ServiceWatch Integration with BMC Atrium

Installation and Operation

August 2014

Contact Information

For customer support:
http://www.servicenow.com/support/contact-support.html

For all other purposes:
http://info.servicenow.com/contact-us

Contents

Figures

Introduction

The integration of ServiceNow **ServiceWatch** with **BMC Atrium's CMDB** transfers ServiceWatch-discovered CIs, attributes, relationships and business services to BMC Atrium's CMDB. This document describes the integration architecture and explains how to install, operate, configure, and tailor the integration for your BMC Atrium CMDB implementation.

The package for integrating ServiceWatch with BMC Atrium CMDB is provided with the ServiceWatch installation and can be found in the **cmdb\atrium** folder of the server installation directory.

The package for integrating ServiceWatch with BMC Atrium's CMDB works on both the SaaS and On-Premise versions of ServiceWatch. It consists of a batch program that runs at a specified time to synchronize the CIs and business services in BMC Atrium with ServiceWatch-discovered CIs and business services. Because standard web service APIs access both ServiceWatch and BMC Atrium, the integration can run in local and cloud environments whenever there is HTTPS connectivity to the environments.

Depending on the parameters provided by the operator when running the integration package, the package will synchronize all of the business services active in ServiceWatch or just a single service.

Accurate up-to-date business service data is required inside BMC Atrium for these purposes:

- Understanding the service impact when an incident is reported for a specific CI
- Understanding the service impact of a planned change to a CI
- Understanding the risk to service availability based on proposed changes
- Analyzing current service topology to determine if a disaster recovery plan is adequate

However, instructions about how to use ServiceWatch-generated CI and business service data for these purposes in BMC Atrium's CMDB is *not* covered in this document.

Architecture & Functional Overview

The package for integrating ServiceWatch with BMC Atrium's CMDB is a batch file that runs on a Microsoft Windows server. The integration accesses the ServiceWatch application using the REST web service API to retrieve all (or the requested) business services and all the CIs and relationships that are part of those business services, including applications, servers, and network devices.

The ServiceWatch-discovered CIs are translated to BMC Atrium CI format. The transformation results in CIs that are identical to BMC Atrium-discovered components. This CI-to-CI mapping is configurable via metadata configuration files that allow any type of ServiceWatch-discovered CIs, attributes and relationships to be translated and imported as BMC Atrium CMDB objects. See TAILORING THE INTEGRATION on page 13 for details.

After the data is converted to BMC Atrium format, the integration package accesses BMC Atrium using the BMC AR Web Services API described in the BMC ATRIUM PRODUCT DOCUMENTATION website.

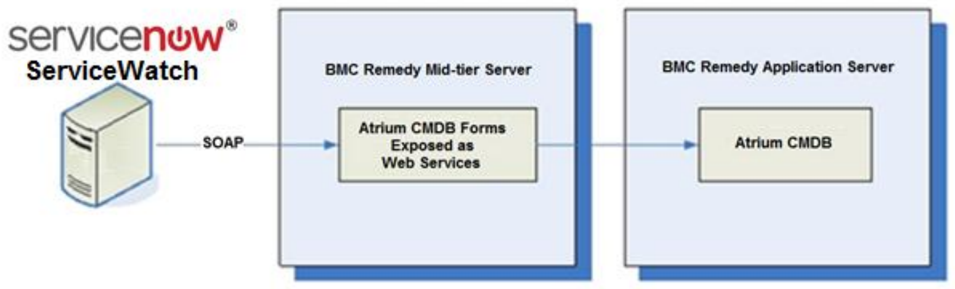
The integration uses the following processes to populate BMC Atrium CMDB objects:

- CIs inside the BMC Atrium CMDB that are part of the business service are identified and marked as business service members by adding the business service name as a Ci attribute.
- The integration checks the relationships in between each CI and its business service. If the ServiceWatch-discovered application flow indicates a change is needed, the relationship is updated and any missing CIs are added in BMC Atrium CMDB.

The integration currently supports all major CIs and CI subtypes defined in BMC Atrium. The full CI list is in APPENDIX A: LIST OF SUPPORTED CI TYPES. Any CI that does not have a subtype in BMC Atrium is tagged as a Generic BMC Atrium CMDB CI. You can easily extend support for new CIs by using the metadata configuration file **mapping.xml** as explained in TAILORING THE INTEGRATION on page 13.

Figure 1 is a graphic illustration of how the ServiceWatch and BMC Atrium CMDB integration works. ServiceWatch uses a unique "top-down" discovery method to map the business service and all of its associated CIs. The integration runs as a batch file on a Microsoft Windows server and retrieves business service data from ServiceWatch via a web services API, translates the retrieved data to BMC Atrium CI types, and populates the relationships in the BMC Atrium CMDB.

Figure 1: Graphic illustration of how the integration works


servicenow
ServiceWatch
SOAP
BMC Remedy Mid-tier Server
Atrium CMDB Forms Exposed as Web Services
BMC Remedy Application Server
Atrium CMDB

Prerequisites

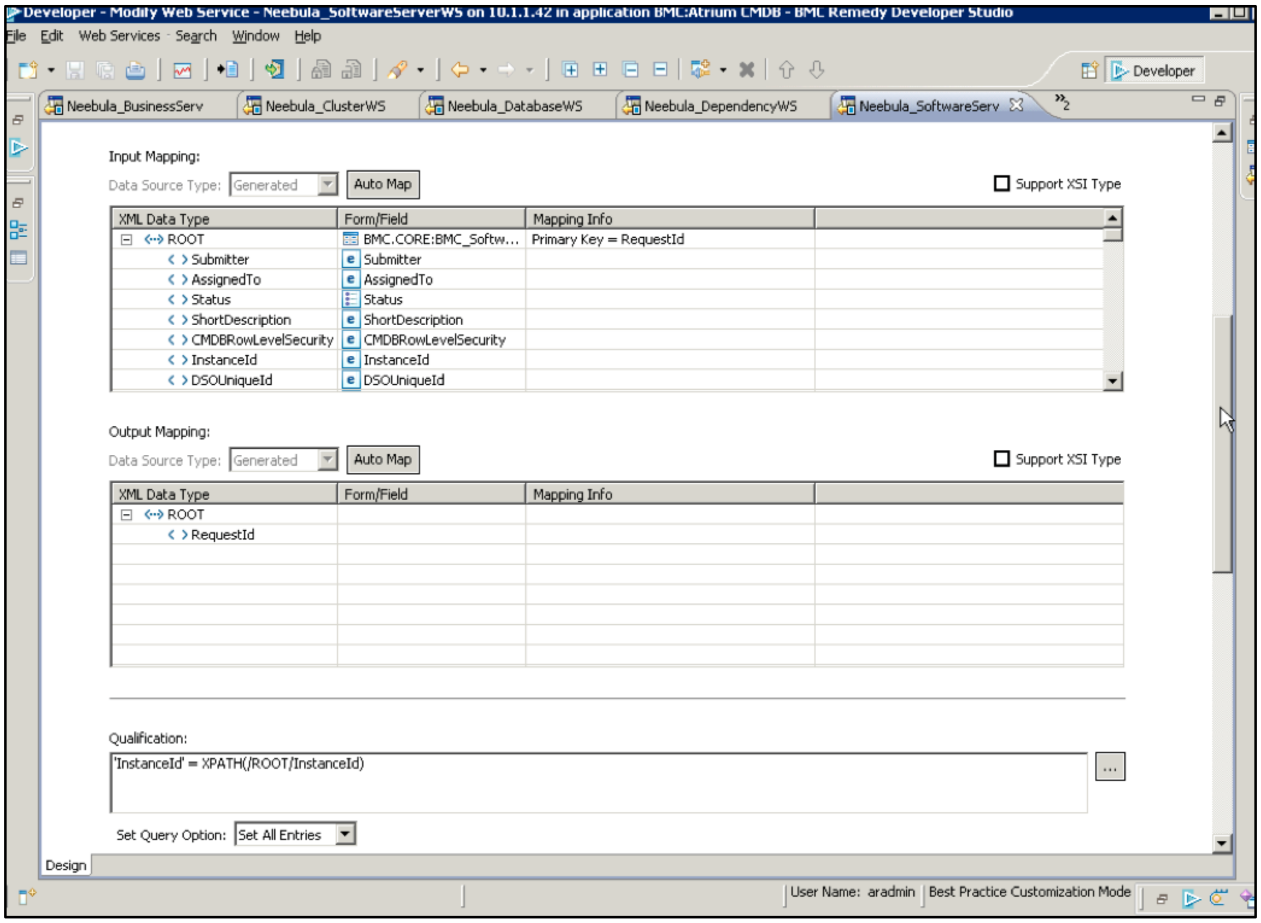
Before starting the integration, create the following web services with exactly the same parameters on the BMC AR side using BMC Remedy Developer Studio:

