Tivoli Business Systems Manager
1.5-BSM-0029 Release Notes

Tivoli Business Systems Manager

1.5-BSM-0029 Release Notes

Tivoli Business Systems Manager Patch 29 Release Notes

Contents

Preface

Tivoli Business Systems Manager provides a robust, system management functionality. In support of the operational perspectives of that functionality, Tivoli Business Systems Manager installs components on both the OS/390 and NT platforms.

Accessing Publications Online

Publications in the product libraries are included in PDF or HTML formats, or both, on the product CD. To access the publications using a Web browser, open the infocenter.html file, which is located in the appropriate publications directory on the product CD.

When IBM publishes an updated version of one or more online or hardcopy publications, they are posted to the Tivoli Information Center. You can access updated publications in the Tivoli Information Center from the following Tivoli Customer Support Web site:

http://www.tivoli.com/support/documents/

The Tivoli Information Center contains the most recent version of the books in the product library in PDF or HTML formats, or both. Translated documents are also available for some products.

Note: If you print PDF documents on other than letter-sized paper, select the **Fit to page** check box in the Adobe Acrobat Print dialog (which is available when you click **File --> Print**) to ensure that the full dimensions of a letter-sized page are printed on the paper you are using.

Providing Feedback about Publications

We are very interested in hearing about your experience with Tivoli products and documentation, and we welcome your suggestions for improvements. If you have comments or suggestions about our products and documentation, contact us in one of the following ways:

- Send an e-mail to pubs@tivoli.com.
- Complete our customer feedback survey at the following Web site:

  http://www.tivoli.com/support/survey/

Contacting Customer Support

If you have a problem with any Tivoli product, you can contact Tivoli Customer Support. See the *Tivoli Customer Support Handbook* at the following Web site:

http://www.tivoli.com/support/handbook/

The handbook provides information about how to contact Tivoli Customer Support, depending on the severity of your problem, and the following information:

- Registration and eligibility
- Telephone numbers and e-mail addresses, depending on the country in which you are located

- What information you should gather before contacting support

Chapter 1. What's in this Document

This document provides important information about Patch 1.5-BSM-0029 for Tivoli Business Systems Manager Version 1, Release 5. These release notes contain the most current information about the product and include all documentation from previous v1.5 patches. Therefore, this document takes precedence over all other v1.5 documentation.

Please review these notes thoroughly before installing or using this product.

Each chapter in this document describes the changes to Tivoli Business Systems Manager v1.5 documentation as a result of Patch 29. Changed documentation includes the following:

- Tivoli Business Systems Manager Patch 24 Release Notes (1.5-BSM-0024.pdf) (page 3)
- Tivoli Business Systems Manager v1.5 - Installation and Configuration Guide (page 9)
- Tivoli Business Systems Manager v1.5 - Administration Guide (page 15)
- Tivoli Business Systems Manager v1.5 - Console User's Guide (page 21)
- Tivoli Business Systems Manager v1.5 - CA7 Release Notes (page 25)

Chapter 2. Changes to the Patch 24 Release Notes (1.5-BSM-0024.pdf)

Additions to "Installing the History Server" section

To the "Installing the History Server" section on page 21 of the Patch 24 Release Notes, please add the following after step 14:

Note: *History server* refers to the secondary server that contains the History database, to which events from the Tivoli Business Systems Manager primary database server are forwarded. The data contained on the History database is used to run Historical Reporting.

Primary server refers to the server to which events are forwarded from a variety of data sources. This data is used for real-time monitoring of an enterprise's critical resources.

Attention: Insure you have the most recent copy of the Object and Meta databases copied from your primary database server.

If installing the Two-Server BCP History setup, also perform the following:

1. Apply <install_dir>\SQL\SchemaTypes.sql to the master database.

2. Apply <install_dir>\SQL\SchemaTypes.sql to the model database.

3. Apply <install_dir>\SQL\SchemaTypes.sql to the tempdb database.

4. Apply <install_dir>\SQL\MasterDBChanges.sql to your master database located on the History server. If the SQL server is installed on a drive other than D:\, modify the script as necessary to point to the appropriate drive where the SQL server is installed.

5. Apply <install_dir>\SQL\SchemaMessages.sql to your master database located on the History server.

Changes to DB2 Subsystem State Event Flow

On page 38, please replace the section entitled, "DB2 Subsystem State Event Flow" with the following:

To provide better processing performance of DB2 subsystem down and up events, the processing behavior is the following:

On receiving a DB2 subsystem down event for a DB2 subsystem that is not currently in a DB2_Down state:

1. The DB2 subsystem gets a red critical DB2 subsystem down message that is propagated.

2. Upon receiving the red critical DB2 Subsystem Down Message, any descendant of that DB2 subsystem that participates in a Line of Business View will get a red scheduling violation (SCHV) exception that is then propagated in an upwards direction within that LOB.

On receiving a DB2 subsystem up event:

1. The DB2 subsystem gets a green medium DB2 subsystem up message that is propagated.

   If the state is defined for the descendent's subsystem up message then the state is directly applied to the descendent.

2. Any descendant of that DB2 subsystem that participates in a Line of Business View will get a clearing exception for each of its current exceptions.

