table matches new and existing service nodes.

New service node	Existing service nodes
App Hosts - Switches	txsw235 - App Hosts, txsw172 - App Hosts
DB Hosts - Switches	txsw235 - DB Hosts, txsw172 - DB Hosts
Firewalls - Routers	txfw25 - Routers, txfw17 - Routers
Switches - Firewalls	txsw235 - Firewalls, txsw172 - Firewalls

Refer to the preceding steps for detailed instructions.

- 4 Add the following global policies (availability state triggers) to each new service node.
 - `ATRISK if >= 50% DynamicService is DOWN`
 - `DOWN if >= 100% DynamicService is DOWN`

Refer to the preceding steps for detailed instructions.

Related Links

[Service Impact home page](#) on page 36

The home page of the Service Impact feature of Zenoss Service Dynamics displays the health summaries of all services.

[Add to Service dialog](#) on page 38

The interface for finding and adding nodes to a service node.

[Members](#) on page 37

The **Members** view provides details about a service node, including the list of nodes that are associated with it.

4. Create summary nodes for critical paths

Services nodes for all of the major network segments are in place, so now it's time to consider nodes to represent the critical paths. The following, minimal conditions must be satisfied to characterize the CRM application as available.

- One connection between internet users and an application server is UP.
- One connection between an application server and a database server is UP.

If nodes for these paths already existed, we could create the service node that summarizes the network service for CRM. However, only one segment is defined, by the **Routers - Internet** subservice. Two additional subservices are required, for the following paths:

- the path between the routers and the application hosts
- the path between the application and database hosts

- 1 In the Resource Manager browser interface, select **SERVICES > Dynamic Services**.
- 2 In the tree view, open the **CRM - Development** organizer, and then open and select the **Network** organizer.

- 3 Create new service nodes for the network paths that affect users.

The following table matches new and existing service nodes.

New service node	Existing service nodes
Routers - App Hosts	App Hosts - Switches, Firewalls - Routers, Switches - Firewalls
App Hosts - DB Hosts	App Hosts - Switches, DB Hosts - Switches

Refer to the preceding steps for detailed instructions.

Note The correct policy for these new service nodes is the default policy, because each subservice is critical.

Related Links

[Service Impact home page](#) on page 36

The home page of the Service Impact feature of Zenoss Service Dynamics displays the health summaries of all services.

[Add to Service dialog](#) on page 38

The interface for finding and adding nodes to a service node.

[Members](#) on page 37

The **Members** view provides details about a service node, including the list of nodes that are associated with it.

5. Create a node for the network service

The critical network paths are defined; now, create a service node to represent the network service for the CRM application.

- 1 In the Resource Manager browser interface, select **SERVICES > Dynamic Services**.
- 2 In the tree view, open the **CRM - Development** organizer, and then select it.
Service nodes that represent a category should be peers of their sub-organizers.
- 3 Create a new service node, named `CRM - Network Service`.
- 4 Add the following subservice nodes to the new service node.
 - App Hosts - DB Hosts
 - Routers - App Hosts
 - Routers - Internet

The correct policy for this node is the default policy.

Related Links

[Service Impact home page](#) on page 36

The home page of the Service Impact feature of Zenoss Service Dynamics displays the health summaries of all services.

[Add to Service dialog](#) on page 38

The interface for finding and adding nodes to a service node.

[Members](#) on page 37

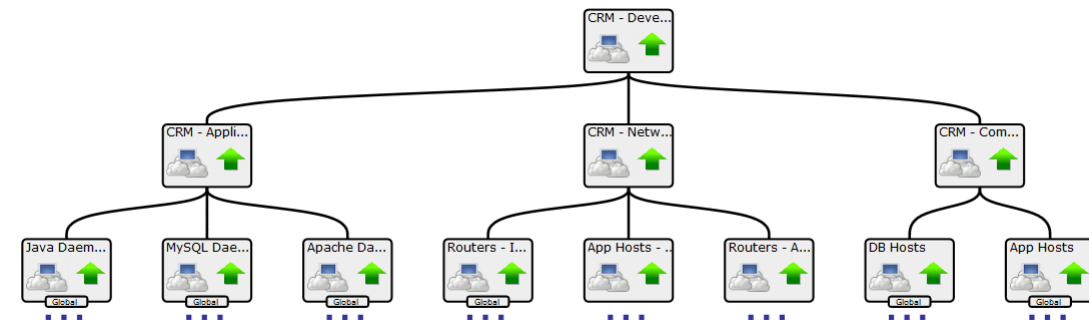
The **Members** view provides details about a service node, including the list of nodes that are associated with it.

6. Create a service node for the service model

With all of the resource categories modeled, the application as a whole can be modeled.

- 1 In the Resource Manager browser interface, select **SERVICES > Dynamic Services**.
- 2 In the tree view, select the **Dashboard** organizer.
Service nodes that represent a service model as a whole should be kept in a separate, root-level organizer. This way, you can quickly determine the state of all service models in the environment.
- 3 Create a service node named `CRM - Development Service` and add the following subservice nodes to it.
 - CRM - Application Service
 - CRM - Compute Service
 - CRM - Network Service

To view the model, select the new service in the tree view, and then select **Impact View**. After hiding child nodes and zooming in, the graph looks similar to the following image.



If necessary, click and drag the graph to center it.

Related Links

[Service Impact home page](#) on page 36

The home page of the Service Impact feature of Zenoss Service Dynamics displays the health summaries of all services.

[Add to Service dialog](#) on page 38

The interface for finding and adding nodes to a service node.

[Members](#) on page 37

The **Members** view provides details about a service node, including the list of nodes that are associated with it.

7. Send events to fake devices

Send events to the fake devices to see how they affect the availability of the CRM application.

