HOPEX Administration

HOPEX Aquila 6.2

MEGA

a Bizzdesign company

Administrator Guide

Contents

ABOUT HOPEX ADMINISTRATION

Hopex products can be used on a stand-alone workstation or in a configuration including dozens of users.

This guide is for the person responsible for repository administration. When there are only a few users, administration is usually done by one of the users. In such cases, it mainly consists of carrying out regular backups and reorganizing repositories when required.

Chapter Repositories cover most administration requirements of a structure with only a few users. In a structure with many users, the administrator must respond to more specific requirements, which are detailed in the following chapters.

Most of the functions described here can be used by the repository administrator, whatever the products enabled through his/her security key. However, certain functions are only available with specific technical modules (**Hopex Power Studio** or **Hopex Power Supervisor**). These are indicated by a note in the text.

Hopex administration is managed from the **Hopex Administration** application (Windows Front-End) "Administration.exe" or from the **Hopex Administration** desktop (Web Front-End). Some actions can also be performed from the **Hopex** application (Windows Front-End) "Hopex.exe" or from certain **Hopex** desktops (Web Front-End).

The **Hopex** administration applications (Windows Front-End and Web Front-End) are designed for **Hopex** administrators: they are used to manage environments, repositories, users, etc.

A **Hopex** installation can contain a large number of environments, repositories and users. To facilitate their management, **Hopex Administration** provides all the key concepts and tools required for their administration in a unified hierarchical structure.

PRESENTATION OF THIS GUIDE

This guide concerns **Hopex** administration in the **Hopex Administration** application (Windows Front-End).

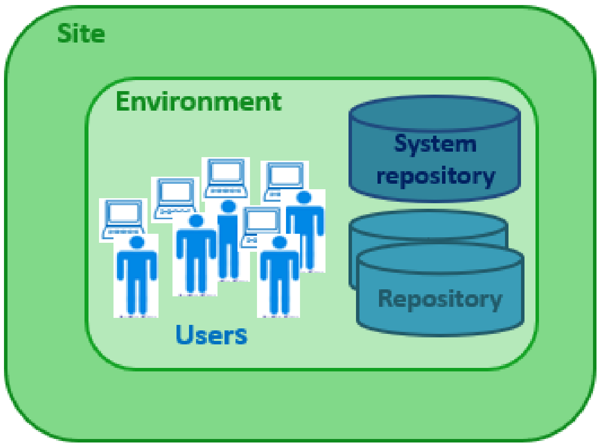
> ☛ *To manage users, permissions, workspaces, and perform **Hopex** administration tasks in **Hopex Administration** desktop (Web Front-End), see the Hopex Administration - Supervisor Web guide.*

It includes the following chapters:

- Connecting to Hopex Administration: how to access **Hopex Administration**.
- Repositories: to create, save, restore, check, reorganize, copy and move repositories.
- Environments: to create, back up, restore, check, copy and move an environment.
- Supervision and Events: to supervise events with the **Hopex Server Supervisor** tool.
- Scheduling (Scheduler): to create triggers and schedule jobs.
- Objects: Advanced administration functions available with:
  - the **Hopex Power Studio** technical module to extract objects
  - the **Hopex Power Supervisor** technical module to compare objects in two repositories.
- Data Writing Access: to set up management of organized projects in the form of data writing access, from the **Hopex Power Supervisor** technical module.
- Data Reading Access: to install a confidentiality strategy using a reading access diagram and access areas.
- Command File Syntax: description of the syntax used in command files.
- Frequently Asked Questions: answers to frequently asked questions and some administration tips.
- Glossary: definition of the main terms used in this guide.

HOPEX STRUCTURE

Some basic knowledge is required to understand the architecture and operation of **Hopex**.



Hopex is organized on the following levels:

- ***site***
  A site groups together everything that is shared by all **Hopex** users on the same local network: the programs, standard configuration files, online help files, standard shapes, workstation installation programs, and version upgrade programs.
- ***environment***
  An environment groups a set of users, the repositories on which they can work, and the system repository. It is where user private workspaces, users, system data, etc. are managed.
- ***user***
  A user is a person (or person group) with a login and a profile. A user:
  - has a specific workspace in each repository.
  - can connect to a repository from all workstations connected to the environment in which this repository is referenced.
  - has a specific configuration and is authorized to access specific product functions and repositories in the environment.

CONNECTING TO HOPEX ADMINISTRATION

The **Administration** application (Windows Front-End) is the **Hopex** administration application accessible from the Windows desktop. This application contains the tools required for management of environments and its repositories. It is also used to manage scheduling and data accesses (writing access as well as confidentiality using reading access management).

The **Administration** desktop (Web Front-End) is the **Hopex** administration application available via an internet browser. This application is used to manage users (persons, person groups) and repositories (workspaces, locks, repository health, repository activity, repository snapshots). This application also provides access to tools (Scheduler, Import/Export of command files) .

☛ *The **Administration** desktop (Web Front-End) is described in the Web Administration Guide.*

The points covered here are:

✓ Accessing Hopex Administration
✓ Connecting to an Environment

Accessing Hopex Administration

The **Administration** application is the **Hopex** *administration* application accessible from the Windows desktop. It accesses **Hopex Application Server** (HAS) via the URL of the Web server.

☛ *For detailed information regarding HAS, see "HAS Installation guide" Installation and Deployment Technical Article.*

To access the **Hopex Administration** (Windows Front-End) application:

1. Access the HAS instance repository.
   Default path: C:\ProgramData\MEGA\Hopex Application Server\ <name of the HAS instance on which is created the environment>

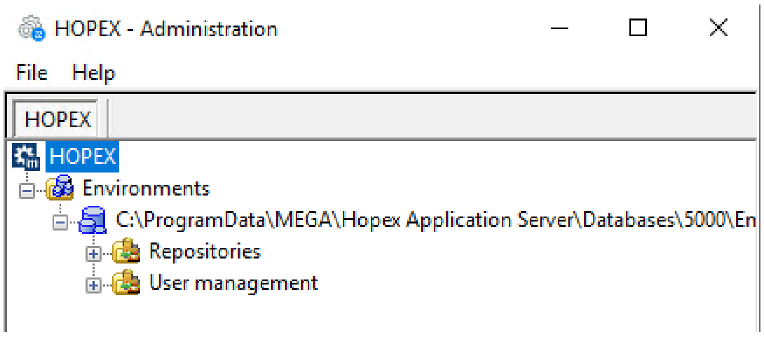
   `E.g.: C:\ProgramData\MEGA\Hopex Application Server\5000`

2. Double-click the "Administration.exe" file .

   ☛ *The administration icon is created during setup of an administrator workstation.*

   The **Hopex - Administration** main window opens.

   ☛ *An environment preceded by a red icon is no longer accessible.*



Hopex Administration shows as a tree the elements you can manage:

- *site*
- *environment*
- *repositories*

☛ *users are managed in the Administration desktop only.*
See Managing users in the Web Administration Guide.

CONNECTING TO AN ENVIRONMENT

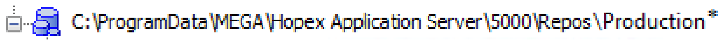
To connect to an environment:

1. Access **Hopex - Administration**.

   ☛ *See Accessing Hopex Administration.*

2. Expand the **Environments** folder.

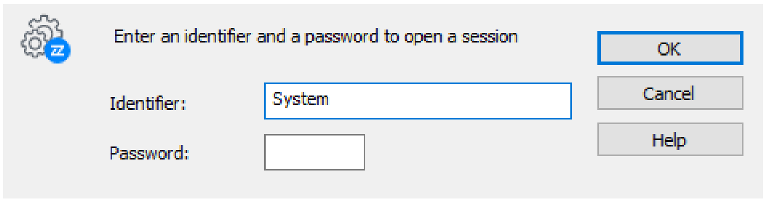
   ☛ *The asterisk that may appear after the environment name means that you must compile the metamodel and/or the technical data, see Compiling an Environment.*



C:\ProgramData\MEGA\Hopex Application Server\5000\Repos\Production*

3. Right-click the environment you want to connect to and select **Open**. The environment connection dialog box appears.
4. In the **Identifier** field, enter the identifier of an *Administrator*.