Addition to the rmLOB.sh script

APAR OW52836 added the -k option (which deletes a Line of Business) to the rmLOB.sh script. Please replace the rmLOB script on page 49 with the following:

rmLOB.sh

Purpose

Removes LOBs (Line Of Business) from Tivoli Business Systems Manager (created through addLOB.sh or addpvToLOB.sh).

Synopsis

sh rmLOB.sh [**-S** *server*] [**-U** *userid*] [**-P** *password*] **-l** *lob_name* [**-p** *parent_lob_name*] [**-k** *parent_lob_key*]

Arguments

-S *server*
> Specifies the name of the SQL server machine

-U *user* Specifies the SQL userid to the SQL server machine

-P *password*
> Specifies the SQL password to the SQL server machine

-l *lob_name*
> Specifies the name of the Tivoli Business Systems Manager LOB to remove

-p *parent_lob_name*
> Specifies the name of the parent LOB under which the current LOB is created. Default is top level LOBC (Line of Business Object Container)

-k *parent_lob_key*
> Specifies the key of the parent under which the current LOB is created.

Usage Notes

Note: This command must be run from the MKS korn shell.

Typically in Tivoli Business Systems Manager, when you delete a LOB from the Tivoli Business Systems Manager client, it will be re-created at the next heartbeat of any of the software components under it. This script lets you permanently delete those dynamically created LOBs. It removes the LOB(s) with name lob_name from the parent LOB parent_lob_name (or parent LOBKey parent_lob_key). The parent LOB can be specified either using the -p or -k options, whichever identifies the parent uniquely. Also note that the -k option takes precedence over -l if both are specified.

Note: Before deleting a LOB, you need to delete all objects contained within the LOB designated for deletion. The script will return an error if any child objects are contained within the selected LOB.

-p which takes the parent lob name is optional. If it is omitted, then LOBC, which is the top-level LOB container is taken as parent.

-S, -U, and/or **-P** may be set from environment variables so you don't need to specify them on the command line. The variable names are tbsm_server, tbsm_userid, and tbsm_password and can be set from an MKS korn shell as follows:

```
$ export tbsm_server=sql1.abc.com
```

You may specify some options via environment variables and others via the command line. Specifying an option on the command line takes precedence over the respective environment variable.

Run updateLOBViews.sh following successful completion of rmLOB.sh to update the LOB View to remove the various filtering conditions which affect the LOB View.

The output data of this script is directed into rmLOB.out file.

See also addLOB.sh, addpvToLOB.sh.

Examples

The following example removes LOB test_lob that lives under ptest_lob as long as there are no instances/LOBs under test_lob:

```
sh rmLOB.sh -S sql1.abc.com -U user1 -P secret -l test_lob -p ptest_lob
```

The next example removes LOB test_lob from LOBC (LOBC->test_lob):

```
sh rmLOB.sh -S sql1.abc.com -U user1 -P secret -l test_lob
```

Additional CommonListener Tables

[Please add the following to the CommonListener Tables section starting on page 5.]

CL_AutoPlacement

Table Name

CL_AutoPlacement

Database

Object

Purpose

Used to map objects to an Enterprise (ENT) using CommonListener InstrumentationID

InstrumentationID is the id used by Instrumentation while registering with CommonListener. All valid InstrumentationIDs can be found in instrId column of CL_Registration table in Object database.

It is strongly recommended that the table's row with the highest seq value specify a _in value of '%'. This assures that you will have at least one match for an Enterprise (a default). To avoid confusion, we also recommend that no other rows in the table specify a _in value of '%'.

There is no special tool to edit this table. We suggest that the Enterprise Manager that comes with Microsoft SQL Server be used to update it.

Columns

- **_in:** The name of the InstrumentationID to associate with an enterprise name. This can be wildcarded in SQL-style. An example might be '%NetView%' to match any InstrumentationID that has "NetView" string in it.

- **_out:** The name of the enterprise to use when the InstrumentationID matches the _in column of the row. The enterprise object with this name gets created if one does not already exist.

- **seq:** A sequence number. This is a key and is used to prioritize the search for the enterprise. The row with the lowest seq value whose _in column matches the InstrumentationID is the "best" match during a search.

Examples

Name	_in	_out	seq
CLInstrumentation_to_E NT	%[Tt][Nn][Gg]%	TNG	1
CLInstrumentation_to_E NT	%[Nn][Ee][Tt][Vv][Ii][Ee][W w]%	NetView	2
CLInstrumentation_to_E NT	%	Your Company, Inc.	3

CL_Options

Table Name

CL_Options

Database

Object

Purpose

This is a table that allows customers to enable or disable certain features in CommonListener. It is structured in such a way that adding new options is very easy. Rather than adding a new column to the table, each option is a separate row in the table and you search with the option name as the key to find the value of the option.

There is no special GUI interface to manage this table. To set the value of an option, we suggest using the GUI of Enterprise Manager or by running similar command as the following one through the Query Analyzer (both of which come with Microsoft SQL Server)

```
update CL_Options set _Value = '24' where _Option = 'cleanup interval in
hours'
```

Columns

- **_Option:** The name of the option. This is a key
- **_Value:** The value of the option. This can be any character string but it will probably have certain requirements based on the purpose of the option. For instance, an option might only expect values of "yes" and "no" or numbers between 1 and 10.

Known Options

Option = ' cleanup interval in hours', Value should be a positive integer greater than zero: This option controls how often the CommonListener tables get cleaned up.

