Using wildcard symbols in the advanced search bar



When you specify search criteria to find requests, you can use wildcard symbols as shown in the following table to indicate one or more characters:   
  
**Wildcards**

|  |  |
| --- | --- |
| **Use this wildcard:** | **To match these characters:** |
| % (Percent) | Matches any string of 0 or more characters. For example: **J%son** matches Jackson, Johnson, Jason, and Json. |
| \_ (Underscore) | Matches any single character. For example: **B\_b** matches Bab, Bob, and Bub. |
| - (Hyphen) | Indicates a range. Always use within square brackets ([]). |
| [ ] (Square brackets) | Matches any single character within a specified range or set. For example, **[a-f]** matches the range of characters a through f, and **[abcf]** matches the set of characters a, b, c, or f. |
| [^] (Square brackets with caret) | Matches any single character not within a specified range or set. For example, **[^a-f]** matches all characters except the range a through f, and **[^abcf]** matches all characters except a, b, c, or f. |

<https://docs.bmc.com/docs/display/public/ars81/Using+the+advanced+search+bar>

<http://rowshay.com/wp/archives/category/bmc-remedy/atrium-cmdb-1-x>

<http://rowshay.com/wp/archives/category/bmc-remedy>

Category Archives: Atrium CMDB 1.x

[Asset Management](http://rowshay.com/wp/archives/category/bmc-remedy/7-6-04/bmc-itsm-7-6-04/asset-management) [Atrium CMDB](http://rowshay.com/wp/archives/category/bmc-remedy/7-6-04/atrium-cmdb) [Atrium CMDB 1.x](http://rowshay.com/wp/archives/category/bmc-remedy/atrium-cmdb-1-x) [Atrium CMDB 2.x](http://rowshay.com/wp/archives/category/bmc-remedy/atrium-cmdb-2-x)

[Adding an Attribute to the CMDB via a CMDB Driver Script](http://rowshay.com/wp/archives/365)

April 3, 2014 – 8:11 PM

There are a number of ways to create an attribute in the CMDB. You could use the BMC Atrium Core Console, Remedy API, or the CMDBDriver.

The BMC Atrium Core Console offers a nice GUI interface, and is fairly slick when used via the mid-tier. I have noticed some issues using the mid-tier version vs the UserTool version. One issue that I’ve noticed is that sometimes custom fields do not show up when your viewing the classes. When viewed from the UserTool, everything is there. Another issue I have noticed is that the Menu Style field shows different results when compaired to what one see’s in the UserTool.

The Remedy API is a very powerfull feature, if used correctly, and you understand what your doing. I have built everything from forms, fields, active links, filters, and guides with the API and really, one takes for granted what the GUI does for you.

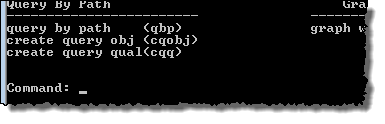
But the topic is the CMDBDriver tool. This is not a user friendly tool, but once you understand it, its great for automating alot of your work.

The one feature is the “execute” command. This allows you to execute a driver script that is basically a list of commands (like a macro) that you want the driver to execute. The following text is what is used to create a selection field, with 3 entries, on the BMC\_Application class:

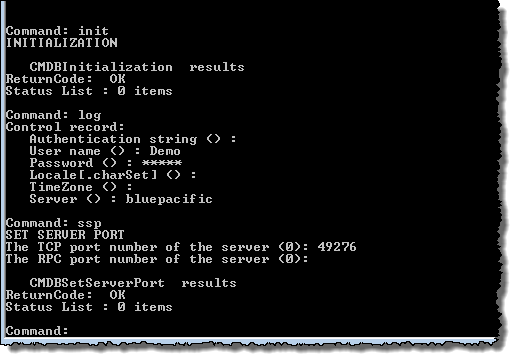
ca  
BMC.CORE  
BMC\_Application  
CUS\_BusinessGroup  
OS-0050368f00f39DRrUwYP1cqw21KL  
6  
900100102  
2  
2  
3  
Corporate Group  
12100  
Customer Group  
12200  
Enterprise Group  
12300  
F  
1  
8  
4  
CUS.CUSTOM  
0

These lines would sit in a text file, and the driver would execute the contents. The first line is the “CA” command which stands for Create Attribute. The following lines would be the supplied input for the process of manually creating an attribute.

The power of this feature is that you could maintain a number of these commands in the text file, allowing you to create as many fields, on as many forms as you like. For example, when I build out a new server environment, I can execute just one file that will add every custom field we added to the CMDB, bringing it upto speed with our existing environments.

[](http://rowshay.com/wp/wp-content/uploads/2014/04/cmdbdriver_ex_01.png)CMDBDriver – Start Up

 The next step is to enter the “init” command. This initizlizes the session, closing off any other session that you may have been already logged into. This is the first step that you must do when first opening the application. The next step is to enter the “log” command. Here you are logging in with your UserID, Password, and Server Name. Finally, if you are using a port number to connect to your server, enter the “ssp” command. This allows you to enter your TCP port number.