   ☛ *Administrator and Mega users own administration rights. The Administrator user identifier is System. The Mega user identifier is Mega.*



Enter an identifier and a password to open a session

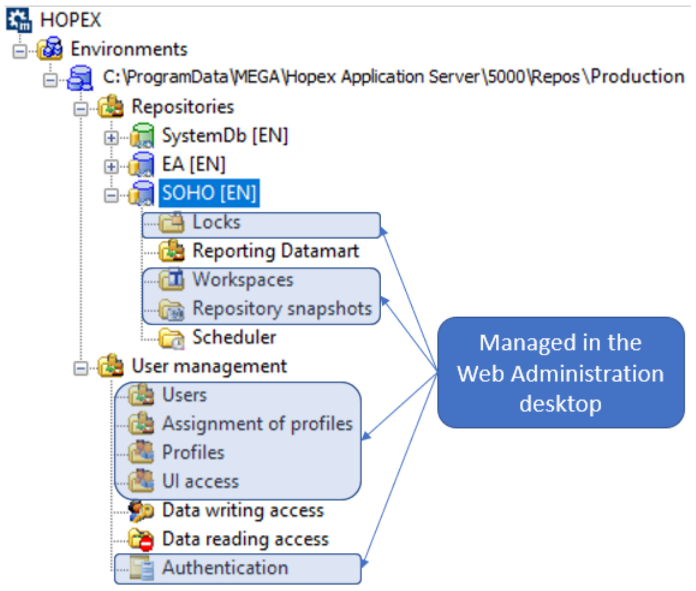
Identifier: System

Password:

5. Enter the user **Password**.

   ☛ *By default, "Hopex" is the password for Administrator and Mega users.*

6. Click **OK**.
   The content of the environment folder is available.



This environment folder contains the folders:

- **Repositories** containing the SystemDb repository as well as **Hopex** repositories referenced in the environment, to manage:
  - *Reporting Datamarts*
  - its Triggers and Jobs (Scheduler)

  ☛ *private workspaces, locks, repository snapshots are managed in the Administration desktop only.*

  See Managing Workspaces (Web Administration guide) and Repository Snapshots (Common Features guide).

- **User Management** to manage:
  - the *Data Writing Access areas*
  - the *Data Reading Access areas*

  ☛ *users, Assignment of profiles, profiles, and UI Access and authentication (authentication groups and parameters) are managed in Administration desktop only.*

  See Managing users and Managing UI Access (Permissions) in the Web Administration Guide.

REPOSITORIES

This chapter covers points relating to working in a repository and the use of repositories:

Introduction to Repositories

An environment includes:

- a system repository
- one or several **Hopex** repositories

Repository management is carried out in the **Hopex Administration** application (not available in the Web Administration Desktop).

The following points are covered here:

System Repository (SystemDb)

The system repository (SystemDb) contains the configuration required to run **Hopex**:

- the metamodel, constituting the structure of repositories
- programmed *queries* and macros
- the elements to produce outputs: report templates, *report templates (MS Word)*, Web site templates
- users and their rights.

A system repository exists for each environment, automatically created in the **Sysdb** folder at creation of the environment.

Hopex Repository

A **Hopex** repository constitutes the workspace in which modeling data is stored. Several users can connect to it and work simultaneously on the same project.

A repository depends on an *environment*. Different policies of data distribution can be implemented. You can, for example, work on two repositories:

- a Development repository, which groups all projects
- a Production repository, which groups stable states of each project.

Any user of an environment can access a single repository or all of the repositories of the environment according to his/her profile assignment definition.

☛ *See Assigning a Profile to a Person.*

Storage type

The **Hopex** repository is stored in the RDBMS format (Relational Database Management System): SQL Server.

> ☛ *For more details on structure of RDBMS storage, see **RDBMS Repository Installation** deployment guide.*

Repository Structure

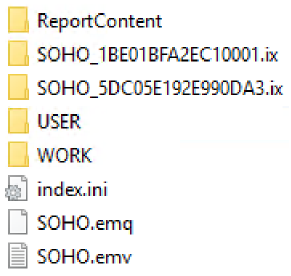
The files enabling use of a repository are all in the same folder, the location of which is stored in the *system repository* (SystDb).

> 🕮 *SystDb is a particular repository containing the metamodel and technical data (descriptors, Web site templates, queries, etc.). The metamodel and technical data are common to all repositories in the same environment. Definition of users and their rights are stored in this repository, essential for operation of the software.*

A repository comprises <RepositoryName>.XXX files of which XXX format depends on the information type:

- ***.EMV*** : repository evolution logfile (e.g.: creation, update)
- ***.EMQ*** : pointer to data stored in the SQL Server

For each repository, folders are created dynamically while the user is working:


ReportContent
SOHO_1BE01BFA2EC10001.ix
SOHO_5DC05E192E990DA3.ix
USER
WORK
index.ini
SOHO.emq
SOHO.emv

- ***ReportContent***: internal use only, for report cache
- ***<RepositoryName>.ix***: contains the result of indexing for full-text search.

  > ☛ *See Enabling/Disabling repository indexing for full-text search.*

- ***USER***: contains a folder for each user, in which all work files generated by the user are stored. It groups the files created by backup and extraction procedures, as well as control files.

  > ☛ *Each folder is named "CCC", where CCC is the code associated with the user.*

- ***WORK***: contains work files created by administration operations carried out using the administration application.

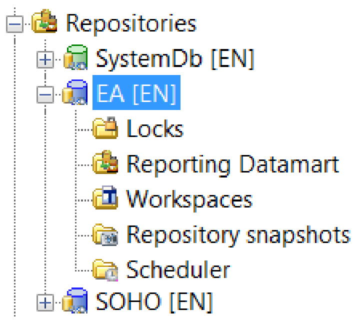
Accessing Repositories

To access a repository:

1. From **Hopex Administration**, connect to the environment.

   ☛ *See Connecting to an Environment.*

2. Expand the **Repositories** folder.
   The list of the repositories in the environment appears.



Each repository is represented by:

- an icon
  - system repository
  - data repositories
- its installation language.

Creating a Repository

You can create:

- an empty repository, then you can transfer data from another repository into it by importing updates or by restoring a backup file.

  ☛ *See **Installation and Deployment > RDBMS Repository Installation Guide**.*

- a repository already fed with data.

  ☛ *See **Installation and Deployment > RDBMS Repository Installation Guide > Hopex RDBMS repositories specific administration actions > Restoring a Hopex environment from formatted data > Restoring a data repository**.*

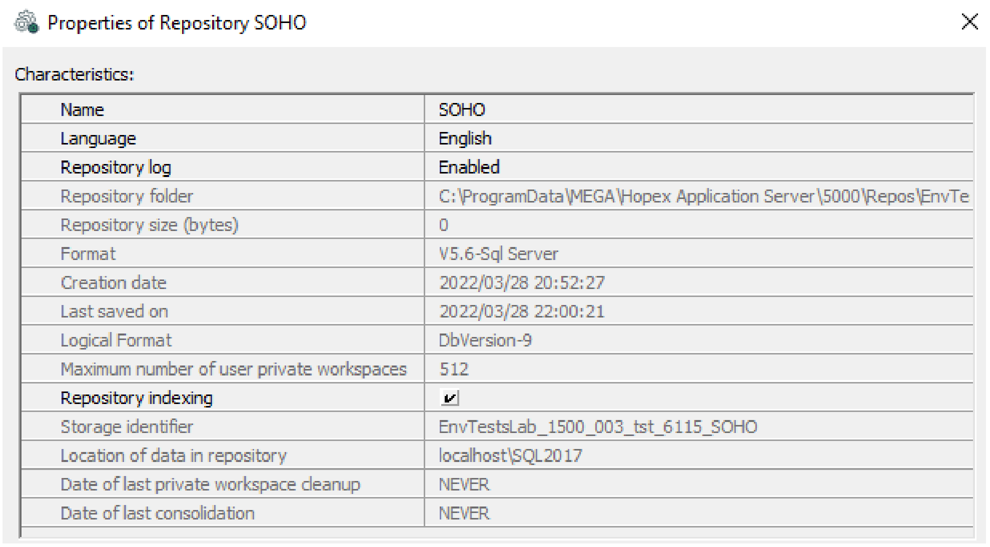
Consulting and Modifying Repository Properties