It is used by the CLCleanUp scheduled job to clean-up transaction (CL_Stage) and status (CL_Status) data from tables

CL_Severities

Table Name

CL_Severities

Database

Object

Purpose

This table maps a threshold severity (from TEC, NetView or TNG) to an alert state and priority pair that is used in Tivoli Business Systems Manager to represent status of a resource.

Below is a summary of the Tivoli Business Systems Manager alert states and priorities:

- **alert states:** green (1), yellow (2), red(3)
- **priorities:** ignore(5), low(4), medium(3), high(2), critical(1)

Columns

- **name:** Name of the product that uses severity to represent event status. (for example, tec)
- **severity:** A text string that describes different levels of severity. (for example, normal, critical, fatal, etc.)
- **excp_alertstate:** The exception alert state (1-3) to be associated with severity
- **excp_priority:** The exception priority to be (1-5) associated with severity
- **mesg_alertstate:** The message alert state (1-3) to be associated with severity
- **mesg_priority:** The message priority (1-5) to be associated with severity

Chapter 3. Changes to the Installation and Configuration Guide

Event Handler Server Installation

On page 96 of the Tivoli Business Systems Manager v1.5 Installation and Configuration Guide, under "Install Event Handler Server (Configuration 1)," please add a fifth bulleted item:

- * Generate a output log of the entire process in <TivoliManager>/logs/makemvs_*

Converting from a Two-Server Linked History Setup to a Two-Server BCP Maintained History Setup

Prerequisite: Please read and implement the "Additions to Installing the History Server" section on page 3 of this guide before following the below instructions.

Please add the following information to the History server section:

These instructions assume that a linked server for history has been installed, and that the existing historical events residing on the History server will be retained. It also assumes that the Object and Meta databases on the Tivoli Business System Manager SQL Server are being backed up periodically, and that there are successful backup files of the Object and Meta databases present.

The following variables are defined:

- <HistoryServer> - The host name of the server that contains the History database
- <HistsaUID> - The sa User ID for the History database
- <HistsaPWD> - The sa Password for the History database
- *History server* refers to the secondary server that contains the History database, to which events from the Tivoli Business Systems Manager primary database server are forwarded. The data contained on the History database is used to run Historical Reporting.
- *Primary server* refers to the server to which events are forwarded from a variety of data sources. This data is used for real-time monitoring of an enterprise's critical resources.

1. **Export the existing historical events:** The events residing on the History server are exported by running the following BCP command:

   ```
   bcp "EventHistory..event_table" out <drive>\dbEventHistory.txt -w
   -q -S<HistoryServer> -U<HistsaUID> -P<HistsaPWD>
   -e<drive>:\dbEventHistoryERR.txt
   ```

   Use the appropriate values for <HistsaUID>, <HistsaPWD>, and <HistoryServer> on the History server database.

2. **Uninstall the linked server:** Run the following Kornshell command on the Database server:

   ```
   sh HistoryServerUninstall.ksh -N <HistoryServer> -R <HistsaUID>
   -W <HistsaPWD>
   ```

Use the appropriate values for `<HistsaUID>`, `<HistsaPWD>`, and `<HistoryServer>` on the History server database.

3. **Install the History server using the BCP approach:** After you install the History server using this approach you can perform the following tasks:

   • Move BCP events from the primary Database server to the History server at five (tunable) minute intervals.

   • A periodic restoration of the Tivoli Business Systems Manager Object & Meta databases on the History server from the primary Database server.

   • Perform cleanup on events transferred from the Object database on the primary Database server to the History server.

   Prerequisites:

   • A periodic backup of the Object and Meta databases on the primary Database server.

   • You have installed the Internet Information Server 4.0 or greater and Tivoli Business Systems Manager Historical Reporting System on the History server.

   • The network connections and the necessary permissions between the primary Database server and the History server are configured properly. Ensure SQLserver and SQLagent start/login accounts on the primary Database server have the write permission to the History server.

   • Tivoli Business Systems Manager is installed on the History server.

   To install the History server:

   a. Insert the Tivoli Business Systems Manager Base Services CD in the History server computer. Double-click the **BaseServices** folder.

   b. Double-click the **Setup.exe** icon.

   c. When the System File Upgrade dialog opens, read the text and click Next.

   d. From the Choose Destination Location dialog, we recommend that you place these files in a Temporary directory. Click the **Browse** button to select a directory (or use the default directory). Once a desired directory is selected or to use the default, click **Next**.

   e. When the System File Upgrade is complete, click **Finish**. If you are prompted to reboot the computer, do so now.

   f. From the Setup Type dialog, select Custom and click Next.

   g. From the Select Components dialog, select the following components:
      • Workstation Program Files
      • Help Files
      • Reporting System
      • Active Documentation
      • Tools and Utilities
      • SQL Extension Files

   h. Click Next.

   i. Enter the hostname of the local server machine. (By default, the hostname should be in the hostname field, If it is not, enter it.) Click Next.

   j. Enter the hostname of the local server machine, the SQL Administrator Username, and SQL Administrator Password. Click Next.

   k. Enter the name of the Application server machine. Click Next.

   l. The Select Program Folder dialog box opens. Select Program Folder to host all applications. Click Next.

m. From the Setup Complete dialog, when asked if you want to restart the computer select No and click Finish.