- ServiceWatch_ApplicationServiceWS
- ServiceWatch_BusinessServiceWS
- ServiceWatch_ClusterWS
- ServiceWatch_ComputerSystemWS
- ServiceWatch_Datab aseWS
- ServiceWatch_DependencyWS
- ServiceWatch_ImpactWS
- ServiceWatch_SoftwareServerWS

After the installation, typical examples of each of these web services will be available on disk.

Each created web service must have the qualification attribute values shown in Figure 2:

Figure 2: Qualification Attribute Values


Developer - Modify Web Service - Neebula_SoftwareServerWS on 10.1.1.42 in application BMC:Atrium CMDB - BMC Remedy Developer Studio
File Edit Web Services Search Window Help
Developer
Neebula_BusinessServ
Neebula_ClusterWS
Neebula_DatabaseWS
Neebula_DependencyWS
Neebula_SoftwareServ
Auto Map
Design

Input Mapping:

Data Source Type: Generated

Support XSI Type

XML Data Type	Form/Field	Mapping Info
ROOT	BMC.CORE:BMC_Softw…	Primary Key = RequestId
Submitter	Submitter	
AssignedTo	AssignedTo	
Status	Status	
ShortDescription	ShortDescription	
CMDBRowLevelSecurity	CMDBRowLevelSecurity	
InstanceId	InstanceId	
DSOUniqueId	DSOUniqueId	

Output Mapping:

Data Source Type: Generated

Support XSI Type

XML Data Type	Form/Field	Mapping Info
ROOT		
RequestId		

Qualification:

'InstanceId' = XPATH(/ROOT/InstanceId)

Set Query Option: Set All Entries

User Name: aradmin | Best Practice Customization Mode

ServiceWatch in On-premise Mode

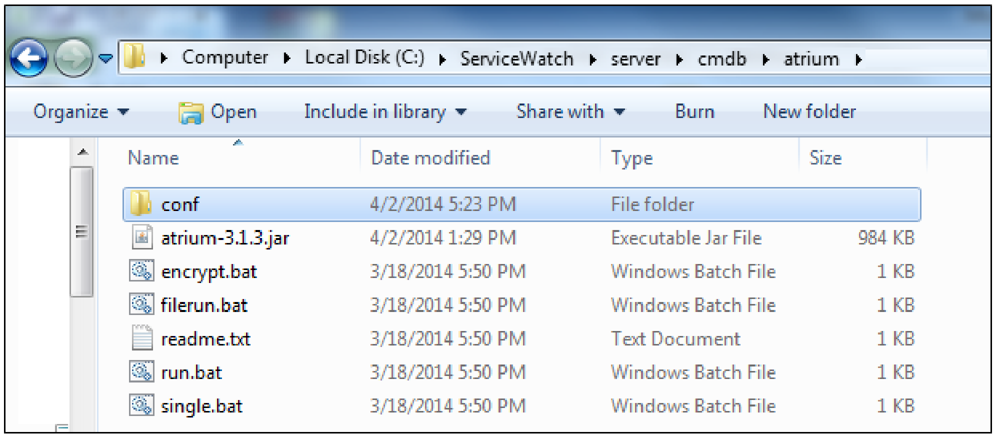
If you are using ServiceWatch as a Service, skip to ServiceWatch as a Service on page 11.

Installation

The ServiceWatch with BMC Atrium CMDB integration files are distributed as part of the ServiceWatch installation package.

1. Navigate to the **ServiceWatch > server > cmdb > atrium** folder on the same Microsoft Windows server that will run the integration. You should see the following files in that folder:

Figure 3: Contents of the ServiceWatch > server > cmdb > atrium folder



Name	Date modified	Type	Size
conf	4/2/2014 5:23 PM	File folder	
atrium-3.1.3.jar	4/2/2014 1:29 PM	Executable Jar File	984 KB
encrypt.bat	3/18/2014 5:50 PM	Windows Batch File	1 KB
filerun.bat	3/18/2014 5:50 PM	Windows Batch File	1 KB
readme.txt	3/18/2014 5:50 PM	Text Document	1 KB
run.bat	3/18/2014 5:50 PM	Windows Batch File	1 KB
single.bat	3/18/2014 5:50 PM	Windows Batch File	1 KB

2. Use the **encrypt.bat** file to produce the two encrypted passwords that are in the **config.properties** file. The only parameter for the **encrypt.bat** command is the unencrypted password. The output is a 50-character encrypted password. For example:

```
C:\neebula\servicewatch\collector\cmdb\atrium>encrypt.bat mypassword
Encrypted key:< 50-character encrypted password >
```

Configuration

3. Specify values for the parameters in the **config.properties** file.
   - Set **cmdb_uname** to a BMC Atrium username with admin authority.
   - Set **cmdb_password** to that username's 50-character encrypted password.
   - Set **cmdb_url** to the Base AR web services URL.
   - Set **neebula_uname** to an administrator user of your ServiceWatch application.
   - Set **neebula_password** to the 50-character Encrypted key for that administrator user.
   - Set **neebula_url** to your ServiceWatch server URL.
   - Set **dataset_id** to the data set ID that the integration will push data into.
   - When the **create_hosts** and **network** parameters are set to **true**, the integration adds hosts and network devices if they do not exist in the BMC Atrium CMDB. CIs and relationships are automatically created if they do not exist in the BMC Atrium CMDB and their hosts do exist.
   - **business_service_rel_attr** specifies the BMC Atrium attribute name that contains the names of the Business Services.