To consult and/or modify certain repository properties:

1. Connect to **Hopex Administration** and select the repository concerned.

   ☛ *See Accessing Repositories.*

2. Right-click the concerned repository and select **Properties**.
   The **Repository Properties** dialog box opens.



Properties of Repository SOHO

Characteristics:

Name	SOHO
Language	English
Repository log	Enabled
Repository folder	C:\ProgramData\MEGA\Hopex Application Server\5000\Repos\EnvTe
Repository size (bytes)	0
Format	V5.6-Sql Server
Creation date	2022/03/28 20:52:27
Last saved on	2022/03/28 22:00:21
Logical Format	DbVersion-9
Maximum number of user private workspaces	512
Repository indexing	✓
Storage identifier	EnvTestsLab_1500_003_tst_6115_SOHO
Location of data in repository	localhost\SQL2017
Date of last private workspace cleanup	NEVER
Date of last consolidation	NEVER

From the **Repository Properties** dialog box, you can:

- click the row of one of the following characteristics to:
  - modify the **Name** of the repository
  - modify the repository **Language**

    Only if you need to modify the repository language right after its creation. Never change the repository language at any time, to avoid irreparable loss of object names.
  - enable/disable the **Repository log**

    ☛ *For more details, see Repository log.*
- consult its following characteristics:
  - **Metamodel last compiled date** (SystemDb repository specific)
  - location of the **Folder** in which the repository is stored
  - **Dates** of repository creation and last save
  - repository **Logic Format**.
  - **Date of last private workspace cleanup** and **Date of last consolidation**, which provide useful information for the maintenance plan of an RDBMS repository.

    ☛ *See Following the Deletion of Repository Temporary Data and Historical Data.*
- enable/disable **Repository Indexing**.

  ☛ *See Managing Repository Indexing.*

Accessing the Log of Repository Changes (.EMV file)

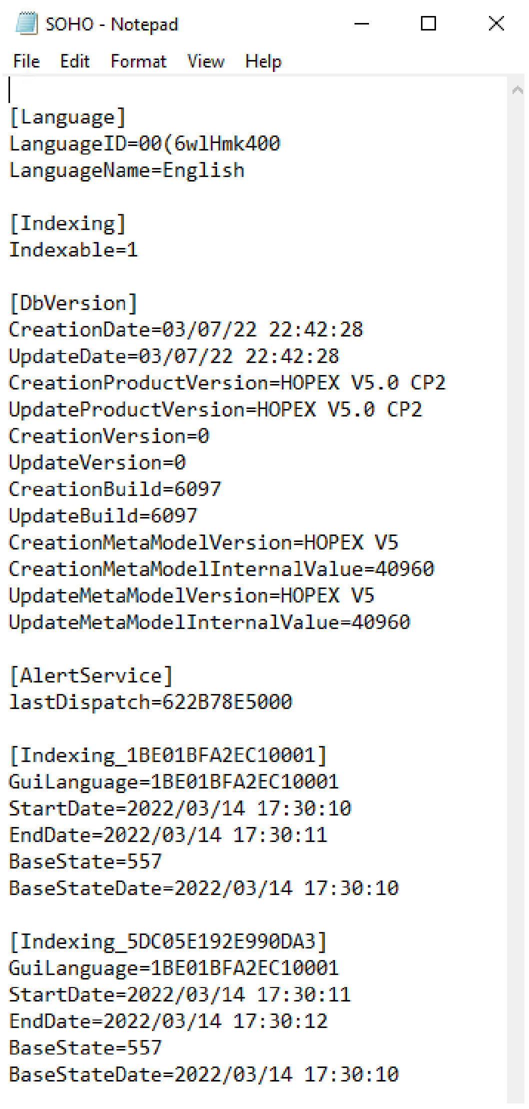
The .EMV file contains repository changes (eg: creation, update, version).

To directly access the .emv file of a repository:

1. Connect to **Hopex Administration** and select the repository concerned.

   ☛ *See Accessing Repositories.*

2. Right-click the repository concerned and select **See EMV File**.



```
[Language]
LanguageID=00(6wlHmk400
LanguageName=English

[Indexing]
Indexable=1

[DbVersion]
CreationDate=03/07/22 22:42:28
UpdateDate=03/07/22 22:42:28
CreationProductVersion=HOPEX V5.0 CP2
UpdateProductVersion=HOPEX V5.0 CP2
CreationVersion=0
UpdateVersion=0
CreationBuild=6097
UpdateBuild=6097
CreationMetaModelVersion=HOPEX V5
CreationMetaModelInternalValue=40960
UpdateMetaModelVersion=HOPEX V5
UpdateMetaModelInternalValue=40960

[AlertService]
lastDispatch=622B78E5000

[Indexing_1BE01BFA2EC10001]
GuiLanguage=1BE01BFA2EC10001
StartDate=2022/03/14 17:30:10
EndDate=2022/03/14 17:30:11
BaseState=557
BaseStateDate=2022/03/14 17:30:10

[Indexing_5DC05E192E990DA3]
GuiLanguage=1BE01BFA2EC10001
StartDate=2022/03/14 17:30:11
EndDate=2022/03/14 17:30:12
BaseState=557
BaseStateDate=2022/03/14 17:30:10
```

Repository Performance and Health

See:

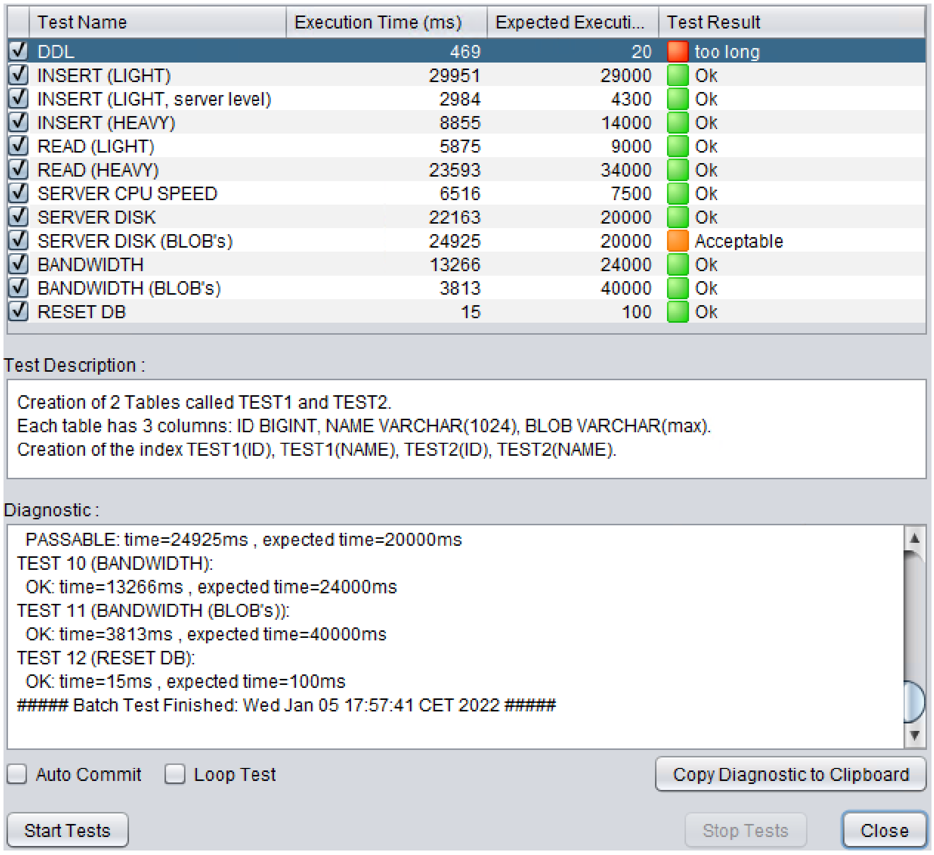
- Consulting Repository Performance
- Generating a Repository Health Report

Consulting Repository Performance

Before you start working in an RDBMS repository, **Bizzdesign** recommends that you run the **RDBMS diagnostic** utility (available in the **Bizzdesign store**: store.mega.com).