To Configure the History Server:

a. Stop auto-execution of xp_rule_startpulsing by using the following SQL command from the TivoliManager\sql directory:

`Apply hist_packup.sql:`

For example,

`$isql -U<user>-P<password>-S<server>-dmaster -ihist_packup.sql`

Use appropriate values for `<user>`, `<password>`, and `<server>` on the History Server database. Restart the SQL Server service.

b. Create the following databases as defined in the Microsoft SQL Server documentation if they do not already exist:

- History
- Object
- Meta
- Webserver - only if not previously created during the History server installation

c. Create database backup files.

You can either use the SQL database backup to create the files, or use the touch command to create them. The content of the file is not important, but the existence is important. The files are overwritten at a later time.

- Create the Meta database backup file, `Meta.BAK` under `mssql7\Backup.`
- Create the Object database backup file, `Object.BAK` under `mssql7\Backup.`

d. Apply `event_hist.sql`:

`$isql -U<user>-P<password>-S<server>-dObject -ievent_hist.sql`

Use the appropriate values for `<user>`, `<password>`, and `<server>` on the History server database.

e. Create tasks using the `schedule_task_bcp_hist.sql`:

This SQL creates the following two tasks:

- `Copy Backups to History Server` – Periodically copies latest version of Object and Meta database backup files to History SQL Server.
- `Update Object and Meta Databases` - Restores the Object and Meta database on the History server using the backups from the primary Database server.

1) Edit `schedule_task_bcp_hist.sql`:

- Replace "XXXX" with the primary database server hostname
- Replace "YYYY" with the primary database server network drive-database install root directory such as "e$/mssql7". Remember to use the forward slash character '/'.
- Replace "HHHH" with the History database server host name
- Replace "IIII" with the History database server network drive- Tivoli Business Systems Manager install directory such as "c$/TivoliManager/bin"
- Replace "ZZZZ" with the History database server network drive-database install root directory "d$/mssql7"

- Replace "GGGG" with the History server local drive-database install root directory such as "d:\mssql7"
- Save to the original file name.
- Make sure that no Tivoli Business Systems Manager jobs are currently running.

2) Apply `schedule_task_bcp_hist.sql` using the Query Analyzer.

To Configure the Reporting System: Use the Reporting System Database Configuration program found in the Tivoli Manager programs folder to set up the DefaultData and LiveData databases with the following parameters:

Parameter	Default Data	LiveData	WebServer
Connection Name	DefaultData	LiveData	WebServer
Driver	(SQL Server)	(SQL Server)	(SQL Server)
Server Name	\<History Server Name\>	\<Tivoli Business Systems Manager SQL Server Name\>	\<History Server Name\>
User ID	sa	sa	sa
Password	\<sa_histserver\>	\<sa_servername\>	\<sa_histserver\>
Database Name	Object	Object	WebServer

4. **Import the historical events:** The saved events from step 1 are imported into the new History server setup by running the following BCP command:

```
bcp "History..eventhist" in <drive>\dbEventHistory.txt -w

-b10000 -q -S<HistoryServer> -U<Hist_saUID> -P<Hist_saPWD>

-e<drive>:\dbEventHistoryERR.txt
```

Use the appropriate values for `<HistsaUID>`, `<HistsaPWD>`, and `<HistoryServer>` on the History server database.

5. **Configure the primary Database server:**

Note: The following variables are defined:

- \<PrimaryServer\> - The host name of the server that contains the primary Tivoli Business Systems Manager database that receives events from multiple sources

- \<PrimarysaUID\> - The sa user ID for the primary database

- \<PrimarysaPWD\> - The sa password for the primary database

The following steps enable you to configure the primary Database server so it will work with the Reporting system:

a. Apply `event_live.sql`:

```
$isql -U<PrimarysaUID>-P<PrimarysaPWD>-S<PrimaryServer>-dObject -ievent_live.sql
```

Use appropriate values for \<PrimarysaUID\>, \<PrimarysaPWD\>, and \<PrimaryServer\> on the primary Database server.

b. Create tasks from `schedule_task_bcp_live.sql`:

The `schedule_task_bcp_live.sql` creates the following two scheduled tasks:

- `Move events to History Database` - Moves entries in the event table every five minutes.

- `Delete old entries in event table` - Deletes old entries in the event table. Old entries are defined as events that have been transferred from the primary database to the History database. The default time for the job that deletes old entries is currently seven days. This time can be modified in the SQL job `Delete old entries in event table`.

1) Edit `schedule_task_bcp_live.sql`

   - Replace XXXX with the primary database server host name
   - Replace HHHH with the history database server host name
   - Replace FFFF with the primary database server database network root install drive directory, such as e$\mssql7"
   - Save to the original file.

2) Apply `schedule_task_bcp_live.sql` using the Query Analyzer.

Additional Reporting System Configurations: The following configuration steps are optional. Only use them if you need to change the default schedule of the jobs. The modification of the jobs can be done through SQL Server Enterprise Manager by locating the task under Management-->SQL Server Agent-->Jobs.

- The `Move events to History Database` task is started at five minute intervals.

- The SQL job `Copy Backups to History Server` has been moved off the primary Database server and now runs from the History database server. The reason for this move is that it considerably reduced the time necessary to copy backups from the primary Database server to the History server.