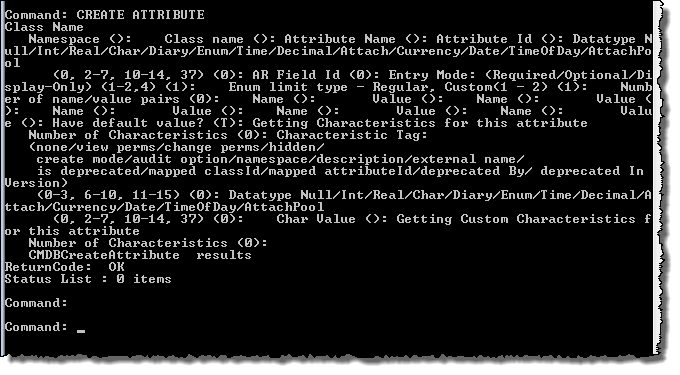
[](http://rowshay.com/wp/wp-content/uploads/2014/04/cmdbdriver_ex_04.png)Initialization and Login

The next step is to issue the “ex” command. This will prompt you for the command/driver file to load. Enter in the full path and filename. No need to enclose everything in quotes, as spaces dont matter.

[](http://rowshay.com/wp/wp-content/uploads/2014/04/cmdbdriver_ex_02.png)Execute Command

This is the part of the process that gets cluttered, and leaving you wondering whats going on. As the CMDBdriver executes the script, prompts for data are appearing on the screen. Normally you would manually enter the values, but our command file is supplying the data for us. I would have been nice if the values were actually displayed as the system exectued the file. Depending on how many attributes, which class your hitting, and the amount of data in your CMDB, this could take a few seconds or minutes to complete.

If you have multiple attributes being created, this screen will repeat the the cluttered block for each one. Once done, you should finally see the Command prompt come back.

[](http://rowshay.com/wp/wp-content/uploads/2014/04/cmdbdriver_ex_03.png)Execution of the command file

Over the years, dealing with problem tickets with BMC Support and talking with the application engineers, they all have said that they use the CMDBDriver for most of the work that they do. When you use the Atrium Core/Class Manager to add attributes, the CMDBDriver is actually being called in the background to do the work. It pulls its information from the OBJSTR:Class form (these are your Pending records). Once the driver completes the task, the record in the OBJSTR:Pending form is removed.

One nice thing about the CMDBDriver is that if something goes wrong, it tells your right away. If something goes wrong using the AtriumCore/ClassManager, you have to go in and start deleting records in various forms.

One last helpful hint if AtriumCore/ClassManager does not appear to be completing the building of your new field, you can grab the ‘Pending ID’ and from within the CMDBDriver, issue the “sync” command, which will prompt you for the Pending ID. This will force the attribute to complete for you, and the matching record in the OBJSTR:Pending form will disapear (As long as there are no errors in your system, but thats another issue to resolve). Note that the ‘Pending ID’ is not visible on the form, but you have to export the data, or manually type it out from the report option. On most of my sites, we have made this field visible (read only) so its easy to grab.

Tags [atrium](http://rowshay.com/wp/archives/tag/atrium), [Class Manager](http://rowshay.com/wp/archives/tag/class-manager), [cmdb](http://rowshay.com/wp/archives/tag/cmdb), [CMDBDriver](http://rowshay.com/wp/archives/tag/cmdbdriver), [driver](http://rowshay.com/wp/archives/tag/driver) | [Permalink](http://rowshay.com/wp/archives/365) | Comments Off on Adding an Attribute to the CMDB via a CMDB Driver Script

[Atrium CMDB 1.x](http://rowshay.com/wp/archives/category/bmc-remedy/atrium-cmdb-1-x)

[How can I rename a Class in CMDB 1.1](http://rowshay.com/wp/archives/204)

September 27, 2011 – 11:21 AM

As per Remedy’s knowledge base article KA337587, there is no feature to rename a class in CMDB 1.1. – that is, to rename the datastore forms and metadata.  
   
However, there is an easy configuration step to change the name of a class throughout the User Interface.  
   
Using Remedy User, query on the form named SHR:SchemaNames to find the record created for the class.  
Change the value of the ‘Proper Name’ field to the desired value and save the record.  
   
If you now access the class from Remedy Asset Manager, and bring up the AST (GUI form), it will have the new name.  
   
Note: If you change the name of a class using this procedure, you will need to update the existing data so that it will be available in queries.

– Open the form BMC:BMC\_AssetBase in Remedy Administrator

– Choose from the menu, Form –> Current View –> Fields in View, and add the field named UserDisplayObjectName(490021100)

to the view.

– Save the form.

– Login to Remedy User and query on the UserDisplayObjectName field for the old value of the Proper Name (before you changed it) to find the records that need to be updated.

– Update the records returned with the correct, updated value of the ‘Proper Name’ as set above.