4. After setting values in the **config.properties** file, it will look similar to:

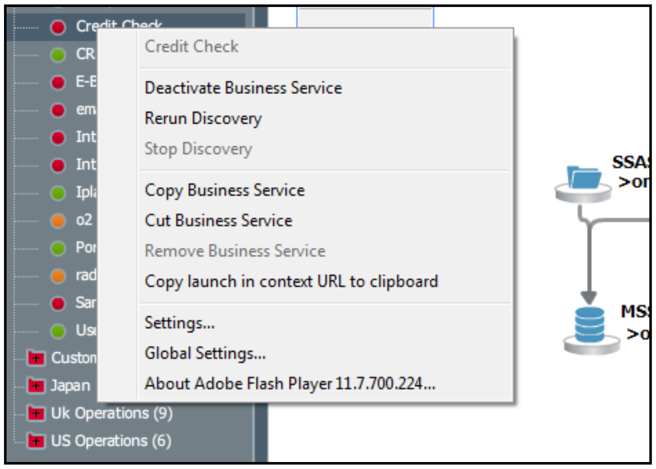
```
cmdb_uname=aradmin
cmdb_password=JVxCgyArl3uSh0nw52zBGw==||5LpSuI7MdKJIOnLUlCxTYw==
cmdb_url=http://10.1.1.42:8080/arsys/services/ARService?server=v-cmdb
neebula_uname=admin
neebula_password=6se+NSirh6UuJm9SQhaPRQ==||b1mZ7kg1CVL9UUw1EPGkVg==
neebula_url=http://localhost:8080/ws/bss
create_hosts=true
network=false
dataset_id=BMC.ASSET
alias_field_name=Alias
business_service_rel_attr=ciTag
lb_types=F5 BigIP LTM,Alteon,Cisco CSM,Cisco CSS,Citrix Netscaler,Network
Load Balancer,Radware Load Balancer
```

5. Save the **config.properties file**. Configuration of ServiceWatch with BMC Atrium CMDB integration is complete.

6. The **run.bat** and **single.bat** files can be scheduled to run at a suitable time based on your update policy:
   - **run.bat** has no configurable parameters. It copies all active ServiceWatch-discovered business services to the BMC Atrium CMDB.
   - **single.bat** receives one parameter whose value is an integer that identifies a specific business service. The integration package will populate only this business service, including all of its CIs, in the BMC Atrium CMDB. The syntax is: `single.bat < Business Service ID # >`
     For example, `single.bat 1234`

   To obtain the Business Service ID number, right click the desired Business Service in the **Active** tree and select **Copy launch in context URL to clipboard** in the pop-up menu. See the **config.properties** and **mapping.xml** files for credentials and advanced mapping settings.

   Figure 4: Business Service right-click pop-up menu



   The clipboard now contains a URL similar to
   `http://localhost:8080/ui/?screen=monitor&bsid=nnnn`

Copy the integer after **bsid=** from the clipboard and use it as the parameter value for the **single.bat** command. When you run the **single.bat** command, the business service whose number you specified, including its CIs, is added to or modified in BMC Atrium.

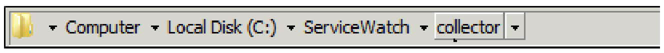
Skip to Tailoring the Integration on page 13.

ServiceWatch as a Service

If you are using ServiceWatch in on-premise mode, run the installation and configuration steps in ServiceWatch in On-premise Mode on page 8, then skip to Tailoring the Integration on page 13.

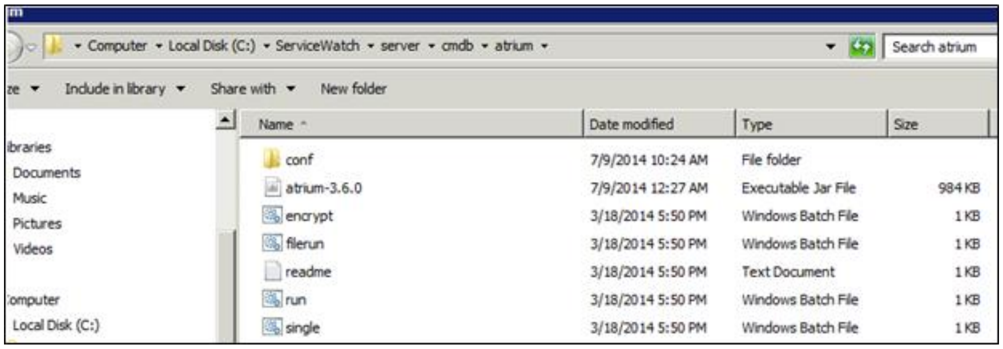
Installation

1. Obtain ServiceWatch with the BMC Atrium CMDB installation zip file from the ServiceWatch customer support team.
2. Create the **cmdb** folder in the **ServiceWatch > collector** path on the same Microsoft Windows server that will run the integration.



3. Extract the compressed files. The extraction should create an **atrium** folder that contains a subfolder and files similar to those in Figure 5.

Figure 5: List of extracted files



Name	Date modified	Type	Size
conf	7/9/2014 10:24 AM	File folder	
atrium-3.6.0	7/9/2014 12:27 AM	Executable Jar File	984 KB
encrypt	3/18/2014 5:50 PM	Windows Batch File	1 KB
filerun	3/18/2014 5:50 PM	Windows Batch File	1 KB
readme	3/18/2014 5:50 PM	Text Document	1 KB
run	3/18/2014 5:50 PM	Windows Batch File	1 KB
single	3/18/2014 5:50 PM	Windows Batch File	1 KB

4. Use the **encrypt.bat** file to produce the two encrypted passwords that are in the **config.properties** file. The only parameter for the **encrypt.bat** command is the unencrypted password. The output is a 50-character encrypted password. For example:

```
C:\neebula\servicewatch\collector\cmdb\atrium>encrypt.bat mypassword
Encrypted key:< 50-character encrypted password >
```

Configuration

5. Specify values for the parameters in the **config.properties** file.
   - Set **cmbd_uname** to a BMC Atrium username with admin authority.
   - Set **cmbd_password** to that username's 50-character encrypted password.
   - Set **cmbd_url** to the URL of your Base AR web services URL.
   - Leave the **neebula_uname value** empty (null).
   - You can leave **neebula_password** as it is. The **apikey** is used to authenticate against the ServiceWatch as a Service server.
   - Do not change the **neebula_url** setting. It should point to **https://saas.neebula.com/ws/bss**
   - Set **dataset_id** to the data set ID the integration will push data into.
   - Set **apikey** to the value obtained from the ServiceWatch Customer Service Center.
   - When the **create_hosts** and **network** parameters are set to **true**, they direct the integration to add hosts and network devices if they so not already exist in the BMC Atrium CMDB. CIs and

relationships are automatically created if they do not already exist in the BMC Atrium CMDB and their hosts do exist.
- **business_service_rel_attr** specifies the BMC Atrium attribute name that contains the names of the Business Services.