☛ *For more details on this utility, see the deployment guide:* ***Installation and Deployment > RDBMS Repository Installation Guide****.*

This utility indicates repository performance compared to optimized performance.



	Test Name	Execution Time (ms)	Expected Executi...	Test Result
✓	DDL	469	20	🟥 too long
✓	INSERT (LIGHT)	29951	29000	🟩 Ok
✓	INSERT (LIGHT, server level)	2984	4300	🟩 Ok
✓	INSERT (HEAVY)	8855	14000	🟩 Ok
✓	READ (LIGHT)	5875	9000	🟩 Ok
✓	READ (HEAVY)	23593	34000	🟩 Ok
✓	SERVER CPU SPEED	6516	7500	🟩 Ok
✓	SERVER DISK	22163	20000	🟩 Ok
✓	SERVER DISK (BLOB's)	24925	20000	🟧 Acceptable
✓	BANDWIDTH	13266	24000	🟩 Ok
✓	BANDWIDTH (BLOB's)	3813	40000	🟩 Ok
✓	RESET DB	15	100	🟩 Ok

Test Description :

Creation of 2 Tables called TEST1 and TEST2.
Each table has 3 columns: ID BIGINT, NAME VARCHAR(1024), BLOB VARCHAR(max).
Creation of the index TEST1(ID), TEST1(NAME), TEST2(ID), TEST2(NAME).

Diagnostic :

```
 PASSABLE: time=24925ms , expected time=20000ms
TEST 10 (BANDWIDTH):
 OK: time=13266ms , expected time=24000ms
TEST 11 (BANDWIDTH (BLOB's)):
 OK: time=3813ms , expected time=40000ms
TEST 12 (RESET DB):
 OK: time=15ms , expected time=100ms
###### Batch Test Finished: Wed Jan 05 17:57:41 CET 2022 ######
```

Generating a Repository Health Report

With **Hopex** you can generate a daily RDBMS repository health report. This report enables to detect:

- performance or usage anomalies that users can face daily.
- any significant change.

For this purpose, performance and health tests are run daily. Events are generated when anomalies are detected.

☛ *See **Event Supervision Description > Repository Health** Technical Article.*

Performance test description

Hopex standard use scenarios are performed every afternoon ("RepositoryHeath Daily Afternoon Trigger" job, 04:00 pm GMT):

- Reading of comments on read-only data ("Reading of").
- Read-only data loading.
- Request execution on read-only data.
- Comment writing on reading and writing data.
- Data creation.
- Data deletion.
- Search request execution on reading and writing data.

☛ *In a cluster-type configuration, performances are measured on all of the machines.*

Each scenario generates a result, which is stored in the repository. These results are analyzed daily in the evening ("RepositoryHeath Daily Evening Post Trigger" job, 11:05 pm GMT)

An history of 36 results are needed before generating an alert.

Health test description

It is essential to analyze certain usages to identify anything that might compromise data integrity, whether in the daily work or following a **Hopex** update.

For all of the repositories of all of the environments, the following checks are performed every evening ("RepositoryHeath Daily Evening Trigger" job, 11:00 pm):

- Administration
  - SQL compatibility of repositories and server
  - table fragmentation
  - index fragmentation
  - SQL maintenance plan execution
- Customization
  - Hopex data modification
  - Hopex data volume
- Usage
  - workspace volume

☛ *In a cluster-type configuration, usage tests are performed randomly on a single machine only.*

Health report description

The health report includes a short description of the anomaly detected at performance or usage level.

```
For example, in case of an index fragmentation anomaly:

Sent from: <Name of the machine that ran the report
formatting and emailing task>

Environment: <Environment name>

Repository: <Repository name>

Table A_BLOB

Index GBM_INDEX_A_BLOB_IDABS_BEGIN_VALIDITY

Fragmentation level 70%
```

Configuring Hopex health report emailing

☛ *Hopex health reports, automatically generated every day, are available from Hopex Administration desktop, see Viewing the Hopex Health Reports.*

You can receive a short report of all of the anomalies detected on each repository of each environment.

The Hopex health report is emailed daily (by default):

- You must check that the mailing configuration parameters are configured (in the site level options).
- You must define the list of recipients of the report.
- You can modify the report receiving frequency.

```
E.g.: Daily, Weekly (Sunday), Daily (working days), Monthly
(1st day of the month).
```

To configure Hopex for emailing its health report:

1. Access the site level options.

   ☛ *See Accessing Options.*

2. Check the mailing configuration parameters:
   - In the Options tree, select **Installation > Electronic Mail** folder.
   - In the right pane, check that the mailing configuration parameters are specified.

     Example:

     Default address of author via SMTP (FROM): admin@domain.com

     SMTP Server: mail.server.domain.com

     SMTP Port: 25

     ☛ *See Configuring SMTP Settings.*
3. Define the recipients of the report:
   - In the Options tree, select **Repository > Repository Health** folder.
   - In the field of the **Repository Health Report: recipients** option, enter the email of the recipient of the Hopex health reports.

     ☛ *You can enter several recipients (the separating character is the coma: ",").*

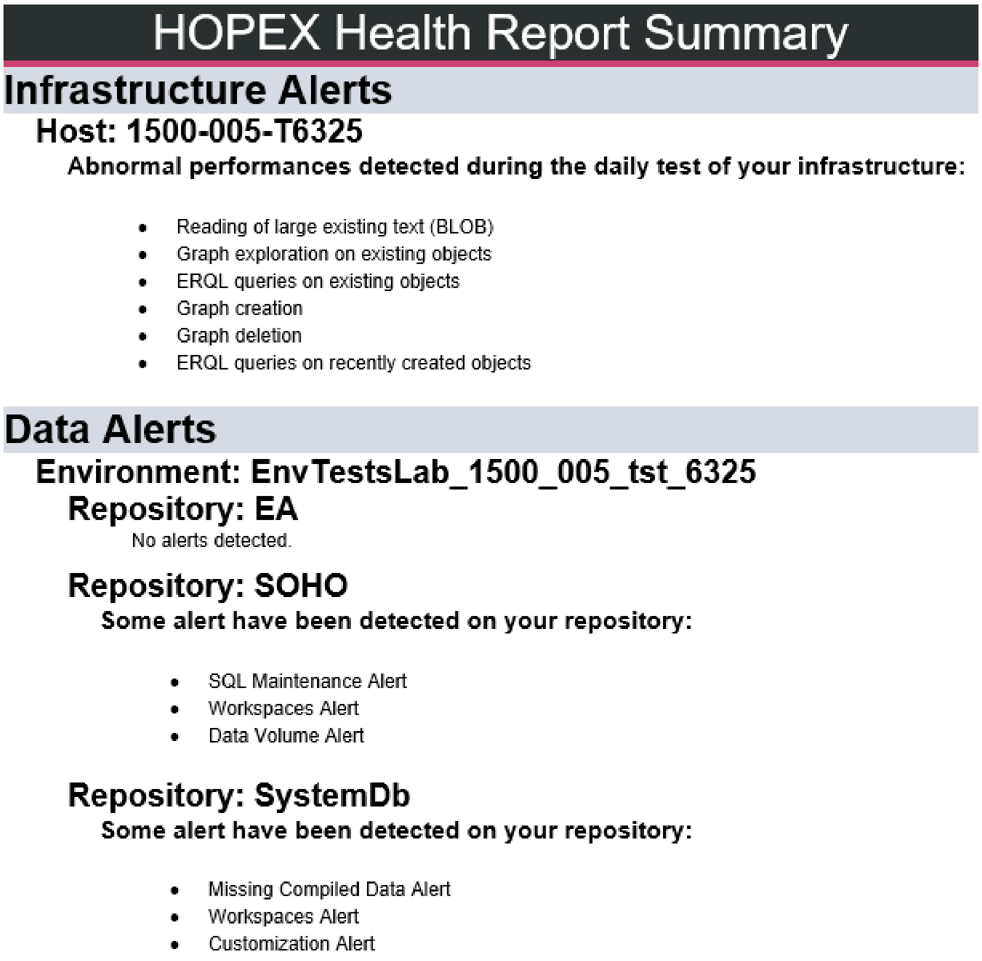
     E.g. : dan.woods@bizzdesign.com,julia.perri@bizzdesign.com

4. (If needed) For the **Repository Health Report: frequency** option, use the drop-down menu to modify the report receiving frequency.