- 1 In the tree view, open the **Dashboard** organizer, and then select **CRM - Development Service**.
- 2 In the main view area, select **Impact View**.
- 3 **Note** Skip this step if Service Impact is deployed with Resource Manager 4.2.x.

Gain access to the Resource Manager CLI environment in the ZenHub container.

- a Log in to the Control Center master host as a user with `serviced` CLI privileges.
- b Start an interactive session in a Zope container.

```
serviced service attach zenhub
```

- c In the new session, switch user to `zenoss`.

```
su - zenoss
```

- 4 **Note** Skip this step if Service Impact is deployed with Resource Manager 5.0.x.

Log in to the Resource Manager master host as `zenoss`.

- 5 Send ping down events to the fake network interface card components in the `txap15` and `txap16` hosts.

```
zensendevent -d fake-txap15-nic-0 -c /Status/Ping -s Critical \
  "Impact Tutorial - fake device is DOWN"
zensendevent -d fake-txap15-nic-1 -c /Status/Ping -s Critical \
  "Impact Tutorial - fake device is DOWN"
zensendevent -d fake-txap16-nic-0 -c /Status/Ping -s Critical \
  "Impact Tutorial - fake device is DOWN"
```

- 6 In the browser, click the **Refresh** button.

The **Impact View** shows the CRM application availability state changed to **ATRISK**, and the nodes involved in the state change highlighted with yellow and red.

To see all of the nodes in the service model graph, click **Expand All**.

- 7 Change the view to **Impact Events**, and then click **CRM - Development Service** in the **Impact Events** list.

Status	Severity	Service	Event Class	Summary	First Seen	Last Seen	Count	
⚠		CRM - Develo...	/Service/State/Availability	Service Dashboard/CRM - Development Service is ATRISK.	2013-10-24 12:37:05	2013-10-24 12:37:05	1	
+	38		fake-txap16-n...	/Status/Pi...	Impact Tutorial - fake device is DOWN	2013-10-24 12:37:05	2013-10-24 12:37:05	1
+	38		fake-txap16-n...	/Status/Pi...	Impact Tutorial - fake device is DOWN	2013-10-24 12:37:04	2013-10-24 12:37:04	1
+	26		fake-txap15-n...	/Status/Pi...	Impact Tutorial - fake device is DOWN	2013-10-24 12:37:04	2013-10-24 12:37:04	1

The events that contribute to the current state of the CRM application are weighted by the root cause analysis that Service Impact performs (the **Confidence** column). To view the impact chain of an event, click the plus button in the left column.

- 8 Send ping down events to the fake network interface card components in the `txdb27` and `txdb28` hosts, and then click the **Refresh** button.

```
zensendevent -d fake-txdb27-nic-0 -c /Status/Ping -s Critical \
  "Impact Tutorial - fake device is DOWN"
zensendevent -d fake-txdb27-nic-1 -c /Status/Ping -s Critical \
  "Impact Tutorial - fake device is DOWN"
zensendevent -d fake-txdb28-nic-0 -c /Status/Ping -s Critical \
  "Impact Tutorial - fake device is DOWN"
```

The list of contributing events grows, and the rankings change to reflect the added events.

- 9 Send clear events to all of the fake devices that are down.

```
zensendevent -d fake-txap15-nic-0 -c /Status/Ping -s Clear \
  "Impact Tutorial - fake device is UP"
zensendevent -d fake-txap15-nic-1 -c /Status/Ping -s Clear \
  "Impact Tutorial - fake device is UP"
zensendevent -d fake-txap16-nic-0 -c /Status/Ping -s Clear \
  "Impact Tutorial - fake device is UP"
zensendevent -d fake-txdb27-nic-0 -c /Status/Ping -s Clear \
  "Impact Tutorial - fake device is UP"
zensendevent -d fake-txdb27-nic-1 -c /Status/Ping -s Clear \
```

```
"Impact Tutorial - fake device is UP"
zensendevent -d fake-txdb28-nic-0 -c /Status/Ping -s Clear \
"Impact Tutorial - fake device is UP"
```

Related Links

[Service Impact home page](#) on page 36

The home page of the Service Impact feature of Zenoss Service Dynamics displays the health summaries of all services.

[Impact Events](#) on page 38

The **Impact Events** view shows summary and detail information about events that are affecting a service node.

8. Remove tutorial service elements

This step removes all of the tutorial-defined nodes.

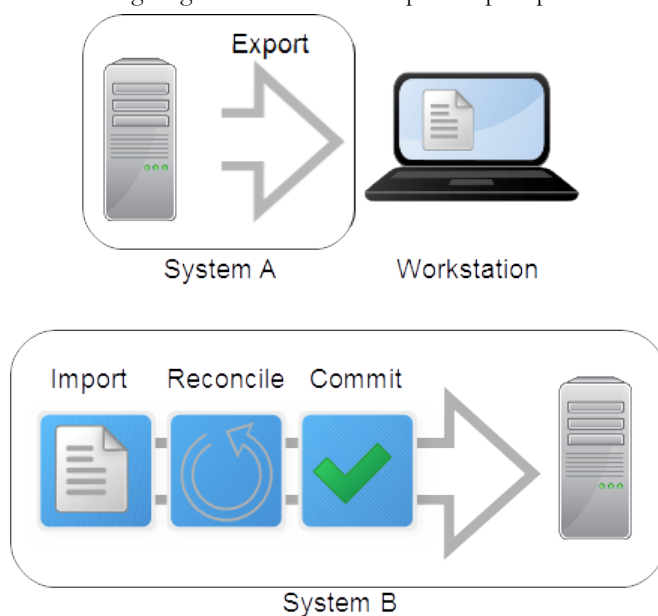
- 1 Log in to the Resource Manager browser interface as a user with ZenManager or Manager privileges.
- 2 Click **SERVICES**.
- 3 Remove the dynamic services and logical node.
 - a In the tree view, select the **Dashboard** organizer.
 - b At the bottom of the tree view, click the **Delete** button.
 - c In the tree view, select the **CRM - Development** organizer.
 - d At the bottom of the tree view, click the **Delete** button.
 - e Select the **Logical Nodes** view mode.
 - f In the tree view, select the **CRM - Development** organizer.
 - g At the bottom of the tree view, click the **Delete** button.
- 4 Remove the fake devices.
 - a In the Resource Manager browser interface, select **INFRASTRUCTURE**.
 - b In the tree view, select **FakeImpactDevices**.
 - c At the bottom of the tree view, click the **Delete** button.