6. After setting values in the **config.properties** file, it will look similar to:

```
cmdb_uname=aradmin
cmdb_password=JVxCgyArl3uSh0nw52zBGw==||5LpSuI7MdKJIOnLUlCxTYw==
cmdb_url=http://10.1.1.42:8080/arsys/services/ARService?server=v-cmdb
neebula_uname=admin
neebula_password=6se+NSirh6UuJm9SQhaPRQ==||b1mZ7kg1CVL9UUw1EPGkVg==
neebula_url=https://saas.neebula.com/ws/bss
apikey=YbJb9OTSds4dd6L7NFGaew
create_hosts=true
network=false
dataset_id=BMC.ASSET
alias_field_name=Alias
business_service_rel_attr=ciTag
lb_types=F5 BigIP LTM,Alteon,Cisco CSM,Cisco CSS,Citrix Netscaler,Network
Load Balancer,Radware Load Balancer
```

7. Save the **config.properties** file. Configuration of ServiceWatch with BMC Atrium CMDB integration is complete.

8. The **run.bat** and **single.bat** files can be scheduled to run at a suitable time based on your update policy:
   - **run.bat** has no configurable parameters. It copies all ServiceWatch-discovered business services to the BMC Atrium CMDB.
   - **single.bat** receives one parameter whose value is an integer that identifies a specific business service. The integration package will populate only this business service, including all of its CIs, in the BMC Atrium CMDB. The syntax is: `single.bat < Business Service ID # >`
     For example, `single.bat 1234`

   To obtain the Business Service ID number, right click the desired Business Service in the **Active** tree and select **Copy launch in context URL to clipboard** in the pop-up menu (see Figure 4: Business Service right-click pop-up menu).

   The clipboard now contains a URL similar to
   `http://localhost:8080/ui/?screen=monitor&bsid=nnnn`
   Copy the integer after **bsid=** from the clipboard and use it as the parameter value for the **single.bat** command. When you run the **single.bat** command, the business service whose number you specified, including its CIs, is added to or modified in BMC Atrium.

Tailoring the Integration

The integration of ServiceWatch with the BMC Atrium CMDB is very flexible. It can be modified to add any CI or relationship, CI Key, or CI identification algorithm. These capabilities will be available in the next ServiceWatch GA release. For most users, no changes will be required to any of the default configurations.

The following paragraphs explain how to change the mapping in the integration. For additional help, contact the ServiceWatch Customer Service Center.

The **mapping.xml** file in the **orchestrator-server > cmdb > atrium > conf** directory controls how the ServiceWatch with BMC Atrium CMDB integration works.

Deleting/Changing/Adding ServiceWatch CIs & Relationships in Atrium

Locate the following section of the **mapping.xml** file. It contains the list of BMC Atrium relationships and CIs handled by the integration. All non-defined CIs are mapped to a Generic CI in the BMC Atrium CMDB.

The process for converting attributes and keys and binding them to the CMDB has been specified for each defined CI. To add a new CI to the ServiceWatch integration, first display its xml in BMC Atrium and then copy its CI name (such as cmdb_ci_computer) and add it to this list.

```
<bean id="Mappings" class="java.util.ArrayList">
            <constructor-arg>
                <list>
                    <ref bean="BMC_SoftwareServer" />
                    <ref bean="BMC_DataBase" />
                    <ref bean="BMC_ApplicationService" />
                    <ref bean="BMC_ComputerSystem" />
                    <ref bean="BMC_BusinessService" />
                    <ref bean="BMC_Cluster" />
                    <ref bean="BMC_Dependency" />
                    <ref bean="BMC_Impact" />
                    <ref bean="BMC_Impact2" />
                    <ref bean="AllOther" />
                </list>
            </constructor-arg>
    </bean>
```

Mapping ServiceWatch-Discovered CIs to BMC Atrium CIs

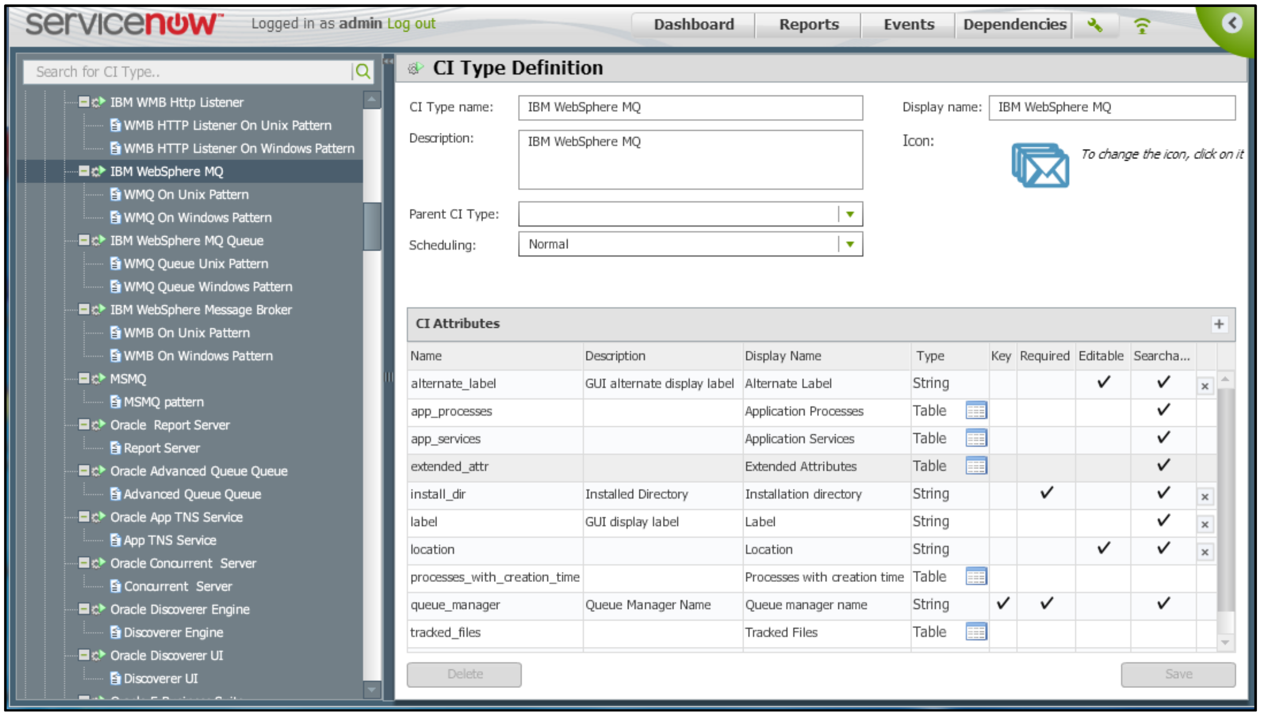
The ServiceWatch with BMC Atrium CMDB integration contains an XML representation that enables the default procedures for converting and binding CIs to be changed**.** Any ServiceWatch attribute that is available for a CI (as defined in the CI-Type definition) can be used in the conversion and binding procedures. In addition, the following attributes are automatically populated and available for conversion for each ServiceWatch CI.