[Permalink](http://rowshay.com/wp/archives/204) | Comments Off on How can I rename a Class in CMDB 1.1

[Atrium CMDB 1.x](http://rowshay.com/wp/archives/category/bmc-remedy/atrium-cmdb-1-x)

[Cannot correct an attribute definition if it fails to build in CMDB](http://rowshay.com/wp/archives/201)

September 27, 2011 – 11:14 AM

**PROBLEM: Cannot correct an attribute definition if it fails to build in CMDB 1.1:**

If you create an attribute on a class, and define it so that it will not build – for example if you include a hyphen in the attribute name – then you cannot open the attribute defintion and change the name. When you attempt to do this, the attirbute dialog opens up blank.

**Solution:**

This has been logged as a defect SW00224813, and below is the workaround for this issue.

Perform the following steps to delete the “Change Pending” attribute record from the back-end form, and then re-add the attribute correctly from the Class Manager

1. Search in the OBJSTR:AttributeDefinition form to locate the offending attribute. Delete it.
2. Search for the entry which is pending related to the class .Delete it.
3. Search in the OBJSTR:Class form for the pending entry related to that class and delete it.

Category Archives: Atrium CMDB 1.x

[Atrium CMDB 1.x](http://rowshay.com/wp/archives/category/bmc-remedy/atrium-cmdb-1-x)

[Understanding and Troubleshooting problems making changes via the Class Manager](http://rowshay.com/wp/archives/198)

September 27, 2011 – 11:07 AM

**The process flow of saving a class in the Class Manager of CMDB 1.1 is the following:**

1. When you click on the Save button on a class from the Class Manager, the Active Link OBJSTR:AL\_toCallFilterAPI2 does a push fields to create a new record in the OBJSTR:Pending form

2. The filter OBJSTR:Pend\_InitiateApplicationCommand fires on creation of a record in the OBJSTR:Pending form, and does a run process:

* Application-Command CMDB Sync-Meta-Data -o “$PendingID$

which is an internal process, which puts a record in the Application Pending form and sends a signal to 390308

3. The dispatcher receives that signal, and since arcmdbd process registers with the dispatcher for entries where category = CMDB, the dispatcher signals the arcmdbd process that it has work to do. (the same basic mechanism applies to all the application server processes – including Approval Server, Assignment Engine, and Brie)

4. The arcmdbd process receives the signal, deletes the record from the Application Pending form, and makes the CMDBSyncMetaData API call

5. The creation of the forms/workflow, and updates to metadata to be Active, occurs inside the CMDB API. Upon completion of the API, the CMDB API updates the status of the entry in OBJSTR:Pending form to “Complete”

6. The filter OBJSTR:Pend\_DeleteCompletedEntry is trigged by the status change, and sets ‘deleted’ field to True.

7. This triggers the filter OBJSTR:DeleteEntry to delete the record from the OBJSTR:Pending form

Note: The mechanism described above is specific to saving the class from the Class Manager – it is designed to give visibility in the client, for a process that fundamentally all happens on the server. If you make changes via cmdbdriver, no record is created in OBJSTR:Pending or Application Pending, and the dispatcher and arcmdbd process are not used.

**Troubleshooting problems where the class stays in “Pending” status:**

1. Click on the Change Pending record in the Class Manager and click on the View Log button. If there was an error in synching the class, the status will say Error and give an error message. If the status is “Pending”, it either means the arcmdbd process never issued the CMDBSyncMetaData API call, or the process is still happening within the CMDB API. (There isn’t a very good way to determine the latter – API and SQL logging on the server are really the only guage, but that is imperfect.)

2. Check if the arcmdbd server process is running, and if the dispatcher (arsvcdsp) server process is running on the server. If not, verify both are listed in the armontor.cfg/conf file.

3. Enable arcmdbd logging by adding CMDB-Debug: T to the ar.cfg/conf file (see page 40 of the CMDB 1.1 Install Guide) and restart the AR Server

4. If no log file is generated, comment out the line from armonitor.cfg/conf file and run that line from the console to verify the arcmdbd.

Category Archives: Atrium CMDB 2.x

[Asset Management](http://rowshay.com/wp/archives/category/bmc-remedy/7-6-04/bmc-itsm-7-6-04/asset-management) [Atrium CMDB](http://rowshay.com/wp/archives/category/bmc-remedy/7-6-04/atrium-cmdb) [Atrium CMDB 1.x](http://rowshay.com/wp/archives/category/bmc-remedy/atrium-cmdb-1-x) [Atrium CMDB 2.x](http://rowshay.com/wp/archives/category/bmc-remedy/atrium-cmdb-2-x)

[Adding an Attribute to the CMDB via a CMDB Driver Script](http://rowshay.com/wp/archives/365)

April 3, 2014 – 8:11 PM

There are a number of ways to create an attribute in the CMDB. You could use the BMC Atrium Core Console, Remedy API, or the CMDBDriver.

The BMC Atrium Core Console offers a nice GUI interface, and is fairly slick when used via the mid-tier. I have noticed some issues using the mid-tier version vs the UserTool version. One issue that I’ve noticed is that sometimes custom fields do not show up when your viewing the classes. When viewed from the UserTool, everything is there. Another issue I have noticed is that the Menu Style field shows different results when compaired to what one see’s in the UserTool.