   ```
   E.g.: Daily (working days)
   ```

   The recipients of the report receive an email at the frequency defined, showing a summary and including the attached detailed report.

HopexHealthFullReport2023-01-30_14-24-09.html
63 KB


HOPEX Health Report Summary

Infrastructure Alerts

Host: 1500-005-T6325

Abnormal performances detected during the daily test of your infrastructure:

- Reading of large existing text (BLOB)
- Graph exploration on existing objects
- ERQL queries on existing objects
- Graph creation
- Graph deletion
- ERQL queries on recently created objects

Data Alerts

Environment: EnvTestsLab_1500_005_tst_6325

Repository: EA

No alerts detected.

Repository: SOHO

Some alert have been detected on your repository:

- SQL Maintenance Alert
- Workspaces Alert
- Data Volume Alert

Repository: SystemDb

Some alert have been detected on your repository:

- Missing Compiled Data Alert
- Workspaces Alert
- Customization Alert

MANAGING REPOSITORIES

☛ *For management operations specific to an RDBMS repository, see deployment documentation: **RDBMS Repository Installation Guide**.*

The following points are covered here:

Repository log

At repository creation, the repository log is **Enabled** by default.

The repository log lists all modifications made in the repository. It gives users a better understanding of actions dispatched in the repository from private workspaces.

Each time an action is executed, an occurrence of Change Item is created.

*A **ChangeItem** is a MetaClass corresponding to a change made in a **Hopex** repository.*

A repository log comprises **Hopex** occurrences. These occurrences can be handled using **Bizzdesign** APIs.

☛ *See Viewing the Repository Update Log.*

To keep a history of the actions performed on the repository after dispatch, the repository log must be kept "Enabled".

Configuring the logging

To improve performance you can define some MetaClasses or MetaAssociationEnds as non loggable.

☛ *See Modifying MetaClass loggability.*

Logging of updates enables:

- mainly to view the repository activity, i.e. actions performed on objects.

  ☛ *See Viewing the Repository Update Log.*

  In that case, a truncated log (**Delete** command) is functionally satisfactory.

  ☛ *See Deleting a log or reducing the log size step 4.*

  This is the default configuration.
- to transfer the commands performed from a repository to another.

  ```
  Example: you want to transfer all the commands performed
  during the day from a development repository to a production
  repository.
  ```

  In that case a complete log, including all the actions performed by the users, is necessary so that the inter repository *consolidation* is performed correctly.

  Consolidation groups the updates from stand-alone workstations or remote sites (with Lan) and merges them in a reference site. After dispatch of the private workspaces of each of the users, the repository log is exported and reinitialized. The logfiles are imported into the reference repository, then this is recopied on each of the user sites

Viewing the Repository Update Log

You can view the history of the actions performed in the repository after dispatch in:

- the **Hopex Administration** application (Windows Front-End)

  ☛ *See Viewing the repository update log.*
- **Hopex** (**Repository Activity** navigation window)

  ☛ *See Displaying dispatches.*

  ☛ *See Handling Updates.*
- the **Administration** desktop (Web Front-End)

  ☛ *See the **Hopex Administration - Supervisor Web** guide, section "Displaying Updates Made in the Repository".*
- the object **History**

Viewing the repository update log

To view the repository update log:

1. Connect to **Hopex Administration** and select the repository concerned.

   ☛ *See Accessing Repositories.*

2. Right-click the repository concerned and select **Repository Log > Open**.
   The **View repository updates log** appears.

   ☛ *From **Hopex**, select the **File > Properties** menu. In the dialog box that appears, select the **Update** tab. The log of the current private workspace is displayed by default.*

3. Select (/Clear) **System Repository** to display the system repository (/ **Hopex** repository) updates.
4. In the **Period** field, select the period you are interested in.

   ```
   E.g.: Today, Current week, Current month, From the
   beginning.
   ```

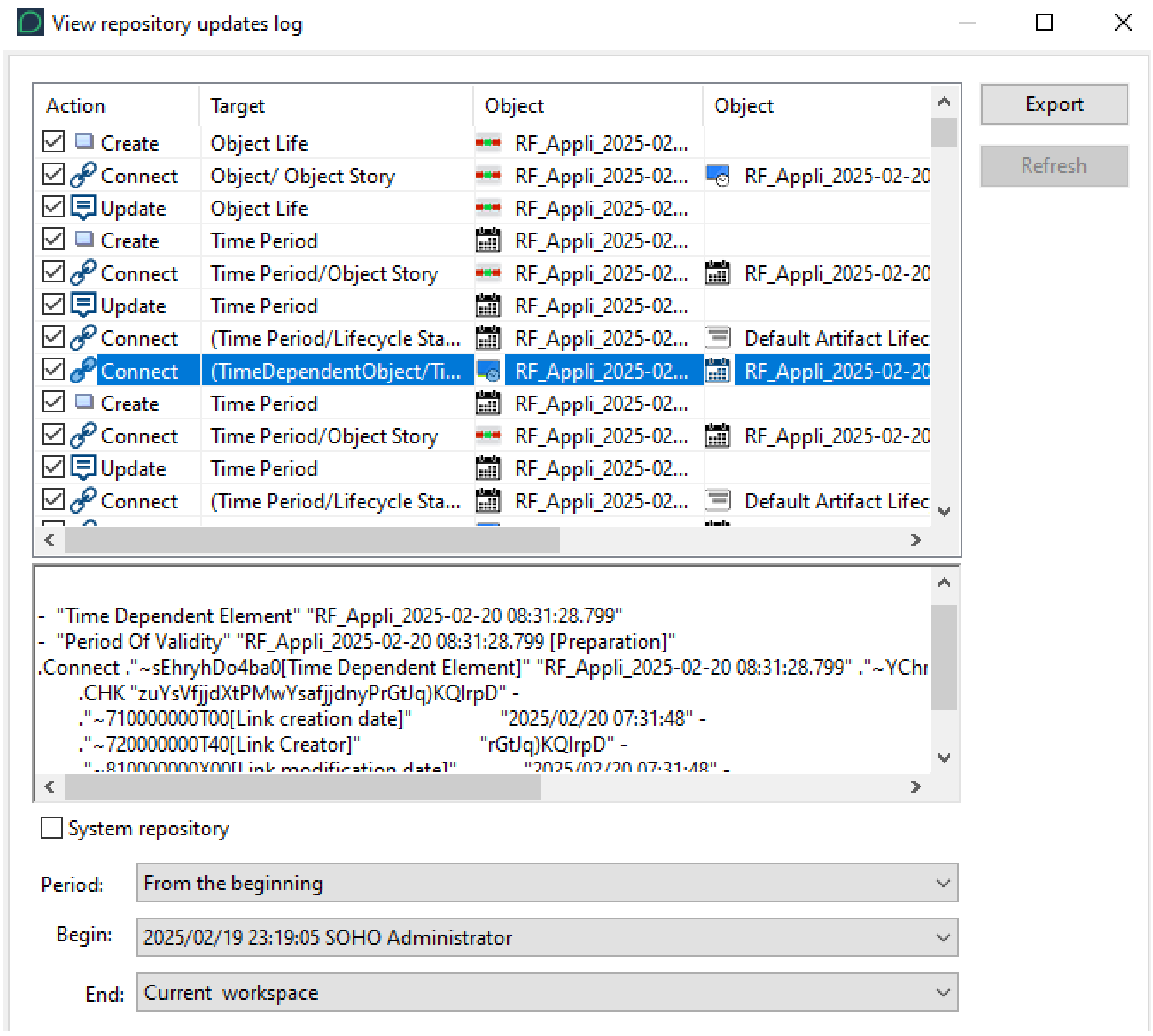
   The selected period defines the list of dispatches available in the **Begin** drop-down list.
5. In the **Begin** field, select the dispatch from which you want to display the updates.