Note: The attribute name below is the same name that should be used in the definition:

Attribute	**Description**
Id	ServiceWatch CI ID number – used to create an attribute inside BMC Atrium for a change impact analysis launch in context URL (example: http://ui/?screen=physical&ciid=8763)
version	Version of the software including patch level
label	Name of the application
description	Description entered by the user
typeName	Type of CI in ServiceWatch
bsId	List of numbers (IDs) separated by ‘,’ of the business services to which the CI belongs
bsName	List of names separated by ‘,’ of the business services the CI supports
host.Short_host_name	Short representation of host name (not fqdn)
host.Serial_number	Serial number of the host where the CI resides; primary key for host binding
host.Primary_management_IP	
host.Primary_host_name	fqdn
host.Primary_host_name_or_ip	
host.OS_Type	operating system name
host.OS_Version	Includes the patch level of the OS
host.Model	
host.Domain	
host.Address_Width	
type	represents connection type (HTTP, TCP, etc.)

In addition, all CI Type attributes (as defined in the ServiceWatch CI) are available as seen in this screen:

Figure 6: Table of CI Attributes in the CI Type Definition screen



CI Type Definition

CI Type name: IBM WebSphere MQ
Display name: IBM WebSphere MQ
Description: IBM WebSphere MQ
Icon: *To change the icon, click on it*
Parent CI Type:
Scheduling: Normal

CI Attributes

Name	Description	Display Name	Type	Key	Required	Editable	Searcha...
alternate_label	GUI alternate display label	Alternate Label	String			✓	✓
app_processes		Application Processes	Table				✓
app_services		Application Services	Table				✓
extended_attr		Extended Attributes	Table				✓
install_dir	Installed Directory	Installation directory	String		✓		✓
label	GUI display label	Label	String				✓
location		Location	String			✓	✓
processes_with_creation_time		Processes with creation time	Table				
queue_manager	Queue Manager Name	Queue manager name	String	✓	✓		✓
tracked_files		Tracked Files	Table				

To determine possible candidates for attributes in the BMC Atrium CMDB, look at the XML that defines the CI Type in BMC Atrium.

Mapping a ServiceWatch-discovered WebSphere CI to BMC Atrium CI

How **java.util.ArrayList** defines which CIs are converted is described in DELETING/CHANGING/ADDING SERVICEWATCH CIS & RELATIONSHIPS IN ATRIUM on page 13. How CI attributes are translated is described here. For each CI in **java.util.ArrayList**, the XML below maps a Service-Watch-discovered CI to a BMC Atrium CI. Each ServiceWatch-discovered attribute overrides the corresponding attribute of the matching BMC Atrium CI. The default process maps only required attributes in order to minimize changes to the BMC Atrium CMDB.

CI Attribute Translation

```
<bean id="BMC_ApplicationService"
class="com.neebula.orchestrator.integration.cmdb.sn.handler.BMCToNeebulaMa
pping">
        <property name="nameSpace" value="BMC.CORE"/>
        <property name="bmcCIT" value="BMC_ApplicationService"/>
        <property name="neebulaCIT" value="EMS Queue,Tomcat
WAR,WeblogicModule,Websphere EAR,WMB Flow,IBM WebSphere MQ
Queue,ActiveMatrix Business Works Process,Advanced Queue Queue,EMS
Queue,Jboss module,Oracle Weblogic JMS Queue,MS SQL database,Oracle DB
schema,Virtual Directory,ActiveMatrix Business Works Process"/>
        <property name="mappingRules">
            <map key-type="java.lang.String" value-
type="java.lang.String">
                <entry key="VersionNumber" value="${version}"/>
                <entry key="Name" value="${label}"/>
                <entry key="ShortDescription" value="${label}"/>
                <entry key="Description" value="${description}"/>
                <entry key="Category" value="'Software'"/>
                <entry key="Type" value="'Application Service'"/>
                <entry key="Item" value="'BMC Discovered'"/>
                <entry key="Alias"
value="${host.Short_host_name}+'::'+${shortType}+'::'+${label}"/>
                <entry key="reconciliationIdentity"
value="${host.Short_host_name}+'::'+${shortType}+'::'+${label}"/>
            </map>
        </property>
        <property name="keyRules" ref="AliasRule"/>
    </bean>
```

Sample XML for CI-to-CI mapping

<property name="cmdbCIT" value="cmdb_ci_app_server_websphere"/> - Name of the BMC Atrium CI Type
<property name="neebulaCIT" value="Websphere"/> Name of the ServiceWatch CI Type. This could also contain a list of CI Types separated by commas
<property name="mappingRules"> Start of the attribute mapping session
<map key-type="java.lang.String" value-type="java.lang.String"> Type of mapping; in this case String <-> String
<entry key="Version" value="${version}"/> Key= Name of BMC Atrium attribute
Value= an expression that accepts ServiceWatch attributes in the format ${neebula-CI-att}; ${version} is used in this example.
Implant constants and concatenate variables similar to the following example can also be used:
value="${label}+'@'+${Primary_host_name}"
creates a value of Websphere1@mainhost.mycompany.local
Very complex translations can be created with appropriate code. For more information see
http://groovy.codehaus.org/User+Guide
<property name="keyRules">Defines how to create the CI Key in BMC Atrium and the order in which keys are compared for CI binding in BMC Atrium. The comparison order is predefined. For example, the host sequence is:

```
`<bean id="AliasRule" class="java.util.ArrayList">
        <constructor-arg>
            <list value-type="com.doitwise.neebula.utils.Pair">
                <bean class="com.doitwise.neebula.utils.Pair">
                    <property name="first" value="Name"></property>
                    <property name="second" value="${label}+'@'
                      +${host.Primary_host_name}"></property>
                </bean>
                <bean class="com.doitwise.neebula.utils.Pair">
                    <property name="first" value="Name"></property>
                    <property name="second" value="${label}+'@'
                      +${host.Primary_management_IP}"></property>
                </bean>
            </list>
        </constructor-arg>
    </bean>
```

This rule tells the integration to first compare the Serial number, then the host name, and finally the IP.

Mapping ServiceWatch Relationships

Mapping relationships is performed after ServiceWatch-discovered CIs have been incorporated in the BMC Atrium CMDB. The integration activates the relationships section for each ServiceWatch-discovered CI if there is a matching CI in the BMC Atrium CMDB. The integration sets these BMC Atrium variables to appropriate values.