Exporting and importing service models

You may export some or all of the service models or organizers created on one Service Impact system and import them on another.

- The export process creates a *GraphML 1.1* XML file that details the portion of your environment selected for export. You may edit the file, or use it as a template to generate objects programmatically.
- The import process features an initial import, a reconciliation phase, and a final commit. The reconciliation phase allows you to match devices or components in the originating system with their equivalents in the target system.

The following diagram illustrates the export-import processes.



Export process

From the Resource Manager browser interface, you select the service model, service organizer, or the entire dynamic services environment for export. The export file is generated, and your browser prompts you to save it on your workstation.

You may edit the file to define additional Service Impact service models, organizers, logical nodes, global and contextual policies, and custom state providers, before copying it to the target Resource Manager system for import. However, you can not define new Resource Manager devices or components.

Initial import

The initial import attempts to reconcile the originating and target Service Impact environments, and produces the following, additional files:

File.latest.txt

A three-column text file containing directives and specifications for the next reconciliation attempt. Each subsequent reconciliation attempt generates a new version of this file.

File.0001.txt

A record of the initial reconciliation attempt, which is used to diagnose process defects. Subsequent reconciliation attempts generate new, sequentially numbered record files. Zenoss recommends not modifying these files.

Reconciliation phase

Each reconciliation attempt tries to match devices or components from the originating system with their equivalents in the target system. The *File.latest.txt* file identifies the nodes that can not be reconciled, and provides both information and commands to enable you resolve each node. You may attempt to reconcile the two environments as many times as you wish before making the final commit.

Final commit

You are ready to perform the final commit when no UNRECONCILED entries are present in *File.latest.txt*. A final commit can not be reversed, but a new export-import process may be initiated.

Exporting objects from a Service Impact environment

- 1 In the Resource Manager browser interface, select the **SERVICES** tab.
- 2 In the tree view, select the objects to export.
You may select a service node, an organizer, or the entire **Dynamic Services** hierarchy.
- 3 From the **Action** menu at the bottom of the tree view, select **Export Selected**.
Resource Manager generates a GraphML file of the selected service graph, and sends it to your browser.
- 4 Save the file on your workstation.
The generated file name includes four fields, separated by the low line character (`_`). The file name extension is `graphml`.

export

The name of the action performed.

impact

The name of the Resource Manager plugin performing the action.

Object

The name of the object selected for export.

YYYYMMDDHHMMSS

The timestamp of the export action.

For example, `export_impact_DynamicServices_20150331135219.graphml`.

Importing a service graph

Perform this procedure to start the process of importing a GraphML file originating on one Service Impact system into another system.

- 1 **Note** Skip this step if Service Impact is deployed with Resource Manager 4.2.x.

Copy the GraphML file, and then gain access to the Resource Manager CLI environment on the target system, in a Zope container.

- a Copy the GraphML file from the originating Service Impact system to a local directory on the Control Center master host.

The directory containing the GraphML file must a local directory (not mounted) and must be readable, writable, and executable by all users. The following commands create a directory in /tmp:

```
mkdir /tmp/impact && chmod 777 /tmp/impact
```

- b Log in to the Control Center master host as a user with serviced CLI privileges.
c Start an interactive session in a Zope container.

```
serviced service attach zope/0
```

- d In the new session, switch user to zenoss.

```
su - zenoss
```

- 2 **Note** Skip this step if Service Impact is deployed with Resource Manager 5.0.x.

Copy the GraphML file, and then log in to the Resource Manager master host.

- a Copy the GraphML file from the originating Service Impact system to a directory on the Resource Manager master host.
b Log in to the Resource Manager master host as zenoss.

- 3 Initiate the import.

For convenience, the name of the file to import is truncated to *File.graphml*.

```
zenimportimport -i File.graphml
```

The `zenimportimport` command attempts to reconcile the originating and target Service Impact environments, and generates the following files:

- *File.latest.txt*
- *File.0001.txt*

Edit *File.latest.txt* as required to prepare for the next reconciliation attempt. For more information about the file, see [Reconciliation file](#) on page 32.

Reconciliation file

The initial import attempt and each reconciliation attempt creates a new version of the *File.latest.txt* file. The file contains the following three columns:

[ACTION]

Actions to perform in the next reconciliation attempt. For more information, see the following table.

[IMPORT NODE]

The ID of the imported node. In most cases, the ID should not be modified.

[OPERATIONAL NODE]

The DMD ID, element name, or GUID of the Resource Manager element to apply to the imported node. This column only applies to MAP and DELETE actions.

Action	Description
UNRECONCILED	Could not find a matching device in Resource Manager.
MAP	Found a matching element in Resource Manager.
CREATE	Could not find a matching element in Resource Manager, but it can be created through import.
DELETE	Found a matching element in Resource Manager and it is marked to be deleted in the import file.
IGNORE	Do nothing with the imported node.

Reconciliation attempts

Perform this procedure after editing the `File.latest.txt` file generated in previous reconciliation attempts. You may perform this procedure as often as you wish. For faster progress, Zenoss recommends making multiple, small changes and reconciliation attempts.

- Note** Skip this step if Service Impact is deployed with Resource Manager 4.2.x.

Gain access to the Resource Manager CLI environment on the target system, in a Zope container.