  From the History server, run `Copy Backups to History Server` after the primary Database server back up is finished. Check that copies of these files have arrived on the History server. Note: The job may indicate a false failure because of there may be missing Meta2 and or Object2 backup sets. This is acceptable as long as there are valid Meta_*.*, or Meta.*, and valid Object_*.* or Object.* backup files present. In this case, the most recent backup of the Meta and Object files will be copied.

- After the `Copy Backups to History Server` is finished, and it has been verified that the backup copies of the Meta and Object databases have arrived on the History server, run `Update Object and Meta Databases`.

- Adjust start time of `Copy Backups to History Server`. Because the Object database can be extremely large, it is highly recommended that `Copy Backups to History Server` be scheduled off-hours. In addition, the interval of this job should be determined based upon how dynamic the monitored configuration is. The less dynamic, the longer the interval, e.g, once every other day, once a week, etc.

- Adjust `Move events to History Database` task Interval.

Adjust start time of `Delete old entries in event table`. The default is to keep the events which have been copied to the History server on the live server for seven days. This retention period may be adjusted if necessary by adding a "@daystokeep=nnn" parameter following the `asisp_deleteoldevent` command in the task's properties.

Chapter 4. Changes to the Administration Guide

Please add the following to the Tivoli Business Systems Manager v1.5 Administration Guide.

Update to gemdmmap.sh

Please add the following to the gemdmmap.sh information on page 286 in Appendix F of the Tivoli Business Systems Manager v1.5 Administration Guide.

The maximum length of -p (product) is 64 and that of -v (version) is 16. However, the maximum length of Tivoli Business Systems Manager class label (which is <product> <version>) is 40. For the label not to be truncated, [(length of product)+(length of version)+1] should be less than or equal to 40.

Tivoli Business Systems Manager will automatically truncate the data if it exceeds the size limit.

Updates to List Views

In order to improve performance and have newer object types available, some views have been updated to take advantage of the database cache introduced in Patch 1.5-BSM-0024. It is important to note that the cache is maintained by scheduled task `Update ObjPathCache`, which is scheduled by default to run every 15 minutes. New objects, such as those inserted from the GUI, or discovered by batch or event, will not display in these views until `Update ObjPathCache` runs. Customers may adjust this job to run more frequently, but we do not recommend running it more often than 5 minute intervals.

The views that take advantage of this cache are:

• VIEW --> Events (Otherwise known as the 'event viewer', available only in the Java console.)

• VIEW --> Batch Management Summary

• VIEW --> CICS Status Summary

• VIEW --> DB2 Resources

• VIEW --> Disk Volumes

• VIEW --> Disk Devices

• VIEW --> Logical Volumes

• VIEW --> IMS Resources

• VIEW --> Managed Objects

• VIEW --> Catalogs

Displaying Default Icons

When an icon cannot be successfully retrieved, the Java Console is designed to display a default icon. There are three scenarios in which a default icon will be displayed.

1. **The icon does not exist in the database:** This information will be noted in the Java Console log. An example is provided below:

```
14:50:17.846 com.tivoli.tbsm.ui.util.UIImageLoader getCIDImageIcon AWT-
EventQueue-0
```

Image not found for image id/size=[18_t.gif] using default instead.

2. **The icon is corrupt in the database and therefore cannot be retrieved properly:** This information is logged in both the Java Console (see scenario 1) as well as the Java Console Server. Below is an example of the Java Server logging the problem:

```
14:50:17.836 com.tivoli.tbsm.server.DBIconRow actionErrorCalled RMI TCP
Connection(7)-146.84.80.11
```

Error decoding bitmap image id=[18] image filename=[cid_arcf_16.gif]

3. **The icon exists in the database but is not of the requested size:** This problem is most likely encountered when using the gemimageimport.sh file. By default, the gemimageimport.sh converts a GIF to a BMP file and creates two icons - a large and a small icon image. These images are placed in the database. When running the Java Console, the appropriate image size will be displayed based on the requested icon size.

Therefore, only images that are of size "large" and "small" will appear for resource icons within the Java Console when using the gemimageimport.sh script to import icons into the database; icon sizes of "tiny" and "medium" will display the default icon. This information is logged in the Java Console log as stated in scenario 1.

How to Configure the Managed Object View

Like many Tivoli Business Systems Manager list views, by default the Managed Objects view can show a maximum of 2000 records based on the maximum 10,000 records retrieved from the database, but it is possible for you to configure the maximum values. This is not recommended for large databases because a serious performance impact to the view and other Tivoli Business Systems Manager users might occur. If you wish to configure this despite the possible impact, there is a stored procedure to follow.

Procedure Name

_SetSystemConfiguration

Setting up the maximum retrieved records

The following is an example of the _SetSystemConfiguration command used for setting the maximum retrieved records to 10,000.

EXEC _SetSystemConfiguration

@groupname = '_GetManagedObjects',

@propertyname = 'DESCENDENTS_LIMIT',

@propertyvalue = '10000',

@propertydatatype = 'INT'

Setting up the maximum displayed records

The following is an example of the _SetSystemConfiguration command used for setting the maximum displayed records to 2000.

```
EXEC _SetSystemConfiguration

@groupname = '_GetManagedObjects',

@propertyname = 'RETURN_LIMIT',

@propertyvalue = '2000',

@propertydatatype = 'INT'
```

There are 4 parameters for this procedure, however the only one you should change is propertyvalue. For the other 3 parameters, please use exactly what is shown here. If there is no value or a value less than 0 for either retrieved records or displayed records, then it will set to the default value.