   ☛ *The first item corresponds to the first dispatch in the repository.*

6. In the **End** field, select the dispatch until which you want to display the updates.

   ☛ *You can view the whole repository log, or between two given intermediate dispatches.*

7. Click **Refresh**.
   The repository log appears as a list of actions displayed in chronological order.



View repository updates log

Action	Target	Object	Object
✓ Create	Object Life	RF_Appli_2025-02...	
✓ Connect	Object/ Object Story	RF_Appli_2025-02...	RF_Appli_2025-02-20
✓ Update	Object Life	RF_Appli_2025-02...	
✓ Create	Time Period	RF_Appli_2025-02...	
✓ Connect	Time Period/Object Story	RF_Appli_2025-02...	RF_Appli_2025-02-20
✓ Update	Time Period	RF_Appli_2025-02...	
✓ Connect	(Time Period/Lifecycle Sta...	RF_Appli_2025-02...	Default Artifact Lifec
✓ Connect	(TimeDependentObject/Ti...	RF_Appli_2025-02...	RF_Appli_2025-02-20
✓ Create	Time Period	RF_Appli_2025-02...	
✓ Connect	Time Period/Object Story	RF_Appli_2025-02...	RF_Appli_2025-02-20
✓ Update	Time Period	RF_Appli_2025-02...	
✓ Connect	(Time Period/Lifecycle Sta...	RF_Appli_2025-02...	Default Artifact Lifec

```
-  "Time Dependent Element" "RF_Appli_2025-02-20 08:31:28.799"
-  "Period Of Validity" "RF_Appli_2025-02-20 08:31:28.799 [Preparation]"
.Connect ."~sEhryhDo4ba0[Time Dependent Element]" "RF_Appli_2025-02-20 08:31:28.799" ."~YChr
       .CHK "zuYsVfjjdXtPMwYsafjjdnyPrGtJq)KQIrpD" -
       ."~710000000T00[Link creation date]"            "2025/02/20 07:31:48" -
       ."~720000000T40[Link Creator]"              "rGtJq)KQIrpD" -
       ."~810000000X00[Link modification date]"          "2025/02/20 07:31:48" -
```

☐ System repository

Period: From the beginning

Begin: 2025/02/19 23:19:05 SOHO Administrator

End: Current workspace

☛ *See Exporting Updates.*

Indicated for each action are:

- the **Action** type performed

  ```
  Example: "Create", "Connect", "Update".
  ```

  ☛ *See Command File Syntax for more information on operators.*

- the type of the object concerned (**Target**)
- the name of the **Object** concerned
- the name of the second **Object** concerned in the case of a "Connect", "Disconnect" or "Change" action
- the person **Responsible** for the action
- the **Delivery date** of the action

  ```
  Format: <D/MM/YYYY h:mm:ss AM/PM>
  ```

  If you have not dispatched your work yet, the date indicated is the execution date of the action.

- The name of the **Dispatch** that contains the action, in the following format:

  ```
  <YYYY/MM/DD hh:mm:ss> <repository> <responsible>
  ```

  With:

  - <YYYY/MM/DD hh:mm:ss>: workspace date and creation time
  - <Repository>: name of the current repository
  - <responsible>: name of the person responsible for the action

  If you have not dispatched your work yet:

  ```
  <your name> (workspace)
  ```

- (when you select the line of a command), the complete text of the update is displayed in the window lower pane.

  ☛ *If needed, to copy the command and paste it in a text file for example, roll the mouse over the whole text, then press <CTRL> + <C> and paste the text in a text file.*

Displaying dispatches

To display in **Hopex** private workspaces dispatched and the content of their updates:

1. Connect to **Hopex**.

   ☛ *See Connecting to Hopex Customizing Desktop.*

2. In the **Repository Activity** Navigation window, expand the **Repository Dispatch** sub-folder.
   All dispatches performed on the current repository and the system repository are contained in the respective **<Repository Name>** and **SystemDb** folders.
3. Expand the repository folder concerned.
   Dispatches are filed by day, week and month.
4. Expand the dispatch concerned.
   Each dispatch details the updates made to an object.
5. Expand the update concerned.
   In the properties of the object concerned, the **General > History** tab details the content of the dispatch in the form of a list of actions performed on the object concerned.

Exporting Updates

You can export updates between consecutive dispatches or not.

☛ *To export the repository log, see Backing up the repository log.*

To export updates:

1. Access the repository update log.

   ☛ *See Viewing the Repository Update Log.*
2. Select the updates to be exported.
3. Click **Export**.
4. Select the export format:
   - *.mgl: text format
   - *.xmg: MEGA XML format.
5. (If needed) Modify the export file name and save folder proposed as default. The **Browse** button ... allows you to browse the folder tree and select the folder in which the file will be saved.
6. Click **Export**.
   The file is exported and saved in the specified folder.

   The **Execution Report** appears.
7. (Optional) Click **Open result file** to view the file.
8. Click **OK** to close the window.

Managing Repository Indexing

To allow the user to perform full-text searches in **Hopex** solutions, the repository must first be indexed.

Indexing runs automatically every 10 minutes (modifiable default value) with the indexing scheduler. This indexing is incremental and concerns created and modified objects.

Some of Hopex metamodel modifications (for example setting a MetaClass to indexable) require an entire indexing, see Indexing a repository manually.

☛ *To modify the scheduler default configuration, see Customizing the indexing scheduler.*

☛ *The initial indexing can take some time depending on the size of your repository (eg: more than 30 hours for a 2 GB repository, of which business documents constitute a major part), and can slow the performance of **Hopex**. Remember to run this initial indexing when other **Hopex** users are not connected.*

Take care to allow sufficient disk space before enabling indexing: statistically for a large repository (eg: 2 GB) of which business documents constitute a major part, the indexing size can represent twice the repository size.

Enabling/Disabling repository indexing for full-text search

By default, repository indexing is enabled.

To enable/disable repository indexing:

1. Access properties of the repository concerned.

   ☛ *See Consulting and Modifying Repository Properties.*

2. Select/Clear **Repository Indexing** to enable/disable repository indexing.
3. Click **OK**.
   The scheduler updates indexing every 10 minutes.

   ☛ *To modify scheduler default configuration, see Customizing the indexing scheduler.*

   Indexing is carried out for all of the languages installed.

   When indexing is completed, the <Repository Name>.IX folder is created in the corresponding folder of the repository. This folder contains indexing results.

Indexing a repository manually

The administrator can index the repository manually :

- for an initial indexing
- to restart a full indexing
  Some of **Hopex** metamodel modifications may have indexing impacts. This requires to delete old indexing folders for each repository (including SystemDb repository) so that the entire indexing is performed.

```
Example: when a MetaClass becomes indexable (in the
MetaClass properties, its Candidate to indexation attribute
is set to "Yes"), the indexing automatic update does not
take into account the objects corresponding to this
MetaClass.
```

To index a repository manually:

☛ *To be able to manually index the repository, the repository indexing option must be selected, see Enabling/Disabling repository indexing for full-text search.*

1. Connect to **Hopex Administration** and select the repository concerned.

   ☛ *See Accessing Repositories.*

2. Right-click the repository to be indexed and select **Index for full-text search**.
   A window shows the repository indexing state (not indexed, already indexed, indexing running).
3. Click **Start Repository Indexing** ▶
   Indexing is carried out for all of the languages installed.