Variable	**Description**
src.id	BMC Atrium ID of the source CI
src.type	Type of source CI in BMC Atrium
target.id	ID of target CI in BMC Atrium
target.type	Type of target CI in BMC Atrium

Relationship mapping example

```
<bean id="BMC_Dependency"
class="com.neebula.orchestrator.integration.cmdb.sn.handler.BMCT
oNeebulaMapping">
        <property name="handler"
value="com.neebula.orchestrator.integration.cmdb.sn.handler.Rela
tionHandler"/>
        <property name="nameSpace" value="BMC.CORE"/>
        <property name="bmcCIT" value="BMC_Dependency"/>
        <property name="neebulaCIT"
value="APPLICATION_FLOW,INCLUSION,CLUSTER,RELATION_BSS_MEMBER"/>

        <property name="mappingRules">
            <map key-type="java.lang.String" value-
type="java.lang.String">
                <entry key="Name" value="${id}"/>
                <entry key="ShortDescription" value="'desc'"/>
                <entry key="sourceInstanceId"
value="${src.id}"/>
                <entry key="destinationInstanceId"
value="${target.id}"/>
                <entry key="reconciliationIdentity"
value="${src.id}+'::'+${target.id}"/>
            </map>
        </property>
    </bean>
```

Conclusion

We hope the information in this document answers most of your questions about how to implement and run the integration. The ServiceNow ServiceWatch Customer Support staff are ready to provide assistance if required. The ServiceWatch team wishes you success in transferring our accurate up-to-date business service models from ServiceWatch into the BMC Atrium CMDB.

About ServiceNow

ServiceNow provides Service-Centric Mapping software that improves IT performance and availability through an automated and unified approach to mapping business services which is about 20 times faster and 80 percent less costly than traditional solutions. Engineered for SaaS delivery, ServiceWatch encourages IT organizations to shift from monitoring data center silos (servers, networks, storage, applications) to managing end-user business services (CRM, billing, tax payments, fund transfers, etc.). Believing that effective IT “Starts with the Map,” ServiceNow’s unique technology automatically creates and maintains a run-time map of business services including underlying physical, virtual, network, and storage infrastructures.

Focused on business impact and realizing that IT should monitor only what matters, ServiceWatchs run-time service map enriches event management and monitored information by presentation in the context of the business service, resulting in improved IT change control, rapid problem isolation, and meaningful service health monitoring. While no CMDB is required, ServiceWatch service maps can be imported into BMC Software, CA Technologies, HP, IBM and BMC Atrium applications, making existing CMDBs “service-aware” with run-time accuracy.

ServiceNow has an installed base of global enterprises, Fortune 10,000 companies, and government/education customers around the world.

Appendix A: List of Supported CI Types

All CIs supported by the integration are defined in the **mapping.xml** file in the main directory. New CIs can be easily added and mapped to the integration. See TAILORING THE INTEGRATION on page 13 for details. The default definition for the attributes associated with the following CIs and relationships is the same format used for BMC Atrium discovery.

Application Servers:

- ✓ BMC_ApplicationService
- ✓ BMC_BusinessService
- ✓ BMC_Cluster
- ✓ BMC_ComputerSystem
- ✓ BMC_DataBase
- ✓ BMC_Dependency
- ✓ BMC_SoftwareServer