The Remedy API is a very powerfull feature, if used correctly, and you understand what your doing. I have built everything from forms, fields, active links, filters, and guides with the API and really, one takes for granted what the GUI does for you.

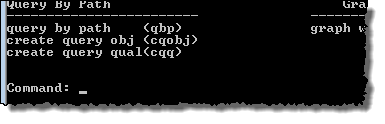
But the topic is the CMDBDriver tool. This is not a user friendly tool, but once you understand it, its great for automating a lot of your work.

The one feature is the “execute” command. This allows you to execute a driver script that is basically a list of commands (like a macro) that you want the driver to execute. The following text is what is used to create a selection field, with 3 entries, on the BMC\_Application class:

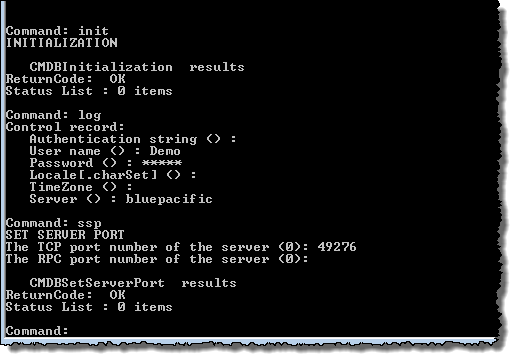
ca  
BMC.CORE  
BMC\_Application  
CUS\_BusinessGroup  
OS-0050368f00f39DRrUwYP1cqw21KL  
6  
900100102  
2  
2  
3  
Corporate Group  
12100  
Customer Group  
12200  
Enterprise Group  
12300  
F  
1  
8  
4  
CUS.CUSTOM  
0

These lines would sit in a text file, and the driver would execute the contents. The first line is the “CA” command which stands for Create Attribute. The following lines would be the supplied input for the process of manually creating an attribute.

The power of this feature is that you could maintain a number of these commands in the text file, allowing you to create as many fields, on as many forms as you like. For example, when I build out a new server environment, I can execute just one file that will add every custom field we added to the CMDB, bringing it upto speed with our existing environments.

[](http://rowshay.com/wp/wp-content/uploads/2014/04/cmdbdriver_ex_01.png)CMDBDriver – Start Up

 The next step is to enter the “init” command. This initizlizes the session, closing off any other session that you may have been already logged into. This is the first step that you must do when first opening the application. The next step is to enter the “log” command. Here you are logging in with your UserID, Password, and Server Name. Finally, if you are using a port number to connect to your server, enter the “ssp” command. This allows you to enter your TCP port number.

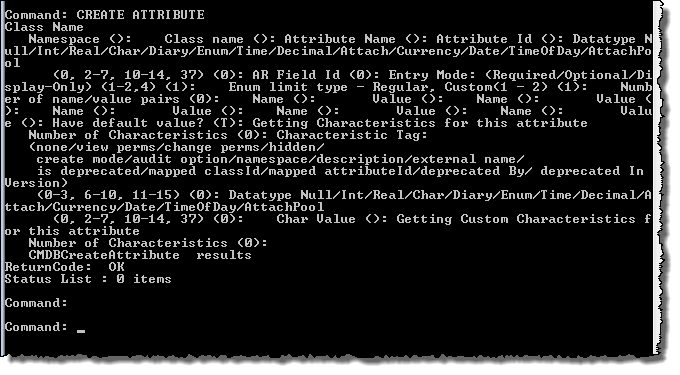
[](http://rowshay.com/wp/wp-content/uploads/2014/04/cmdbdriver_ex_04.png)Initialization and Login

The next step is to issue the “ex” command. This will prompt you for the command/driver file to load. Enter in the full path and filename. No need to enclose everything in quotes, as spaces dont matter.

[](http://rowshay.com/wp/wp-content/uploads/2014/04/cmdbdriver_ex_02.png)Execute Command

This is the part of the process that gets cluttered, and leaving you wondering whats going on. As the CMDBdriver executes the script, prompts for data are appearing on the screen. Normally you would manually enter the values, but our command file is supplying the data for us. I would have been nice if the values were actually displayed as the system exectued the file. Depending on how many attributes, which class your hitting, and the amount of data in your CMDB, this could take a few seconds or minutes to complete.

If you have multiple attributes being created, this screen will repeat the the cluttered block for each one. Once done, you should finally see the Command prompt come back.

[](http://rowshay.com/wp/wp-content/uploads/2014/04/cmdbdriver_ex_03.png)Execution of the command file

Over the years, dealing with problem tickets with BMC Support and talking with the application engineers, they all have said that they use the CMDBDriver for most of the work that they do. When you use the Atrium Core/Class Manager to add attributes, the CMDBDriver is actually being called in the background to do the work. It pulls its information from the OBJSTR:Class form (these are your Pending records). Once the driver completes the task, the record in the OBJSTR:Pending form is removed.

One nice thing about the CMDBDriver is that if something goes wrong, it tells your right away. If something goes wrong using the AtriumCore/ClassManager, you have to go in and start deleting records in various forms.