   ☛ *For initial indexing, the **SearchIndexes** folder is created in the repository folder. This folder contains indexing results: one sub-older by language (sub-folder names: Language hexa IdAbs).*

Customizing the indexing scheduler

You can modify indexing scheduler default configuration.

To customize the indexing scheduler default configuration:

1. Connect to **Hopex Administration** and select the repository concerned.

   ☛ *See Accessing Repositories.*

2. Right-click **Scheduler** and select **Manage Triggers**.
3. Select the **System Triggers** tab.
4. Right-click **Indexing Automaton** and select **Update Scheduling**.
   The indexing scheduler configuration window appears.
5. Modify the configuration.

   ☛ *For information on the scheduler, see Scheduling (Scheduler).*

6. Click **OK**.

Converting a Repository

Repository conversion is only necessary:

- at migration.

  ☛ *For more information on repository conversion, consult the **How to migrate to Hopex** technical article.*

- on request from **Hopex** support.

With **Hopex** you can mass convert technical data. This possibility enables you to mass launch the procedure on all the repositories with no need to validate the execution for each repository.

Mass converting technical data

To mass convert technical data:

1. From **Hopex Administration**, connect to the environment.

   ☛ *See Connecting to an Environment.*

2. Expand the environment folder.
3. Right-click the **Repositories** folder and select **Manage**.
4. In the **Action list** pane, select **Perform SQL conversion on the repository**.
5. In the **Repository list** pane, select the repositories for which you want to convert technical data.
6. Click **Execute**.
   The procedure is automatically launched on each repository selected with no need to validate each execution.

   ☛ *A dialog box enables you to stop the procedure at any time. If needed, click **Cancel** to stop the procedure execution.*

Importing Libraries into a Repository

You can import the libraries you need for your work into a repository.

To import a library into a repository:

1. Connect to **Hopex Administration** and select the repository concerned.

   ☛ *See Accessing Repositories.*

2. Right-click the repository concerned and select **Object Management > Import**.
   The **Import Hopex Data** dialog box opens.
3. Alongside the **Command File** field, click **Browse** ... .
4. From the **Hopex** installation folder, select in the **MEGA_Std** folder the desired Library (*.mol).
5. Click **Open**.
6. In the **Import Hopex data** window, click **Import**.
7. When import is completed, click **Close**.

Repository Physical Backup

In event of a problem, you must have a valid and recoverable data backup.

A physical backup consists of copying the files of a repository from their original location to another one.

Data is stored at **Hopex** environment level. Data to be backed up varies according to repository server type.

☛ *See Creating a Repository.*

The following points are covered here:

- Backup frequencies
- Elements to be backed up
- Other elements to be backed up
- Elements that could be useful to back up

Backup frequencies

For a **Hopex** environment used by an active project, **Hopex** recommends:

- daily backup of the environment
- backup before any major data update

  `Example: system database customization, data reprocessing, CP/RP update of` **`Hopex`** `data.`

- that you keep:
  - daily backups of the last 30 days
  - monthly backups of the last 12 months

Whatever your repository server type, **Bizzdesign** recommends cold backup (no **Bizzdesign** user should be connected).

☛ *In SQL server type mode, hot backup is possible.*

Elements to be backed up

Identify environments that require regular backup.

`Example: design environment.`

From the environment folder, you must back up the complete folders:

- **Db**
- **SysDb**

Db and **SysDb** folders contain an .EMV file and an .EMQ file that points to other folders that you must back up: system and repository databases of SQL server databases.

Elements that can be excluded from the backup file

To save space and time it is not necessary to backup the complete content of the environment folder. You need not back up for example folders that contain:

- user work files
  These files are contained in the **SysDb\USER** and **Db\<RepositoryName>\USER** folders.
- work files linked to the Administration application
  These files are contained in the **SysDb\WORK** and **Db\<RepositoryName>\WORK** folders.

Other elements to be backed up

Bizzdesign recommends that you back up folders concerning:

- licenses:
  the file containing licenses (.Must or .ELF).

(Optional) You can back up folders concerning:

- your customizations (e.g;: installation, authentication, Hopex, java, static Web site)
  in the **HAS instance** directory, the folders included in
  \.shadowFiles\has.custom\<HAS customization module version>
- your generated Web sites
  in the **Hopex** installation directory, the **Intranet** folder

Elements that could be useful to back up

You may need to back up:

- private workspaces in progress
- technical data modifications in progress

To back up private workspaces in progress:

› copy the "RepositoryName.Transactions" folder in the repository folders tree.

☛ *The **USER** and **WORK** folders contain the work documents of users.*

To back up technical data modifications in progress:

- copy the **SysDb** folder and its sub-folders in the tree of the *system repository*.

Reorganizing an RDBMS Repository

You can reorganize a repository when you want to change the storage type.

`E.g.: from an Oracle repository to an SQL Server repository.`

Before reorganizing a repository, you must check that there is no other active or passive private workspace on this repository, see Workspace Administration.

The reorganization process of an RDBMS repository is automated and consists in:

1. A logical backup of the repository, to get the *command file* which contains creation orders of repository objects and their links.
2. A repository initialization.
3. A repository restore by import of the command file in an empty repository.

Reorganizing a repository

☺ *To improve reorganization times and* ***Hopex*** *performance, remember to delete old log elements before repository reorganization.*

When reorganizing a repository, you can choose to:

- delete log (by default)
- keep the entire log or only part of it (keep the most recent log elements)

Depending on its size, the log import might be very long (several days)

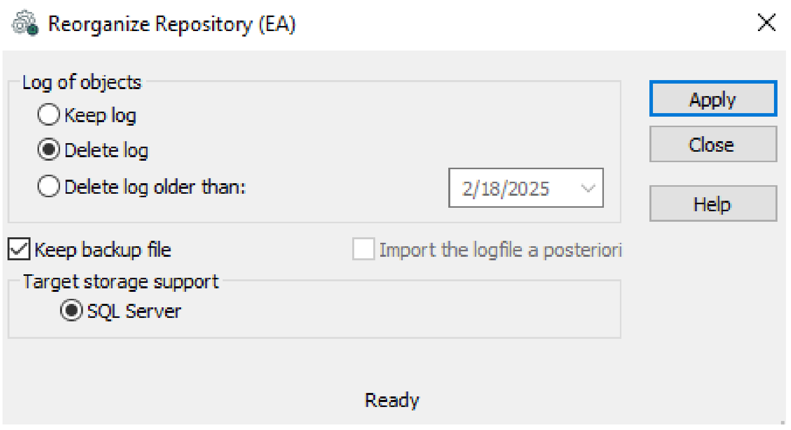
You can select **Import the logfile** a posteriori. Thus, the log is not automatically imported at reorganizing. You must import it manually at a suitable date.

To reorganize a repository:

1. Connect to **Hopex Administration** and select the repository concerned.

   ☛ *See Accessing Repositories.*

2. Right-click the repository concerned and select **Reorganize**.


Reorganize Repository (EA)

Log of objects
- ○ Keep log
- ● Delete log
- ○ Delete log older than: 2/18/2025

☑ Keep backup file   ☐ Import the logfile a posteriori

Target storage support
- ● SQL Server

Ready

3. (Optional) In the **Log of objects** frame, you can select the actions to be executed on the log at reorganization.
4. (Optional, if you chose to keep the log or part of it) Select **Import the logfile** a posteriori to perform the import later.

   ☛ *The logfile is saved in the work folder WORK of the repository, in format: Bkp_<YYYY-MM-dd_HH.mm.ss>_<repository name>-logs.mgr. To import it see Importing command files.*
5. (Optional) If you do not want to keep the repository log, clear **Keep backup file**.