- Log in to the Control Center master host as a user with `serviced` CLI privileges.
- Start an interactive session in a Zope container.

```
serviced service attach zope/0
```

- In the new session, switch user to `zenoss`.

```
su - zenoss
```

- Note** Skip this step if Service Impact is deployed with Resource Manager 5.0.x.

Copy the GraphML file, and then log in to the Resource Manager master host.

- Copy the GraphML file from the originating Service Impact system to a directory on the Resource Manager master host.
- Log in to the Resource Manager master host as `zenoss`.

- Attempt to reconcile the import.

For convenience, the name of the file to import is truncated to `File.graphml`.

- Change directory to the directory that contains `File.latest.txt` and the record files (`File.nnnn.txt`).
- Initiate the reconciliation attempt.

```
zenimpactimport -r File.graphml
```

The `zenimpactimport` command attempts to reconcile the originating and target Service Impact environments, and generates the following files:

- `File.latest.txt`
- `File.nnnn.txt`

Edit `File.latest.txt` and repeat this procedure until no UNRECONCILED entries are present in `File.latest.txt`.

To delete the import from the target system without completing the reconciliation phase, enter the following command:

```
zenimpactimport -r File.graphml --abort
```

Performing the final commit

Perform this procedure when no UNRECONCILED entries are present in `File.latest.txt`.

Note A final commit can not be reversed. If you realize that you need to make additional changes after committing, you may not edit `File.latest.txt`; you must start a new export/import process.

1 **Note** Skip this step if Service Impact is deployed with Resource Manager 4.2.x.

Gain access to the Resource Manager CLI environment on the target system, in a Zope container.

- a Log in to the Control Center master host as a user with `serviced` CLI privileges.
- b Start an interactive session in a Zope container.

```
serviced service attach zope/0
```

- c In the new session, switch user to `zenoss`.

```
su - zenoss
```

2 **Note** Skip this step if Service Impact is deployed with Resource Manager 5.0.x.

Copy the GraphML file, and then log in to the Resource Manager master host.

- a Copy the GraphML file from the originating Service Impact system to a directory on the Resource Manager master host.
- b Log in to the Resource Manager master host as `zenoss`.

3 Commit the import.

For convenience, the name of the file to import is truncated to `File.graphml`.

- a Change directory to the directory that contains `File.latest.txt` and the record files (`File.nnnn.txt`).
- b Initiate the commit.

```
zenimpactimport -r File.graphml --commit
```

On completion, the service models, organizers, logical nodes, global and contextual policies, and custom state providers defined in the import file are added to the target system.

zenimpactimport

NAME

`zenimpactimport` - imports GraphML files containing Service Impact service graph definitions

SYNTAX

```
zenimpactimport -i FILE.graphml
```

```
zenimpactimport -r FILE.graphml {--abort|--commit}
```

```
zenimpactimport --list
```

DESCRIPTION

The `zenimpactimport` command imports *GraphML* files containing Service Impact service graph definitions. The import process includes an initial import, multiple reconciliation attempts, and a final commit.

OPTIONS

-i *FILE*.graphml

Perform the initial import and reconciliation attempt.

-r *FILE*.graphml {--abort|--commit}

Reconcile, abort, or commit the import.

--list

Display a list of all active imports.

4

Service Impact interface resources

This chapter describes the resources that are available on the **SERVICES** tab of the Resource Manager browser interface.

Service Impact home page

The home page of the Service Impact feature of Zenoss Service Dynamics displays the health summaries of all services.

From the home page, you can view information in two modes: **Dynamic Services** and **Logical Nodes**. In both view modes, the Service Impact home page includes the tree view area and its tools, and the primary view area.




The screenshot shows the Zenoss Service Impact interface in 'View mode'. The top navigation bar includes 'DYNAMIC SERVICES' and 'Logical Nodes'. The left sidebar shows a tree view of services under 'DYNAMIC SERVICES', including 'CRM - Development' and 'Dashboard'. The main area displays two large green circles representing service health: 'UP' and 'ACCEPTABLE'. Below this is a table of service health data.

Service	Availability	Performance
Apache Daemons	UP	ACCEPTABLE
App Hosts	UP	ACCEPTABLE
CRM - Application Service	UP	ACCEPTABLE
CRM - Compute Service	UP	ACCEPTABLE
DB Hosts	UP	ACCEPTABLE
Java Daemons	UP	ACCEPTABLE
MySQL Daemons	UP	ACCEPTABLE
trfw17 - Routers	UP	ACCEPTABLE
trfw25 - Routers	UP	ACCEPTABLE
bxsw172 - App Hosts	UP	ACCEPTABLE
bxsw172 - DB Hosts	UP	ACCEPTABLE
bxsw172 - Firewalls	UP	ACCEPTABLE
bxsw235 - App Hosts	UP	ACCEPTABLE
bxsw235 - DB Hosts	UP	ACCEPTABLE
bxsw235 - Firewalls	UP	ACCEPTABLE
zfsake.trchv17 - bvt12	UP	ACCEPTABLE
zfsake.trchv17 - bvt23	UP	ACCEPTABLE
zfsake.trchv25 - bvt12	UP	ACCEPTABLE

At the bottom of the table, it says 'DISPLAYING 1 - 18 of 31 ROWS'. The interface also includes 'Tree view tools' at the bottom left and a 'Refresh' button at the top right.

Tree view area

The tree view area displays service nodes and logical nodes in alphabetical order. You may create organizers and order them as you wish in both view modes. In addition, you may move service nodes and logical nodes into organizers by dragging them in the tree view.