One last helpful hint if AtriumCore/ClassManager does not appear to be completing the building of your new field, you can grab the ‘Pending ID’ and from within the CMDBDriver, issue the “sync” command, which will prompt you for the Pending ID. This will force the attribute to complete for you, and the matching record in the OBJSTR:Pending form will disapear (As long as there are no errors in your system, but thats another issue to resolve). Note that the ‘Pending ID’ is not visible on the form, but you have to export the data, or manually type it out from the report option. On most of my sites, we have made this field visible (read only) so its easy to grab.

Tags [atrium](http://rowshay.com/wp/archives/tag/atrium), [Class Manager](http://rowshay.com/wp/archives/tag/class-manager), [cmdb](http://rowshay.com/wp/archives/tag/cmdb), [CMDBDriver](http://rowshay.com/wp/archives/tag/cmdbdriver), [driver](http://rowshay.com/wp/archives/tag/driver) | [Permalink](http://rowshay.com/wp/archives/365) | Comments Off on Adding an Attribute to the CMDB via a CMDB Driver Script

[Atrium CMDB 2.x](http://rowshay.com/wp/archives/category/bmc-remedy/atrium-cmdb-2-x)

[Troubleshooting Building Classes in CMDB 2.0](http://rowshay.com/wp/archives/194)

September 27, 2011 – 10:59 AM

**The process flow of saving a class in the Class Manager of CMDB 2.0 is the following:**

1. When you click on the Save button on a class from the Class Manager, the Active Link OBJSTR:ClassDef\_SaveChanges\_CallFilterApi does a push fields to create a new record in the OBJSTR:Pending form

2. The filter OBJSTR:Pend\_InitiateApplicationCommand fires on creation of a record in the OBJSTR:Pending form, and does a run process:

* Application-Command CMDB Sync-Meta-Data -o “$PendingID$

which is an internal process, which puts a record in the Application Pending form and sends a signal to 390308

3. The dispatcher receives that signal, and since arcmdbd process registers with the dispatcher for entries where category = CMDB, the dispatcher signals the arcmdbd process that it has work to do. (the same basic mechanism applies to all the application server processes – including Approval Server, Assignment Engine, and Brie)

4. The arcmdbd process receives the signal, deletes the record from the Application Pending form, and makes the CMDBSyncMetaData API call

5. The creation of the forms/workflow, and updates to metadata to be Active, occurs inside the CMDB API. This logic occurs in the server side files – cmdbsvr20.dll / libcmddbsvr20.so / libcmdbsvr20.sl and cmdb20api.dll / libcmdb20api.so / libcmdb20api.sl. Upon completion of the API, the CMDB API updates the status of the entry in OBJSTR:Pending form to “Complete”

6. The filter OBJSTR:Pend\_DeleteCompletedEntry is trigged by the status change, and sets ‘deleted’ field to True.

7. This triggers the filter OBJSTR:DeleteEntry to delete the record from the OBJSTR:Pending form

Note: The mechanism described above is specific to saving the class from the Class Manager – it is designed to give visibility in the client, for a process that fundamentally all happens on the server. If you make changes via cmdbdriver, no record is created in OBJSTR:Pending or Application Pending, and the dispatcher and arcmdbd process are not used.

**Troubleshooting problems where the class stays in “Pending” status:**

1. Click on the Change Pending record in the Class Manager and click on the View Log button. If there was an error in synching the class, the status will say Error and give an error message. If the status is “Pending”, it either means the arcmdbd process never issued the CMDBSyncMetaData API call, or the process is still happening within the CMDB API.

2. Check if the arcmdbd server process is running, and if the dispatcher (arsvcdsp) server process are running on the server. If not, verify both are listed in the armontor.cfg/conf file. This server process is responsible for telling the CMDB to build the class.

3. Enable arcmdbd logging by adding CMDB-Debug: T and CMDB-Debug-Level: 5 to the ar.cfg/conf file and restart the AR Server. If no log file is generated, comment out the line from armonitor.cfg/conf file and run that line from the console to verify the arcmdbd is running properly.

4. Verify the shared libraries for CMDB 2.0 are specified in the ar.cfg/conf file, and if not, correct the issue and restart the AR Server:

* Load-Shared-Library: cmdbsvr20.dll,or libcmdbsvr20.sl, or libcmdbsvr20.so
* Load-Shared-Library-Path: <path to the above file>

5. Enable API logging in BMC Remedy Administrator. Below is an example of how the CMDB API calls appear in the ARAPI log:

<API > <TID: 0000000010> <RPC ID: 0000088636> <Queue: Admin > <Client-RPC: 390696 > <USER: Remedy Application Service > /\* Sat Apr 01 2006 21:19:47.9234 \*/+SYNC CMDBSync — from Unidentified Client (protocol 12) at IP address 172.23.33.40

In CMDB 2.0, the CMDB API calls and their results are written to the ARServer API log. The AR API calls that implement the work are also listed afterward, so it is easy to correlate which AR API calls are issues to perform the work of the CMDB.