If the retrieved records number is less than the displayed records number, it will be set as the same as the displayed records number.

Steps

1. On NT SQL server, open a command window.
2. Type: isql -S<servername> -U<user> -P<password>

   For example: `C:\temp>isql -Smyhost -Usa -Psa_myhost`
3. Type (in an isql environment):

   ```
   EXEC _SetSystemConfiguration
   @groupname = '_GetManagedObjects',
   @propertyname = 'DESCENDENTS_LIMIT',
   @propertyvalue = '10000',
   @propertydatatype = 'INT'
   EXEC _SetSystemConfiguration
   @groupname = '_GetManagedObjects',
   @propertyname = 'RETURN_LIMIT'
   @propertyvalue = '2000',
    @propertydatatype = 'INT'
   GO
   ```
4. Verify that the correct value is set in the database.

   Type (in an isql environment):

   ```
   select * from SystemConfiguration
   GO
   ```

   It will display:

id	groupname	propertyname	propertyvalue	propertydatatype
10	_GetManagedObjects	DESCENDENTS_LIMIT	10000	INT
11	_GetManagedObjects	RETURN_LIMIT	2000	INT
5. Type (in an isql environment, to exit from the isql environment):

   ```
   quit
   ```

How to Configure the "Business Impact" View

By default the "Business Impact view" can show a maximum of 6-level generations for a selected object, but you can configure this number. There is a stored procedure that lets you modify the number of generations that can be viewed.

Procedure Name

```
_SetSystemConfiguration
```

Steps

To modify the maximum number of generations returned by a call to Business Impact View:

```
EXEC _SetSystemConfiguration

@groupname = '_GetDependencies',

@propertyname = 'DEFAULT_GENERATION',

@propertyvalue = '9',

@propertydatatype = 'INT'
```

There are 4 parameters for this procedure, however the only one you should change is propertyvalue. For the other 3 parameters, please use exactly what is shown here. If there is no value or a value less than 0 for this parameter, it will be set to the default value.

1. On NT SQL server, open a command window.

2. Type `'isql -S<servername> -U<user> -P<password>'`

   Example: `C:\temp>isql -Smyhost -Usa -Psa_myhost`

3. The following example sets the maximum number of generations returned by Business Impact View to 9. Type (in isql environment), the following information:

   ```
   EXEC _SetSystemConfiguration

   @groupname   = '_GetDependencies',

   @propertyname   = 'DEFAULT_GENERATION',

   @propertyvalue = '9',

   @propertydatatype = 'INT'

   GO
   ```

   Note: `@Propertyvalue` can be set to any positive integer. Setting `@Propertyvalue` to null, or a negative number, results in the system default, which is 6. Warning: setting the `@PropertyValue` to 0 will set the maximum generations to 0, which will only display the selected object from where the Business Impact View is launched.

4. Verify the correct value is set in the database.

   Type (in isql environment):

   ```
   select * from SystemConfiguration

   GO
   ```

   It will display like:

   ```
   id   groupname        propertyname       propertyvalue propertydatatype
   ```

```
10  _GetDependencies  DEFAULT_GENERATION 9          INT
```
The value customer set for @propertyvalue will list under propertyvalue.

5. Type (in isql environment):

   ```
   quit
   ```

   to exit from the isql environment.

How to Configure HyperView

Procedure Name

```
_SetSystemConfiguration
```

Steps

The following example sets the maximum number of records returned by HyperView to 5000:

```
EXEC _SetSystemConfiguration

@groupname = '_GetDescendents',

@propertyname = 'RETURN_LIMIT',

@propertyvalue = '5000',

@propertydatatype = 'INT'
```

There are 4 parameters for this procedure, however the only one you should change is propertyvalue. For the other 3 parameters, please use exactly what is shown here. If there is no value or a value less than 0 for this parameter, it will be set to the default value.

1. On NT SQL server, open a command window.

2. Type `'isql -S<servername> -U<user> -P<password>'`

   Example: `C:\temp>isql -Smyhost -Usa -Psa_myhost`

3. Type (in isql environment), the parameter @propertyvalue can be a positive value other than 6:

   ```
   EXEC _SetSystemConfiguration
   @groupname = '_GetDescendents',
   @propertyname = 'RETURN_LIMIT',
   @propertyvalue = '5000',
   @propertydatatype = 'INT'
   GO
   ```

4. Verify the correct value is set in the database.

   Type (in isql environment):

   ```
   select * from SystemConfiguration
   GO
   ```

   It will display like:

id	groupname	propertyname	propertyvalue	propertydatatype
10	__GetDescendents	RETURN_LIMIT	5000	INT

The value you set for @propertyvalue will list under propertyvalue.

5. Type (in isql environment):

```
quit
```

to exit from the isql environment.

Chapter 5. Changes to the Console User's Guide

[Please add the following to the Console User's Guide.]

Simplified Warning Exceptions Handling

The Tivoli Business Systems Manager Event Viewer integrates the presentation of messages and exceptions with their associated objects -- all within a line-of-business context. However, production deployments have found that the overall usability of the Event Viewer has been negatively affected by the volume of events which an operator may often need to process. These large numbers of events often represent "noise" that can impair the operator's ability to understand the current state of the enterprise.