Appendix B: Default Installation mapping.xml File

The complete XML code in the **mapping xml** file of ServiceWatch with the BMC Atrium CMDB integration package is listed below. This listing is provided for reference only and is intended to show how each attribute is translated and transferred to the BMC Atrium CMDB by this out-of-the-box integration package. This code may change, so use the copy provided with your installation. If the **mapping xml** file is updated to tailor the integration, back up your custom tailored file when upgrading to ensure that the new version works properly in your environment and produces the desired results.

```
<?xml version="1.0" encoding="UTF-8"?>
<beans xmlns="http://www.springframework.org/schema/beans"
       xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
       xsi:schemaLocation="http://www.springframework.org/schema/beans
http://www.springframework.org/schema/beans/spring-beans-3.0.xsd">

    <bean id="MappingHandler"
class="com.neebula.orchestrator.integration.cmdb.sn.handler.MappingHandler
" scope="singleton">
        <property name="mappingList" ref="Mappings"/>
    </bean>

    <bean id="BMC_SoftwareServer_Service"
class="com.neebula.orchestrator.integration.cmdb.sn.handler.SoftwareServer
Service" scope="singleton" />
    <bean id="BMC_DataBase_Service"
class="com.neebula.orchestrator.integration.cmdb.sn.handler.DatabaseServic
e" scope="singleton" />
    <bean id="BMC_ApplicationService_Service"
class="com.neebula.orchestrator.integration.cmdb.sn.handler.ApplicationSer
viceService" scope="singleton" />
    <bean id="BMC_ComputerSystem_Service"
class="com.neebula.orchestrator.integration.cmdb.sn.handler.ComputerSystem
Service" scope="singleton" />
    <bean id="BMC_BusinessService_Service"
class="com.neebula.orchestrator.integration.cmdb.sn.handler.BusinessServic
eService" scope="singleton" />
    <bean id="BMC_Cluster_Service"
class="com.neebula.orchestrator.integration.cmdb.sn.handler.ClusterService
" scope="singleton" />
    <bean id="BMC_Dependency_Service"
class="com.neebula.orchestrator.integration.cmdb.sn.handler.DependencyServ
ice" scope="singleton" />
    <bean id="BMC_Impact_Service"
class="com.neebula.orchestrator.integration.cmdb.sn.handler.ImpactService"
scope="singleton" />
    <bean id="BMC_Impact2_Service"
class="com.neebula.orchestrator.integration.cmdb.sn.handler.ImpactService"
scope="singleton" />

    <bean id="Mappings" class="java.util.ArrayList">
            <constructor-arg>
                <list>
                    <ref bean="BMC_SoftwareServer" />
                    <ref bean="BMC_DataBase" />
                    <ref bean="BMC_ApplicationService" />
                    <ref bean="BMC_ComputerSystem" />
                    <ref bean="BMC_BusinessService" />
                    <ref bean="BMC_Cluster" />
                    <ref bean="BMC_Dependency" />
                    <ref bean="BMC_Impact" />
                    <ref bean="BMC_Impact2" />
                    <ref bean="AllOther" />
                </list>
            </constructor-arg>
    </bean>

    <!-- CIT Mapping Rules
   available attributes to use:
   version
   label
   description
   typeName
   host.Short_host_name - short representation of host name (not fqdn)
   host.Serial_number
   host.Primary_management_IP
   host.Primary_host_name - fqdn
   host.OS_Name - operating system name
   host.OS_Version
   host.Model
   host.Domain
   host.Address_Width
   type - represents the connection type (HTTP, TCP, etc...)
   -->
    <bean id="AllOther"
class="com.neebula.orchestrator.integration.cmdb.sn.handler.BMCToNeebulaMa
pping">
        <property name="nameSpace" value="BMC.CORE"/>
        <property name="bmcCIT" value="BMC_SoftwareServer"/>
        <property name="neebulaCIT" value="*"/>
        <property name="handler"
value="com.neebula.orchestrator.integration.cmdb.sn.handler.GenericAppServ
erHandler"/>
        <property name="mappingRules">
            <map key-type="java.lang.String" value-
type="java.lang.String">
                <entry key="VersionNumber" value="${version}"/>
                <entry key="Name" value="${label}"/>
                <entry key="ShortDescription" value="${label}"/>
                <entry key="Description" value="${description}"/>
                <entry key="Category" value="'Software'"/>
                <entry key="Type" value="'Application'"/>
                <entry key="Alias"
value="${host.Short_host_name}+'::'+${shortType}+'::'+${label}"/>
                <entry key="reconciliationIdentity"
value="${host.Short_host_name}+'::'+${shortType}+'::'+${label}"/>
            </map>
        </property>
        <property name="keyRules" ref="AliasRule"/>
    </bean>

    <bean id="BMC_Cluster"
class="com.neebula.orchestrator.integration.cmdb.sn.handler.BMCToNeebulaMa
pping">
        <property name="nameSpace" value="BMC.CORE"/>
        <property name="bmcCIT" value="BMC_Cluster"/>
        <property name="neebulaCIT"
value="Applicative_cluster_container"/>
        <property name="mappingRules">
            <map key-type="java.lang.String" value-
type="java.lang.String">
                <entry key="VersionNumber" value="${version}"/>
                <entry key="Name" value="${clusterName}"/>
                <entry key="ShortDescription" value="${label}"/>
                <entry key="Description" value="${description}"/>
                <entry key="Category" value="'Unknown'"/>
                <entry key="Type" value="'Unknown'"/>
                <entry key="Item" value="'BMC Discovered'"/>
                <entry key="Alias" value="${clusterName}"/>
                <entry key="reconciliationIdentity"
value="'cluster::'+${clusterName}"/>
            </map>
        </property>
        <property name="keyRules">
            <list value-type="java.lang.String">
                <value>${clusterName}</value>
            </list>
        </property>
    </bean>

    <bean id="BMC_BusinessService"
class="com.neebula.orchestrator.integration.cmdb.sn.handler.BMCToNeebulaMa
pping">
        <property name="nameSpace" value="BMC.CORE"/>
        <property name="bmcCIT" value="BMC_BusinessService"/>
        <property name="neebulaCIT" value="BUSINESS_SERVICE"/>
        <property name="mappingRules">
            <map key-type="java.lang.String" value-
type="java.lang.String">
                <entry key="VersionNumber" value="${version}"/>
                <entry key="Name" value="${label}"/>
                <entry key="ShortDescription" value="${label}"/>
                <entry key="Description" value="${description}"/>
                <entry key="Category" value="'Software'"/>
                <entry key="Type" value="'Application'"/>
                <entry key="Alias" value="'BS::'+${label}"/>
                <entry key="reconciliationIdentity"
value="'BS::'+${label}"/>
            </map>
        </property>
        <property name="keyRules">
            <list value-type="java.lang.String">
                <value>'BS::'+${label}</value>
            </list>
        </property>
    </bean>

    <bean id="BMC_ComputerSystem"
class="com.neebula.orchestrator.integration.cmdb.sn.handler.BMCToNeebulaMa
pping">
        <property name="nameSpace" value="BMC.CORE"/>
        <property name="bmcCIT" value="BMC_ComputerSystem"/>
        <property name="neebulaCIT" value="host"/>
        <property name="mappingRules">
            <map key-type="java.lang.String" value-
type="java.lang.String">
                <entry key="VersionNumber" value="${version}"/>
                <entry key="Name" value="${label}"/>
                <entry key="ShortDescription" value="${label}"/>
                <entry key="Description" value="${description}"/>
                <entry key="Category" value="'Hardware'"/>
                <entry key="Type" value="'Hardware'"/>
                <entry key="Alias" value="${Primary_host_name}"/>
                <entry key="reconciliationIdentity"
value="'host::'+${Primary_host_name}"/>
            </map>
        </property>
        <property name="keyRules">
            <list value-type="java.lang.String">
                <value>${Primary_host_name}</value>
            </list>
        </property>
    </bean>

    <bean id="BMC_ApplicationService"
class="com.neebula.orchestrator.integration.cmdb.sn.handler.BMCToNeebulaMa
pping">
        <property name="nameSpace" value="BMC.CORE"/>
        <property name="bmcCIT" value="BMC_ApplicationService"/>
        <property name="neebulaCIT" value="EMS Queue,Tomcat
WAR,WeblogicModule,Websphere EAR,WMB Flow,IBM WebSphere MQ
Queue,ActiveMatrix Business Works Process,Advanced Queue Queue,EMS
Queue,Jboss module,Oracle Weblogic JMS Queue,MS SQL database,Oracle DB
schema,Virtual Directory,ActiveMatrix Business Works Process"/>
        <property name="mappingRules">
            <map key-type="java.lang.String" value-
type="java.lang.String">
                <entry key="VersionNumber" value="${version}"/>
                <entry key="Name" value="${label}"/>
                <entry key="ShortDescription" value="${label}"/>