Tool	Menu	
	Dynamic Services view mode	Logical Nodes view mode
	Add Dynamic Service Add Dynamic Service Organizer	Add Logical Node Add Logical Node Organizer
	(no menu)	(no menu)
	View and Edit Details Clone Service... Export Selected	(no menu)

Primary view area

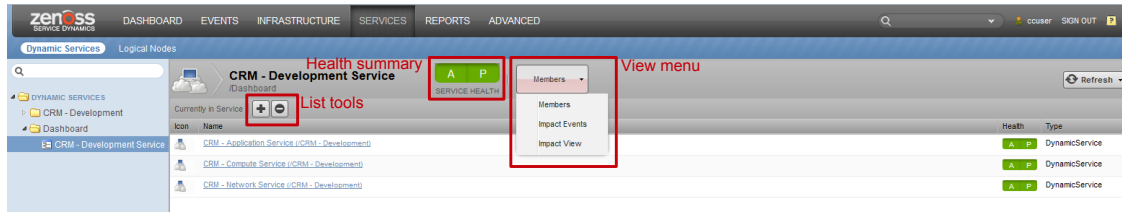
The contents of the primary view area depend on the view mode and the item selected in the tree view area.

View mode	Tree view selection	Primary view contents
Dynamic Services	DYNAMIC SERVICES (the root organizer)	The availability and performance health summaries of all services.
	An organizer	The availability and performance health summaries of the services contained in the organizer.
	A dynamic service	The Overview of the selected service. From this view, you may select Impact Events or Impact View .
Logical Nodes	<ul style="list-style-type: none"> ■ LOGICAL NODES (the root organizer) ■ An organizer 	A blank logical node details view.
	A logical node	The details view of the selected logical node. There are no other views associated with logical nodes.

Members



The **Members** view provides details about a service node, including the list of nodes that are associated with it.

The following example shows the **Members** view of a service node, with key features highlighted, and the view menu selected.



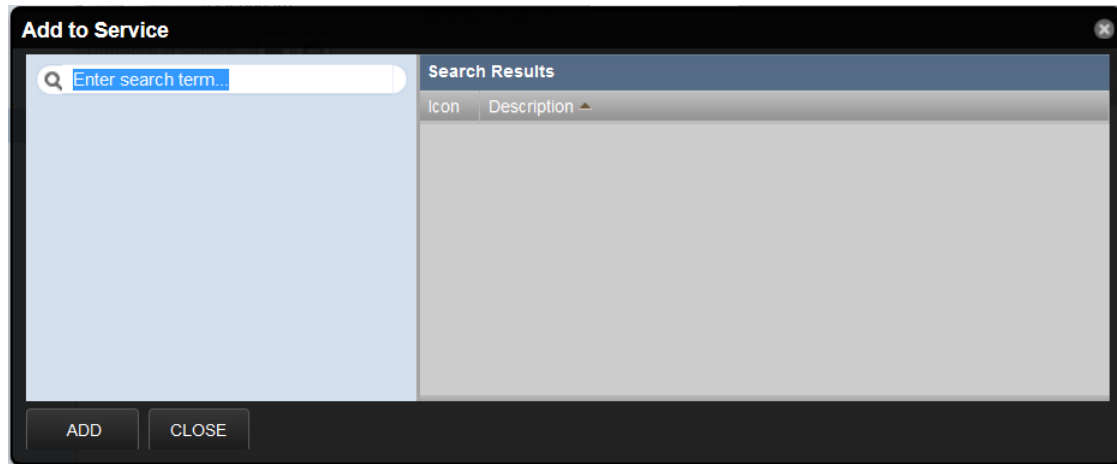
List tools

The list tools add or remove nodes from a service node.

Tool	Function
	Display the Add to Service dialog.
	Remove a selected node from the service node.

Add to Service dialog

The interface for finding and adding nodes to a service node.



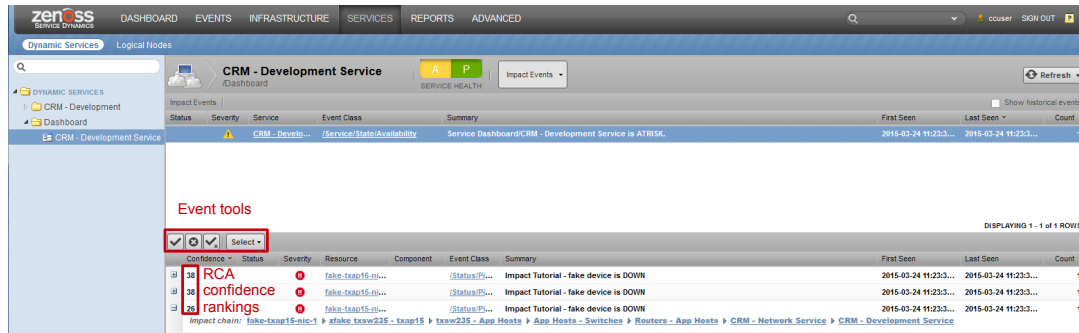
Enter the name of the node to find in the search field. After 3 characters, Service Impact displays matching nodes.

Impact Events

The **Impact Events** view shows summary and detail information about events that are affecting a service node.





The following example shows an **Impact Events** view, with key features highlighted.

Note The last event in the details area is expanded, showing the **Impact chain**, which is the hierarchy of nodes associated with the event.



Event tools

The event tools provide options for manipulating the list of events.