To avoid this performance issue, customers now have the ability to "bulk process" events directly from the Event Viewer in the following ways:

- **Take Ownership:** All of the selected events are assigned to a particular owner.

- **Transfer Ownership:** Operators can bulk-transfer the events they own to the next operator. An example usage of this would be a shift change at an operations center.

- **Closeout:** This facilitates the handling of a large number of relatively insignificant events. Operators can bulk-select any number of events and from within a single dialog close them all out (even if the events aren't currently owned).

Note: All of these functions implicitly apply to object-level ownership, as opposed to event-level ownership.

Steps

1. The user selects one or more events, brings up the context menu, and chooses either Take Ownership, Transfer Ownership, or Closeout. Since the operation may take some time, a progress bar is displayed along with a Cancel button which may be used to halt this operation.

2. A results dialog box appears which indicates how many were selected, how many were processed as expected, how many were not processed successfully (and why), and how many could not be processed due to various conditions.
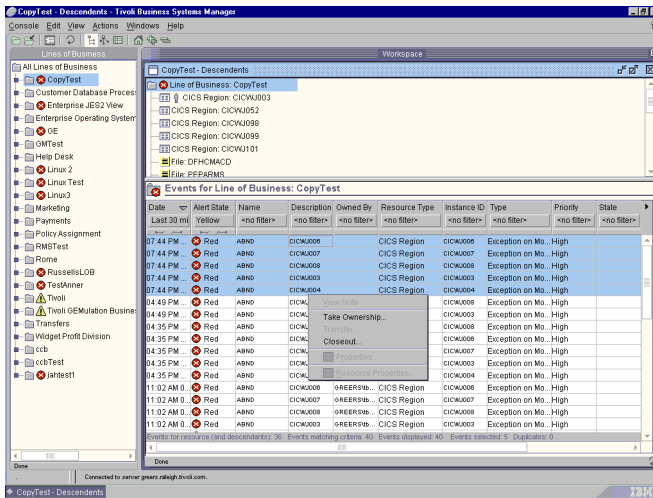
Figure 1.

Take Ownership

The Take Ownership dialog is used to efficiently take ownership of one or more selected events. The operator specifies the following:

- Owner
- Subject
- Description
- Checkbox: Close
- Resolution (if Close is selected)

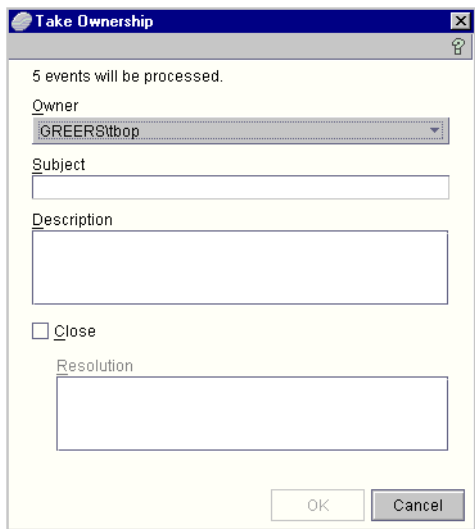Subject, and Description are required fields. Resolution is also required if Close is checked.

Figure 2. Take Ownership dialog box

Transfer Ownership

The Transfer Ownership dialog is used to efficiently re-assign the ownership of one or more selected events to another operator. The operator simply specifies:

• Owner

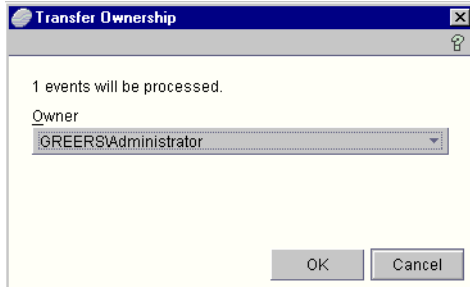and the system reassigns the selected events.

Figure 3. Transfer Ownership dialog box

> **Note:** If the system determines one or more notes aren't fully spanned by all the selected events, it prompts the user as to how to handle those notes, giving the operator the choice of proceeding anyway, skipping these notes, or canceling the operation altogether.

Closeout

The Closeout dialog is used to efficiently close-out the notes associated with one or more selected events. The operator specifies

• Resolution

and the selected events are closed out with the specified resolution. If the selected event includes events which aren't owned, then Subject and Description are both displayed and both are required.
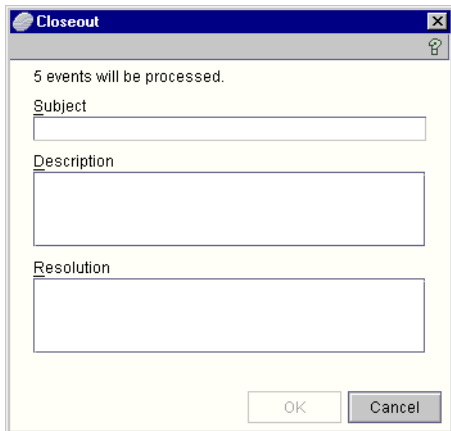

Figure 4. Closeout dialog box

> **Note:** If the system determines one or more notes aren't fully spanned by all the selected events, it prompts the user as to how to handle those notes, giving the operator the choice of proceeding anyway, skipping these notes, or canceling the operation altogether.