                <entry key="Description" value="${description}"/>
                <entry key="Category" value="'Software'"/>
                <entry key="Type" value="'Application Service'"/>
                <entry key="Item" value="'BMC Discovered'"/>
                <entry key="Alias"
value="${host.Short_host_name}+'::'+${shortType}+'::'+${label}"/>
                <entry key="reconciliationIdentity"
value="${host.Short_host_name}+'::'+${shortType}+'::'+${label}"/>
            </map>
        </property>
        <property name="keyRules" ref="AliasRule"/>
    </bean>

    <bean id="BMC_DataBase"
class="com.neebula.orchestrator.integration.cmdb.sn.handler.BMCToNeebulaMa
pping">
        <property name="nameSpace" value="BMC.CORE"/>
        <property name="bmcCIT" value="BMC_DataBase"/>
        <property name="neebulaCIT" value="Oracle DB,MS SQL
server,PostgreSQL DB,MySQLServer,Sybase,DB2"/>
        <property name="mappingRules">
            <map key-type="java.lang.String" value-
type="java.lang.String">
                <entry key="VersionNumber" value="${version}"/>
                <entry key="Name" value="${label}"/>
                <entry key="ShortDescription" value="${label}"/>
                <entry key="Description" value="${description}"/>
                <entry key="Category" value="'Miscellaneous'"/>
                <entry key="Type" value="'Instance'"/>
                <entry key="Item" value="'Database'"/>
                <entry key="Alias"
value="${host.Short_host_name}+'::'+${shortType}+'::'+${label}"/>
                <entry key="reconciliationIdentity"
value="${host.Short_host_name}+'::'+${shortType}+'::'+${label}"/>
            </map>
        </property>
        <property name="keyRules" ref="AliasRule"/>
    </bean>

    <bean id="BMC_SoftwareServer"
class="com.neebula.orchestrator.integration.cmdb.sn.handler.BMCToNeebulaMa
pping">
        <property name="nameSpace" value="BMC.CORE"/>
        <property name="bmcCIT" value="BMC_SoftwareServer"/>
        <property name="neebulaCIT" value="Enterprise Message
Service,Websphere,Weblogic,Jboss,Oracle iAS,IIS,iplanet,Apache,Tomcat,IBM
WebSphere MQ,ActiveMatrix Business Works"/>
        <property name="mappingRules">
            <map key-type="java.lang.String" value-
type="java.lang.String">
                <entry key="VersionNumber" value="${version}"/>
                <entry key="Name" value="${label}"/>
                <entry key="ShortDescription" value="${label}"/>
                <entry key="Description" value="${description}"/>
                <entry key="Category" value="'Software'"/>
                <entry key="Type" value="'Application'"/>
                <entry key="Alias"
value="${host.Short_host_name}+'::'+${shortType}+'::'+${label}"/>
                <entry key="reconciliationIdentity"
value="${host.Short_host_name}+'::'+${shortType}+'::'+${label}"/>
            </map>
        </property>
        <property name="keyRules" ref="AliasRule"/>
    </bean>

    <!-- Connections Mapping Rules
    available attributes to use:
    src.id
    src.type
    target.id
    target.type
    type - represents the connection type (HTTP, TCP, etc...)
    -->
    <bean id="BMC_Dependency"
class="com.neebula.orchestrator.integration.cmdb.sn.handler.BMCToNeebulaMa
pping">
        <property name="handler"
value="com.neebula.orchestrator.integration.cmdb.sn.handler.RelationHandle
r"/>
        <property name="nameSpace" value="BMC.CORE"/>
        <property name="bmcCIT" value="BMC_Dependency"/>
        <property name="neebulaCIT"
value="APPLICATION_FLOW,INCLUSION,CLUSTER,RELATION_BSS_MEMBER"/>

        <property name="mappingRules">
            <map key-type="java.lang.String" value-
type="java.lang.String">
                <entry key="Name" value="${id}"/>
                <entry key="ShortDescription" value="'desc'"/>
                <entry key="sourceInstanceId" value="${src.id}"/>
                <entry key="destinationInstanceId" value="${target.id}"/>
                <entry key="reconciliationIdentity"
value="${src.id}+'::'+${target.id}"/>
            </map>
        </property>
    </bean>

    <bean id="BMC_Impact"
class="com.neebula.orchestrator.integration.cmdb.sn.handler.BMCToNeebulaMa
pping">
        <property name="handler"
value="com.neebula.orchestrator.integration.cmdb.sn.handler.RelationHandle
r"/>
        <property name="nameSpace" value="BMC.CORE"/>
        <property name="bmcCIT" value="BMC_Impact"/>
        <property name="neebulaCIT"
value="APPLICATION_FLOW,INCLUSION,CLUSTER,RELATION_BSS_MEMBER"/>
```

```
        <property name="mappingRules">
            <map key-type="java.lang.String" value-
type="java.lang.String">
                <entry key="Name" value="'ImpactOnly'"/>
                <entry key="ShortDescription" value="'desc'"/>
                <entry key="sourceInstanceId" value="${target.id}"/>
                <entry key="destinationInstanceId" value="${src.id}"/>
                <entry key="reconciliationIdentity"
value="${target.id}+'::'+${src.id}"/>
            </map>
        </property>
    </bean>

    <bean id="BMC_Impact2"
class="com.neebula.orchestrator.integration.cmdb.sn.handler.BMCToNeebulaMa
pping">
        <property name="handler"
value="com.neebula.orchestrator.integration.cmdb.sn.handler.RelationHandle
r"/>
        <property name="nameSpace" value="BMC.CORE"/>
        <property name="bmcCIT" value="BMC_Impact"/>
        <property name="neebulaCIT" value="IMPACT"/>
        <property name="mappingRules">
            <map key-type="java.lang.String" value-
type="java.lang.String">
                <entry key="Name" value="'ImpactOnly'"/>
                <entry key="ShortDescription" value="'desc'"/>
                <entry key="sourceInstanceId" value="${src.id}"/>
                <entry key="destinationInstanceId" value="${target.id}"/>
                <entry key="reconciliationIdentity"
value="${target.id}+'::'+${src.id}"/>
            </map>
        </property>
    </bean>

    <bean id="AliasRule" class="java.util.ArrayList">
        <constructor-arg>
            <list value-type="java.lang.String">

<value>${host.Short_host_name}+'::'+${shortType}+'::'+${label}</value>
            </list>
        </constructor-arg>
    </bean>

    <!-- defines the shortType values: mapping between neebula CIT and
atrium CIT-->
    <bean id="neebula2addm_mapping" class="java.util.HashMap">
        <constructor-arg index="0" type="java.util.Map">
            <map key-type="java.lang.String" value-
type="java.lang.String">
                <entry key="Websphere" value="IBM_WAS"/>
                <entry key="Weblogic" value="WEBLOGIC_AS"/>
                <entry key="Jboss" value="REDHAT_JBOSS"/>
                <entry key="Oracle iAS" value="ORA_APP_SVR"/>
                <entry key="IIS" value="MS_IIS_SRV"/>
                <entry key="iplanet" value="SUN1_HTTP"/>
                <entry key="Apache" value="APACHE_HTTP"/>
                <entry key="Tomcat" value="APACHE_TOMCAT"/>
                <entry key="IBM WebSphere MQ" value="IBM_QMGR"/>
                <entry key="Oracle DB" value="ORA_DB"/>
                <entry key="MS SQL server" value="MSSQL_DB"/>
                <entry key="PostgreSQL DB" value="unknown"/>
                <entry key="MySQLServer" value="unknown"/>
                <entry key="Sybase" value="SYB_DB"/>
                <entry key="DB2" value="DB2_DB"/>
                <entry key="Tomcat WAR" value="unknown"/>
                <entry key="Sybase" value="SYB_DB"/>
                <entry key="WeblogicModule" value="unknown"/>
                <entry key="Websphere EAR" value="unknown"/>
                <entry key="WMB Flow" value="unknown"/>
                <entry key="IBM WebSphere MQ Queue" value="IBM_QMGR_Q"/>
                <entry key="ActiveMatrix Business Works Process"
value="unknown"/>
                <entry key="Advanced Queue Queue" value="unknown"/>
                <entry key="EMS Queue" value="unknown"/>
                <entry key="Jboss module" value="unknown"/>
                <entry key="Oracle Weblogic JMS Queue" value="unknown"/>
                <entry key="MS SQL database" value="unknown"/>
                <entry key="Oracle DB schema" value="unknown"/>
                <entry key="Virtual Directory" value="unknown"/>
                <entry key="ActiveMatrix Business Works Process"
value="unknown"/>
                <entry key="ActiveMatrix Business Works"
value="TIBCO_BW"/>
                <entry key="Enterprise Message Service"
value="TIBCO_EMS"/>
            </map>
        </constructor-arg>
    </bean>

    <!-- defines common attributes values that will be set on each
update/insert action-->
    <bean id="common_attributes_values" class="java.util.HashMap">
        <constructor-arg index="0" type="java.util.Map">
            <map key-type="java.lang.String" value-
type="java.lang.String">
                <entry key="status" value="1"/>
                <entry key="submitter" value="Neebula"/>
                <entry key="datasetId" value="BMC.ASSET"/>

            </map>
        </constructor-arg>
    </bean>
</beans>
```