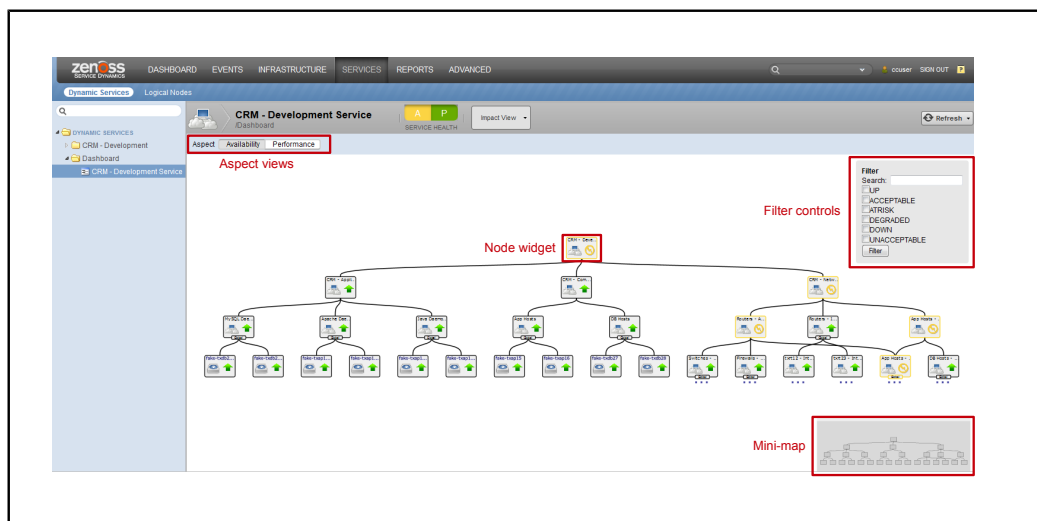
Tool	Function
	Acknowledge the selected event.
	Close the selected event. The event is moved to the archive.
	Undo acknowledgement of the selected event.
	Display the event selection menu. Select All Select all events in the list. None Deselect all of the selected events.

Root cause analysis (RCA) confidence rankings

The second column of the details list contains the event's confidence ranking. Service Impact knows which nodes affect which service nodes, and automatically performs root cause analysis when an event occurs. The analysis yields a probability value that an event is the cause of the service node's current state. Often, more than one event contributes to a state, so the confidence rankings allow you to focus resources on the most likely cause right away.

Impact View

The **Impact View** displays the interactive graph of a service node.



Aspect views

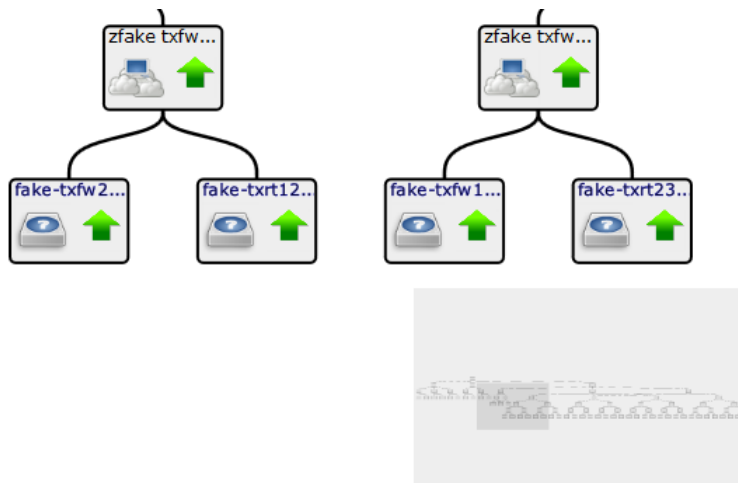
View	Function
Availability	Display availability states in the graph.
Performance	Display performance states in the graph.

Filter controls

Control	Function
Search	Specify a string to select nodes that contain the string in their names. Strings are case-sensitive; regular expressions are not supported. When no string is specified (the default) all nodes are selected..
UP	In the Availability view of the service graph, select nodes whose availability state is UP.
ACCEPTABLE	In the Performance view of the service graph, select nodes whose performance state is ACCEPTABLE.
ATRISK	In the Availability view of the service graph, select nodes whose availability state is ATRISK.
DEGRADED	Select nodes whose availability or performance state is DEGRADED.
DOWN	In the Availability view of the service graph, select nodes whose availability state is DOWN.
UNACCEPTABLE	In the Performance view of the service graph, select nodes whose performance state is UNACCEPTABLE.
Filter	Click Filter to update the service graph with the selected criteria.

Mini-map



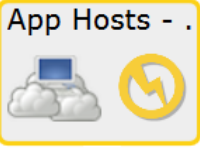
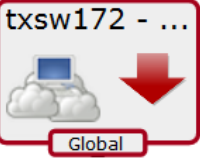
When you zoom in to a portion of a service graph, the mini-map includes a smaller, darker rectangle, which shows the relative position of the current display area in relation to the overall graph.



Additional tools

Item	Function
<div data-bbox="272 808 444 957"> <ul style="list-style-type: none"> Fit Graph To Window Show All Collapse All Toggle Rainbows Toggle Filter </div>	<p>When the pointer is in the Impact View window, right-click displays this menu.</p> <p>Fit Graph to Window Adjust the size of the service graph so that its dimensions match the display area.</p> <p>Show All Display all of the nodes in the service graph.</p> <p>Collapse All Display only the top-level node of the service graph.</p> <p>Toggle Rainbows Display colored tabs at the right edge of the node, indicating the critical, error, and warning events associated with the node.</p> <p>Toggle Filter Hide or display the filter controls.</p>
<div data-bbox="272 1346 435 1409"> <ul style="list-style-type: none"> Toggle Children Edit Impact Policies </div>	<p>When the pointer is positioned over a node, right-click displays this menu.</p> <p>Toggle Children Hide or display the node's child and descendent nodes.</p> <p>Edit Impact Policies Display the Impact Policies dialog.</p>
<div data-bbox="272 1556 548 1661"> <ul style="list-style-type: none"> Name CRM - Compute Service Availability UP Performance ACCEPTABLE Production true Type DynamicService </div>	<p>When you position the pointer over a node, Resource Manager displays a summary of the node.</p>