[Permalink](http://rowshay.com/wp/archives/194) | Comments Off on Troubleshooting Building Classes in CMDB 2.0

[Atrium CMDB 2.x](http://rowshay.com/wp/archives/category/bmc-remedy/atrium-cmdb-2-x)

[Duplicate CIs even when values of Serial Number, HostName, and Domain match, Oracle database.](http://rowshay.com/wp/archives/144)

September 3, 2011 – 8:07 PM

**Solution:**

The identification rules used in Topology Discovery 1.4 and Configuration Management Configuration Discovery Integration for CMDB 7.1.1 use qualifications such as the following:

(‘SerialNumber’ = $SerialNumber$ AND ‘HostName’ = $HostName$ AND’Domain’ = $Domain$)

AND ( ‘SerialNumber’ != $\NULL$ AND ‘HostName’ != $\NULL$ AND ‘Domain’ != $\NULL$)

AND ( ‘SerialNumber’ != “” AND’HostName’ != “” AND’Domain’ != “”)

where the three meta-clauses are:

<attributes match>

AND <attributes are not NULL>

AND <attributes are not empty string>

On a SQL Server database, the third clause is required since NULL and empty-string are two different values at the database, and the qualification needs to check for both values to ensure the attribute value is suitable for using for identification. So the qualifications are correct if the AR Server is using any database other than Oracle.

On an Oracle database, the empty string is the same as NULL. This means the second meta-clause above is sufficient to check for both conditions. It also means the third meta-clause will always return a FALSE result, since

!= NULL

will always return a result of FALSE. (It is improper syntax for checking for NULL)

**Workaround:**

If the database used by ARSystem/CMDB is Oracle, the workaround is to update the Identification rules to remove the clauses which check for Empty String. For example, in the qualication above, it would remove the third meta-clause:  
AND ( ‘SerialNumber’ != “” AND ‘HostName’ != “” AND ‘Domain’ != “”)

————————

Though this problem with Reconciliation can occur with other Reconciliation Jobs, the two Jobs where it has been identified are:

Configuration Discovery Reconciliation Process

BMC Topology Import – Identification, Merge, and Purge

Defects have been submitted to correct these issues in a future version of Topology Discovery and Configuration Management Configuration Discovery Integration for CMDB products:

SW00277889 Defect against CDI

QM001535938 Defect against Topology Discovery

Defect SW00294045 has been logged against AR System, indicating it should provide a way to search for (NULL OR Emptry String) in a way that will work for all databases.

Category Archives: Atrium CMDB 2.x

[Atrium CMDB 2.x](http://rowshay.com/wp/archives/category/bmc-remedy/atrium-cmdb-2-x)

[Verifying CMDB Meta Data in CMDB 2.x](http://rowshay.com/wp/archives/134)

September 3, 2011 – 7:34 PM

It is recommended that your CMDB meta data be correct prior to upgrading the CMDB or making class changes when a superclass is in Change Pending.A quick test of the health of the CMDB is to open the Class Manager, scroll through the list of CI classes to make sure they all have a status of “Active”, toggle the ‘Class Type’ to “Relationship”, and scroll through the list of relationship classes to verify they all have a status of “Active”. However, this check will not capture all kinds of incomplete or incorrect meta data.  
Below are steps to verify the CMDB meta data is correct.

1. Check the arerror.log for errors starting the CMDB Engine. Search for the string “cmdbEngine” to find relevant errors. Validation of the metadata occurs on startup. Below is an example error message.  
Mon Dec 11 16:10:18 2006 390600 : Error encountered while initializing(Initialization routine) the shared library (ARERR 9755)  
Mon Dec 11 16:10:18 2006 bmc.CMDB.cmdbEngine

Mon Dec 11 16:10:18 2006 390600 : The specified super class with given classId is not found. : Class ID: OB005056C00008PlV3RQU3oZAAHO8C, Super Class ID: OB005056C00008F1V3RQd#IYAAD90C (ARERR 120132)

2. Query on the Class form to verify all classes are Active  
– Launch and login to Remedy User as an administrator user.

– Choose File –> Open Object List, and open the Class form ( OBJSTR:Class ) in a Search window

– Choose View –> Advanced Search Bar

– In the Advanced Search Bar at the bottom of the Window, specify the query:

‘Pending ID’ != “0” OR ‘System Status\*’ != “Active” OR ‘OSStatus’ != “Active” OR ‘Record Status’ != “Active” OR ‘Status’ != “Active” OR ‘Session Status’ != “Just Created”

If this query returns any rows, then these should be investigated as classes which have not built or deleted properly.

3. Query on the Class form to determine the number of classes

– Assuming the above test finds no records, perform an Advanced Search on the Class form where:

‘Namespace\*’ = “bmc.CORE”

should return 92 rows. Querying where:

‘Namespace\*’ = “bmc.CORE.CONFIG”

should return 10 rows.

Note: Using a QBE search on Namespace = bmc.CORE will return the sum of the above – 102 rows.

Additional classes can be added via the Class Manager, so this is not an absolute confirmation of the correct number of classes, but the result of the query is expected to be as listed above.

4. Query on the Attribute Definition form to verify all attributes are Active  
– Launch and login to Remedy User as an administrator user.