Chapter 6. Changes to the CA7 Release Notes

Please add the following CA7 monitoring updates to the CA7 Release Notes.

NT rules for CA7 Bulk and Event Data

The CA7 NT object filters and key specifications are changed. You can use the naming conventions of schedule set, schedules, and job name to exclude object creation during bulk discovery and event discovery. It is also possible to use the naming conventions of schedule set, schedules, and job name to specify key jobs during bulk discovery and event discovery. This key job specification on Windows is in addition to the mainframe-based key file. All the naming conventions are applied to the bulk or event data exclusively by their priority from high to low.

In addition, events and records can only be evaluated once. Therefore, if more than one naming convention is used to qualify a record or event, the convention with the highest priority is the one that is used for evaluation.

By default, there are no naming conventions defined for the CA7 data source. You can view, add, delete, and modify the entries with a sql query as described below.

Viewing the List of Existing Filters

To view the list, run the SQL query: `_GetBJDiscoverFilters`

How to Configure Exclude Filters

The following section provides some examples of how to configure exclude filters, which allow you to filter out incoming events or discovery records based on a naming convention.

To exclude all schedules and jobs under a schedule set 'Test':

```
EXEC _SetBJDiscoverFilter @name_schedule_set = 'Test',

@data_source = 'CA7',

@add_entry = 1,

@excluded = 1,

@enabled = 1,

@priority = 1
```

To exclude all jobs under specific scedule 'Test1' in the enterprise:

```
EXEC _SetBJDiscoverFilter @name_schedule = 'Test1',

@data_source = 'CA7',

@add_entry = 1,

@excluded = 1,

@priority = 1
```

To exclude all jobs with a specific name pattern in the enterprise:

```
EXEC _SetBJDiscoverFilter @name_job = '%Test%',

@data_source = 'CA7',

@add_entry = 1,

@excluded = 1,

@priority = 1
```

The above example excludes all jobs with the text "Test" located in the job name.

Key Flags

Jobs with specific schedule set name, schedule name, or job name patterns as defined previously, can be marked as a key job in the enterprise.

To mark all jobs with a schedule job name that contains the text "Text_key" in the name of the job (remember, SQL is case sensitive):

```
EXEC _SetBJDiscoverFilter @name_job = '%Test_key%',

@data_source = 'CA7',

@add_entry = 1,

@key_flag = 1,

@priority = 1
```

Removing a Filter

All filters can be removed using the isql appropriate commands. The following example removes all the filters associated with job names that contain the text "Test_key" within it:

```
EXEC _SetBJDiscoverFilter @name_job = '%Test_key%',

@data_source = 'CA7',

@add_entry = 0,
```

Disabling a Filter

All filters can be disabled using the isql appropriate commands. The following example disables the filters associated with job names that contain the text "Test_key" within it:

```
EXEC _SetBJDiscoverFilter @name_job = '%Test_key%',

@data_source = 'CA7',

@enabled = 0,
```

Note: Customers should use as few types of filters as possible. The more filter types that are used, the more system resources are required.

Note: `SetBJDiscoverFilter` method can add or remove entry. The updateable columns are "enable", "priority", "excluded", and " keyflag".

CA7 Event and Object Cleanup

Excluded jobs are defined as batch jobs that do not receive events for a number of days and who managed to not appear in the most recently loaded bulk discovery file. In the previous release of CA7 support, excluded jobs used to be automatically deleted if they did not appear in two subsequent runs of a bulk discovery. This has been changed so that they do not get deleted automatically. The SQL job `Cleanup Batch Jobs` can be scheduled to clean both excluded jobs and CA7 Audit tables. CA7 audit tables contain events that meet no specific monitoring criteria and therefore not posted to any jobs, but retained. The excluded jobs and old events can be cleared on a configurable number of days. By default no cleanup will be performed for CA7. You can view, add, update, and delete the cleanup configuration as demonstrated in the following examples:

Viewing cleanup configuration

```
EXEC _GetBatchCleanUPs
```

Adding a cleanup configuration entry

The following example adds a cleanup configuration entry for deleting CA7 objects which have not received any events for the last 30 days and have not be loaded in the last running of the bulk discovery file. It will also clean events from the CA7 audit tables that are older than 15 days:

```
EXEC _SetBatchCleanUP @cid = 'ROOT',

@id = 0,

@data_source = 'CA7',

@obj_days_to_keep = 30,

@audit_table = 'ca7_audit_events',

@audit_days_to_keep = 15,

@enabled = 1,

@priority = 1,

@add_entry = 1
```

Updating a cleanup configuration entry

The following example takes the cleanup configuration entry used in the previous example and updates it so it will only clean events from the CA7 tables that are older than 60 days:

```
EXEC _SetBatchCleanUP @cid = 'ROOT',

@id = 0,

@data_source = 'CA7',

@obj_days_to_keep = 30,

@audit_table = 'ca7_audit_events',

@audit_days_to_keep = 60,
```

```
@enabled = 1
```

Deleting a cleanup configuration entry

The following example deletes the configuration entry used in the previous two examples by setting the @add_entry parameter to a value of 0:

```
EXEC _SetBatchCleanUP @cid = 'ROOT',

@id = 0,

@data_source = 'CA7',

@add_entry = 0
```

Tivoli