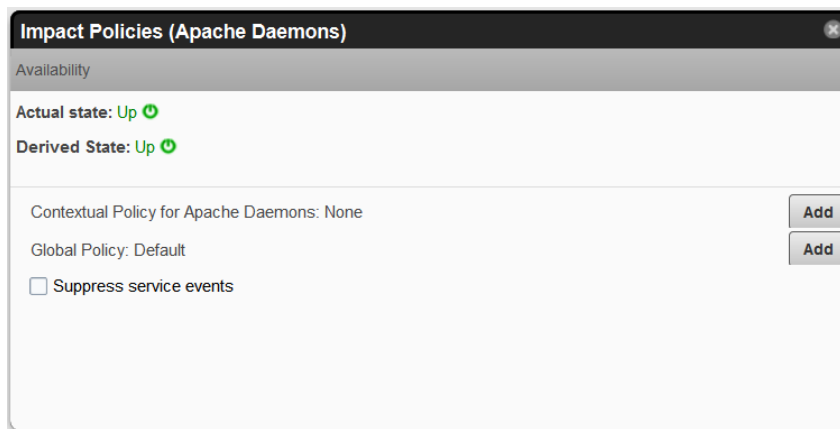
Node widget

Item	Description
	<p>This widget represents a Linux server named <code>epowell-debug.zenoss.loc</code>. (The name is shortened to fit inside the widget.)</p> <ul style="list-style-type: none"> ▪ The computer icon and penguin image represent the node's type, Linux server. ▪ The green arrow and the black border represent the node's availability state, UP.
	<p>This widget represents a service node with a global policy. In addition, event rainbows are displayed.</p> <ul style="list-style-type: none"> ▪ The event rainbow is the 3 colored tabs on the right side of the widget, showing (from top to bottom) the counts of critical, error, and warning events associated with the node. Currently, this node has no events—no numbers are displayed. ▪ When you click an event rainbow tab, Resource Manager displays the overview page of the node. ▪ This widget includes three dots immediately below its bottom border. The dots represent descendant nodes. To display the descendant nodes, double-click the node widget.
	<p>This widget represents a service node. The yellow icon and border represent the node's availability state, ATRISK.</p>
	<p>This widget represents a service node with a global policy. The red arrow and the red border of the widget represent the node's availability state, DOWN.</p>

Impact Policies dialog

The **Impact Policies** dialog provides options for defining global and contextual policies, by creating or editing state triggers.

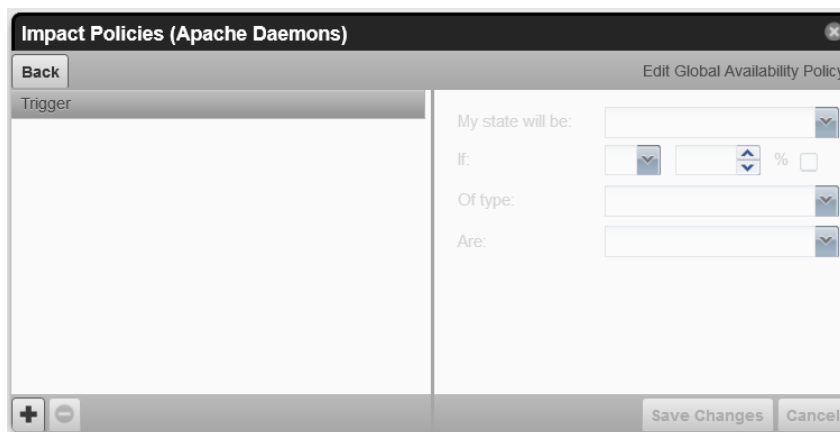
Use the **Impact Policies** dialog to add or edit state triggers for contextual or global policies. If a custom state provider is associated with a node, an additional option is displayed in this dialog as well.



Check the **Suppress service events** box to prevent sending service events when changes affect a node. Choose this option when a service node is used solely to group child nodes.

Edit...Policy tab

The **Edit...Policy** tab of the **Impact Policies** dialog is the interface for adding or editing state triggers for contextual or global policies.



To add a trigger, click the **Add** button in the lower-left corner, modify the trigger fields on the right side, and then click the **Save Changes** button.

Field	Description
My state will be	The new state, if the trigger applies. Selections are DOWN, DEGRADED, ATRISK, and UP.
If	The conditions that trigger a state change. To specify a percentage rather than an absolute value, click the percent (%) check box.
Of type	Restrict the type of child node to which the trigger applies. Types other than Any are exclusive.
Are	The state of child nodes that cause an evaluation of this trigger. Selections are DOWN, DEGRADED, ATRISK, and UP.

Edit Custom...State Provider

The **Custom...State Provider** tab of the **Impact Policies** dialog is the interface for adding or editing state triggers for custom device and component objects.

In the **Event Class** field, specify the Resource Manager event class to monitor.

In the event severity fields, specify the state for this node, if the event severity is observed. Selections are DOWN, DEGRADED, ATRISK, and UP.

In the **Apply to** field, specify the nodes to which the state applies.

Logical node details view

The logical node details view allows you to create and edit logical nodes.

A logical node is a customizable object that captures specific Resource Manager event states. Logical nodes allow you to represent resources or services not monitored by Resource Manager, or, for resources that are monitored by Resource Manager, to capture event states from arbitrary classes.

A logical node may be a child of a service node, but no other node type.

Criteria

The **Criteria** rules allow you to define event triggers, and to associate the triggers with availability and performance states.

Availability State

Use the **Events for this node in this event class** field to specify the event class or subclass associated with the trigger.

The following examples illustrate how to specify event classes.

/Status

Only the /Status class.

/Status/Web

Only the Web subclass of /Status

/Status/

The /Status class and all of its subclasses.

Use the **will result in these availability states** fields to map event severity levels to availability states.

Performance State

The fields and options in this area of the page differ only in that the mapping of event severity levels to performance states uses different states.

Configuring Service Impact

This chapter describes the configuration options available in Service Impact.

Production state propagation threshold

Service Impact relies on device production states to decide whether to propagate availability and performance states within a service graph. The following table characterizes the device production states that are available in Resource Manager.