– Choose File –> Open Object List, and open the Attribute Definition form ( OBJSTR:AttributeDefinition ) in a Search window

– Choose View –> Advanced Search Bar

– In the Advanced Search Bar at the bottom of the Window, specify the query:

‘OSStatus’ != “Active” OR ‘Pending ID’ != “0”

If this query returns any rows, then these should be investigated as attributes which have not built or deleted properly. Export the value of the ‘ForeignKeyID (Class ID)’ field to determine the class and check whether it must be rebuilt to update the attribute definition.

5, Run cmdbdriver from the server, and issue the glc command to get the list of classes.

– Navigate to the CMDB/bin directory on the server

– Run the cmdbdriver program. (On UNIX, you must first set the library path to include the directory, see KM-000000024454)

– At the command prompt, issue the log command, and respond to the prompts to login.

– At the command prompt, issue the ssp command if necessary to set the specific port (if ARServer is not running on portmapper)

– At the command prompt, issue the init command to initialize the connection

– At the command prompt, issue the glc command to get a list of classes. Hit Return at all of the prompts to accept the defaults

The desired output is a list of classes. If cmdbdriver gives an AR 91 RPC call failed message – as in the example below – this indicates the CMDB Engine is not loaded or running. In the particular case, the reason it was not loaded was that validation of the CMDB meta data failed, so use the steps above to find the meta data that is incorrect.

Command: initINITIALIZATION CMDBInitialization resultsReturnCode: OKStatus List : 0 items Command: logControl record: Authentication string () : User name () : Demo Password () : Locale[.charSet] () : TimeZone () : Server () : ltangha-ple-13 Command: sspSET SERVER PORTThe TCP port number of the server (0): 7011The RPC port number of the server (0): CMDBSetServerPort resultsReturnCode: OKStatus List : 0 items  
Command: glc

GET LIST CLASSNamespace ():Class Relationship Name Namespace (): Class name ():Super Class Name Namespace (): Class name ():Getting querying Characteristics for this class Number of Characteristics (0):Retrieve hidden classes? (F): CMDBGetListClass resultsReturnCode: ERRORStatus List : 1 itemsStatus Struct : Message type : ERROR Message number : 91 Message: RPC call failed Appended: RPC: Procedure unavailable

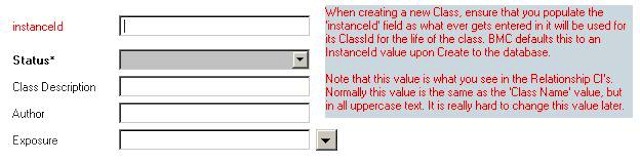
[Permalink](http://rowshay.com/wp/archives/134) | [Comment (1)](http://rowshay.com/wp/archives/134#comments)

[Atrium CMDB 2.x](http://rowshay.com/wp/archives/category/bmc-remedy/atrium-cmdb-2-x)

[When I create a new class in the CMDB Class Manager, the ClassId shows up in the format of an InstanceId. How do I make it show up like “BMC\_MyNewClass”?](http://rowshay.com/wp/archives/129)

September 3, 2011 – 7:24 PM

This is an easy fix, if you havn’t yet created your Class. From the AdminTool, you need to open the form called “**OBJSTR:ClassDefinition**“. Locate the ‘**InstnaceId**‘ field and make it visible on the form (I positioned it over the ‘Status\*’ field. See image below) and then save your changes.

[](http://rowshay.com/wp/wp-content/uploads/2011/09/InstanceId_ClassManager1.jpg)

Now, when ever you goto create a new class, you just need to enter in the value you want to use. If you leave this field blank, the system will populate this with an InstanceId value. Note that the field Id for this field is 179, which means that remedy’s workflow will automatically populate this field with an instanceId if no value is specified. This goes for any form that you create in 7.x and up.

Because this field is hidden from users, it always gets populated with the instanceId. I am not sure why BMC/Remedy did it this way as it causes alot of issues for people down the line.

The next question is, what if you have created the class already. Well, if there is no data in the class as of yet, you just need to delete the class and then add it back in, but this time populate the ‘InstanceId’ field with the name you want to give it.

If you have data in the form, then the only thing you can do is first export the data, drop the form and then recreate it. Once this has been done, you just reimport the data back in. Note that you have to update the exported data to reflect the new classId.

I have completed this process for about 18 Classes, both in BMC\_BaseElement and BMC\_BaseRelationship. Attached is a document that outlines the steps that I use. This document is only a rough copy as I am still updating it with more details.

--- >Last modified date and Last completed date will be same almost in Problem investigation console

--- >Submit date, reported date, create date and Last modified date will be same when first time I create a ticket (in Date/system tab in Problem investigation form) e.g. (5/12/2016 8:01:35 AM)

---> Submitter, creator, requestor and last modified by will be logged in user name or the

User who has raised/logged the request whatever relationships we create that we can see in Relationships tab in Problem management/Incident management

--- >The change request includes planning all the changes approved for implementation, targeting dates, and estimating the risks and costs. You can perform the following actions to the change request during this stage. Enter information into all the required fields on the Dates tab to complete the planning stage. 1. You can build tasks at any change status except Completed. 2. Change Manager reviews and validates the change plans. 3. Change enters the Implementation Approval phase. Each level of approvers must review the change and approve it. 4. Change is included in the Forward Schedule of Changes (FSC) Calendar.