Production State	Devices Monitored?	Appear on Dashboard?
Production	yes	yes
Pre-Production	yes	no
Test	yes	no
Maintenance	yes	may appear
Decommissioned	no	no

For more information about production states, refer to the *Zenoss Resource Manager Administration Guide*.

The default value of the Service Impact production state propagation threshold is Production. You may select a different production state for the threshold.

Configuring the production state propagation threshold

Zenoss recommends changing this threshold only on installation.

- 1 Log in to the Resource Manager browser interface as a user with ZenManager or Manager privileges.
- 2 Select the **ADVANCED** tab, and then click the **Impact** link in the left column.



- From the **Production State Propagation Threshold** list, select a new production state, and then click **Save**. Devices are evaluated against the new threshold at the next modelling interval. To update all devices before the next modelling interval, perform a graph update.

Service Impact server configuration files

The `$IMPACT_HOME/etc` directory contains Service Impact server configuration files.

`$IMPACT_HOME/etc/zenoss-dsa-amqpconf.properties`

This file contains properties for the Service Impact host's connection to the RabbitMQ Server.

`$IMPACT_HOME/etc/zenoss-dsa.env`

This file contains the `JVM_ARGS` variable definition.

`$IMPACT_HOME/etc/zenoss-dsa.properties`

This file contains general Service Impact configuration properties, including properties for backups, log file management, and remote debugging.

Note Configuration settings and their associated default values may appear commented out in the file; however, they are automatically activated at installation. To modify the default value, you must first uncomment the setting. Before you change the values in the preceding files, or any other file in this directory, please consult with Zenoss Support.

Configuration file best practices

Zenoss recommends that you edit configurations through the Control Center user interface. Using this method, edits are preserved and pushed to the Impact container when restarted. When Impact restarts, a new container is created. Any changes applied using the command are lost when Impact is restarted.

Use the Control Center user interface to edit and preserve changes to `/opt/zenoss_impact/etc/logback.xml`. Modifying `logback.xml` in the Control Center user interface requires a restart of the Impact service for the logging changes to take effect. If you use the command line to implement logging changes, they are implemented immediately; however, the changes made using the command line are lost when Impact is restarted.

The following sections describe how to edit Service Impact configuration files for Resource Manager 5.0.x and 4.2.x.

Editing server configuration files with Control Center (RM 5.0.x)

To test changes in a pre-production environment, modify configuration file settings directly in the container. When Service Impact is restarted, all the settings are returned to their default value.

This procedure describes how to use Control Center to edit Service Impact server configuration files with Resource Manager 5.0.x.

- Log in to Control Center as a user with root privileges.
- In the **Applications** table, click `Zenoss.resmgr`.
- Scroll down to the **Services** list and select **Impact**.
- Under **Configuration Files**, click **Edit** to the right of `/opt/zenoss/etc/dsa.zenoss-dsa.properties`.

The `zenoss-dsa.properties` file is displayed in the **Edit Configuration** window.

Note Modifying properties may change the performance of Service Impact. Please consult Zenoss Support prior to modifying default values.

- Make changes as desired or directed, and then click **Save**.

The configuration file is modified; however, the changes will not take effect immediately. You must restart Service Impact to activate the changes.

Editing server configuration files in the Impact container (RM 5.0.x)

Perform this procedure in the Service Impact container to edit Service Impact server configuration files with Resource Manager 5.0.x.

- 1 Complete the following steps to log in to the Impact container as the `zenossimpact` user.
 - a Log in to the Control Center master host as a user with `serviced` CLI privileges.
 - b Start an interactive session in the Impact container.

```
serviced service attach impact
```

- c In the new session, switch user to `zenossimpact`.

```
su - zenossimpact
```

- 2 Change directory to the configuration properties directory.

```
cd $IMPACT_HOME/etc
```

- 3 Edit configuration files as desired or directed.
- 4 Restart Service Impact, and then log off the Service Impact server host.
 - a Restart the Service Impact service using the Java `pkill` command.

```
pkill java
```

Note

If you restart Service Impact using the `serviced` command or the Control Center, the configuration file modifications are not preserved.

- b Exit the `zenossimpact` session.

```
exit
```

Editing server configuration files in the Impact container (RM 4.2.x)

If Service Impact is deployed with Resource Manager 4.2.x., perform this procedure to edit Service Impact server configuration files.

- 1 Log in to the Service Impact server host as `zenossimpact`.
- 2 Change directory to the configuration properties directory.

```
cd $IMPACT_HOME/etc
```

- 3 Edit configuration files as desired or directed.
- 4 Restart Service Impact, and then log off the Service Impact server host.
 - a Restart the Service Impact.

```
service zenoss_impact restart
```


b Log off.

```
exit
```

Service Impact Glossary

component node

An object representing a Resource Manager component within one or more Service Impact models.

contextual policy

A policy that defines how a node's state will change and how it will be propagated to its parent nodes within a single service model context.

custom state provider

Customized nodes that gather state data from classes other than the standard `/Status/Ping` and `/Perf` classes.

device node

An object representing a Resource Manager device within one or more Service Impact models.

global policy

A policy that defines how a node's state will change and how it will be propagated to all parent nodes in all service model contexts in which the node participates.

logical node

An object that represents the existence of specific types of Resource Manager events (instead of monitored entities like devices or components) within one or more Service Impact models. The logical node's definition identifies which events to match against and how its node state within Service Impact models should change according to each type of matched event.

production state propagation threshold

The minimum device production state that Service Impact uses to decide whether to propagate availability and performance states within a service graph.

service event

A Resource Manager event, generated by Service Impact when an availability or performance state change affects a service node.

service node

A Service Impact object that represents either a service model as a whole, or a service on which other service nodes rely (a subservice).

state trigger

A rule that examines the states of child nodes and selects the type of state data to send to the parent level in the service model.