--- >Measurement

Some availability measurements that may be included in SLA:

Mean-Time-Between-Failure (MTBF): elapsed time between services gets up and down.

Mean-Time-To-Repair (MTTR): elapsed time to repair a configuration item or IT service.

Mean-Time-Between-System-Incidents (MTBSI): elapsed time between detection of two consecutive incidents.

Mean-Time-To-Restore-Service (MTRS): elapsed time from the detection of an incident until it gets up.

Based on terminology above,

MTBSI = MTBF + MTRS

, and availability can be calculated by

A = MTBF / (MTBF + MTTR)

--- >Change reasons can be:

New functionality

Fix/repair

Maintenance

Upgrade

Project

Business as usual

939 Error

The AR System Plug-In server is not responding. Cannot connect to the system at this time.

Contact your AR System Administrator for assistance.

There was an RPC time-out when connecting to the arplugin service. To continue, make

sure that the arplugin service and AR System plug-ins are running.

39 Error

Filter or escalation set field process timed out before completion.

A filter or escalation was performing a Set Fields action using the option to run a process and

return a value, but the process was not completed within the time-out interval specified for

filter processes.

The administrator has control over the time-out setting for the process. It can be configured,

within BMC Remedy Administrator, to be from one to 20 seconds with a default of five

seconds.

52 Warning

The field is a core system field and cannot be changed.

You attempted to modify the contents of a core system field (Request ID, Create Date, Last

Modified By, Last Modified Date, or Status History). These fields are managed by AR System

and cannot be changed.

68 Warning

RPC environment variable is out of valid range (390600, 390603, 390619 - 390669, 390680 -

390695). NOTE: The value 390603 and the range 390619 - 390669 are specialty servers and

may have restricted functionality.

The value of the ARRPC environment variable is not a legal value. To use the default RPC

socket, clear this variable. To use the ARRPC environment variable, this value must be

390600 (the admin daemon), 390603 (the escalation daemon), 390619 (the flashboard

daemon), 390620 through 390634 (fast daemon), 390635 through 390669 (list daemon), or

390680 through 390694 (private daemon).

The environment variable is being ignored, and processing continues.

75 Warning

No floating write license tokens are available. Currently accessing the system in read-only

mode. License will upgrade when a token is available.

You are assigned a floating write license, but there are no floating write license tokens

available at this time. You are allowed access to the database for read-only use. The system

will try to upgrade your license type when a token is available.

76 Note

A write token has become available and has been allocated to you—access has been upgraded

to write access.

You have previously received Warning 75. A floating write token has become available, and

it has been allocated to you. You now have full read and write access within your

permissions

93 Error

Time-out during data retrieval due to busy server—try the operation again.

A time-out occurred while information was being retrieved from the data dictionary,

because the server was busy performing other operations. Try the operation again when

there is less activity on the server.

94 Error

Time-out during database search—consider using more specific search criteria.

A time-out occurred while data was being retrieved from the server. This error is often

caused by poorly specified search criteria. Review and, if necessary, modify your search

criteria, and try the operation again.

236 Error

Value specified for the license time-out must be 1 or greater (in hours).

The value specified for the license time-out value was 0 or a negative number. This value

must be greater than or equal to 1 hour. The value remains unchanged (defaults to 2 if

nothing is set).

2404 Error

Unable to connect to any servers.

While trying to log in and connect to the specified servers, BMC Remedy Administrator

failed to successfully connect to a server. This can occur when the server is down, when the

network is too slow (leading to a time-out), or if you are not an administrator or

subadministrator user for the server.

2533 Warning

A time-out occurred while copying form <form\_name> to <form\_name>. The operation

most likely succeeded but the server is still busy.

This warning occurs when using the Save Form As command to copy a form. The server is

busy and has “timed out.” Your form has probably been copied, but verify at a later time

(when the server is not busy) that the copy operation has succeeded.

Release management usually begins in the development cycle with requests for changes or new features. If the request is approved, the new release is planned and designed. The new design enters the testing or quality assurance phase, in which the release is built, reviewed, tested and tweaked until it is ultimately accepted as a release candidate. The release then enters the deployment phase, where it is implemented and made available. Once deployed, the release enters a support phase, where bug reports and other issues are collected; this leads to new requests for changes, and the cycle starts all over again.

Below are the lead times –

• Normal – 5+ working days

• Expedited – 2 to 5 working days

• Emergency – 1 - 48 hours

• Standard – 5 working hours (pre-approved, template changes only).

https://docs.bmc.com/docs/display/servicedesk81/Sample+data+-+Incident+templates

https://docs.bmc.com/docs/display/public/srm81/System+SRDs+shipped+with+the+product

<https://docs.bmc.com/docs/display/public/servicedesk81/Configuring+Problem+Management>