   ☛ *The backup file is generated in the work folder WORK of the repository, in format: Bkp_<YYYY-MM-dd_HH.mm.ss>_<repository name>.mgr*
6. Click **Apply**.
   When the repository has been reorganized, a message indicates that the reorganization is completed.

Repository Logical Backup

Before you make a repository backup, dispatch all current private workspaces so that their changes are included in the backup. To view private workspaces active on your repository, use the private workspaces administration tool, see Workspace Administration.

Logical *backup*:

- creates a *command file* that allows you to reconstruct the repository by update of an empty repository.
- analyzes all repository content.
- is safer than a physical backup since it checks that all data in the repository is readable.
- can be used as a long term archive or to merge repositories.

You can execute updates on the repository after starting logical backup by dispatching a private workspace. However, note that these modifications are not included in the backup.

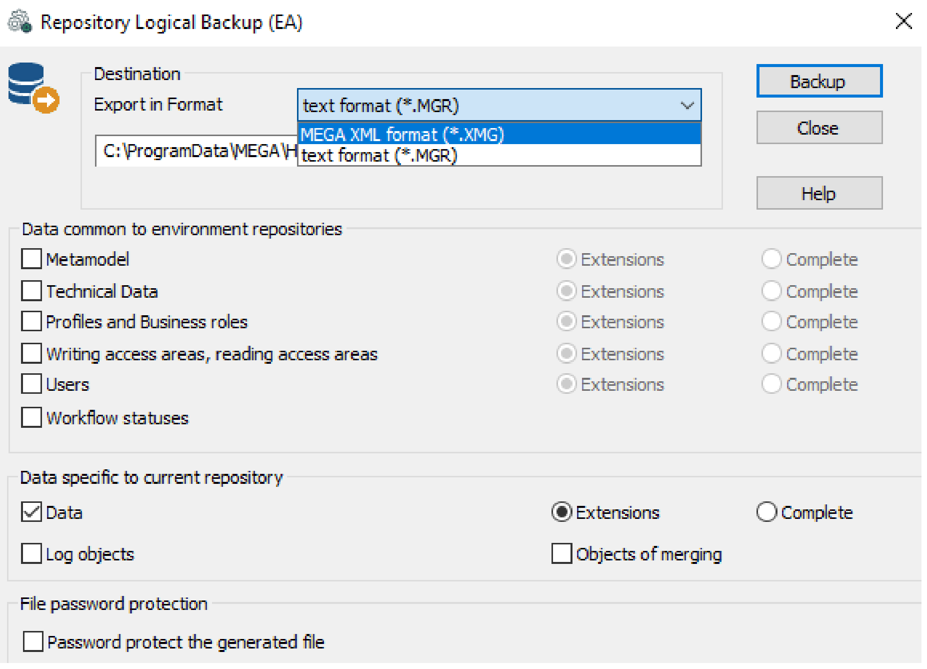
☺ *You can temporarily prohibit users from updating the repository by clearing the* ***Authorize Dispatch for the environment*** *option or* ***Authorize user dispatch*** *option in the environment options.*

To perform a logical backup of a repository:

1. Connect to **Hopex Administration** and select the repository concerned.

   ☛ *See Accessing Repositories.*

2. Right-click the repository concerned and select **Logical Backup**. The **Repository Logical Backup** dialog box opens.


Repository Logical Backup (EA)

Destination
Export in Format: text format (*.MGR)
MEGA XML format (*.XMG)
text format (*.MGR)
C:\ProgramData\MEGA\H

Data common to environment repositories
- [ ] Metamodel — [✓ Extensions] Complete
- [ ] Technical Data — [✓ Extensions] Complete
- [ ] Profiles and Business roles — [✓ Extensions] Complete
- [ ] Writing access areas, reading access areas — [✓ Extensions] Complete
- [ ] Users — [✓ Extensions] Complete
- [ ] Workflow statuses

Data specific to current repository
- [x] Data — [✓ Extensions] Complete
- [ ] Log objects — [ ] Objects of merging

File password protection
- [ ] Password protect the generated file

3. In the **Destination** pane, select the export format of the backup file:
   - **text format** (*.MGR).

     ☺ *For more details on .MGR file syntax, see Command File Syntax.*

   - **MEGA XML format** (*.XMG)
     This format is reserved for exchange of data between **Hopex** and other applications. It includes commands or data (objects and links).

This format cannot be used to extract the metamodel or technical data.

☺ *For more details on MEGA XML data exchange format, see technical article **MEGA Data Exchange XML Format EN**.*

☛ *Environment options enabling configuration of **Hopex** data export (XMG encoding, default export format, etc.). See Options.*

4. (Optional) In the **Destination** pane, click **Browse** ... to browse the folder tree and modify the name and/or location of the backup file.
   By default, the backup creates a file "RepositoryName.mgr" in the WORK *work folder*.

   ☛ *We recommend that this backup be made to a physical device other than the device on which the repository is located. Selection of a different logical device, such as a different partition on the same disk, does not protect your backup in the event of disk failure.*

5. Select the type of data you want to save.

   ☛ *The **Extensions** buttons correspond to data created by users.*

   Carry out a complete backup only if technical support asks you to do so.

   ☺ *You can choose what you want to save: extensions common to environment repositories and data specific to current repository. It is* *recommended that you save all these elements in separate specific files, with names indicating their contents.*

☛ *For a standard logical backup, simply select the* ***Data*** *and its* ***Extensions****.*

In the **Data common to environment repositories** pane, select the data type of the system repository to be saved:

- **Metamodel** allows extraction of the *metamodel*, if the standard metamodel has been modified.
- **Technical Data** allows extraction of data such as *descriptors*, *queries*, and *report templates (MS Word)*.

  A complete logical backup of the repository including the Technical Data can take time and occupies considerable space.

- **Profiles and Business roles** allows extraction of created profiles and business roles (those not provided by Bizzdesign).
- **Writing access areas and reading access areas** allows extraction of created writing and reading access areas (those not provided by Bizzdesign).
- **Users** allows extraction of created users (persons, person groups, logins).
- **Workflow statuses** enables extraction of workflows (workflow instances, transitions and statuses, tasks, validations, requests for change).

In the **Data specific to current repository** pane:

- **Data** allows extraction of repository data. This data includes assignment of business roles to persons.
- **Log objects** allows you to include object histories in the extraction of **Hopex** data.

  ☛ *For more information on object logs, see Viewing Object History.*

- **Objects of merging** allows you to export technical objects resulting from merging objects (_TransferredObject).

  ☛ *For further information on merging objects, see Merging Two Objects.*

6. (If needed) In the **File password protection** pane, select **Password protect the generated file**.

   ☛ *The password is asked at file import.*

7. Click **Backup** to start backup.
   During execution of backup, a series of messages keeps you informed of progress.

   The backup report is displayed in the **Report** area.

   ☛ *For more information on this file, see Viewing the Environment Report File.*

To update a repository, import the repository backup file.

☛ *For more details on importing a command file, see Updating a Repository.*

Deleting a Repository

To delete an SQL Server repository, you need the appropriate administration rights: at least db_creator.

To delete a repository:

1. Connect to **Hopex Administration** and unreference the repository concerned.

   ☛ *See Unreferencing a Repository.*

2. Ask the database administrator to delete the database concerned from the SQL server.

Updating a Repository

Importing command files

You can update a repository by importing the command file produced by the repository backup tool, export of an object or any other means of command file production.

You can import two types of command file into a **Hopex** repository:

- **text format** (.MGR).

  ☛ *For more details on .MG* file syntax, see Command File Syntax.*

- **MEGA XML format** files. These files have .XMG extension and contain commands or data (objects and links).

  ☛ *For more details on MEGA XML data exchange format, see technical article **MEGA Data Exchange XML Format EN**.*

To import a command file:

1. Connect to **Hopex Administration** and select the repository concerned.

   ☛ *See Accessing Repositories.*

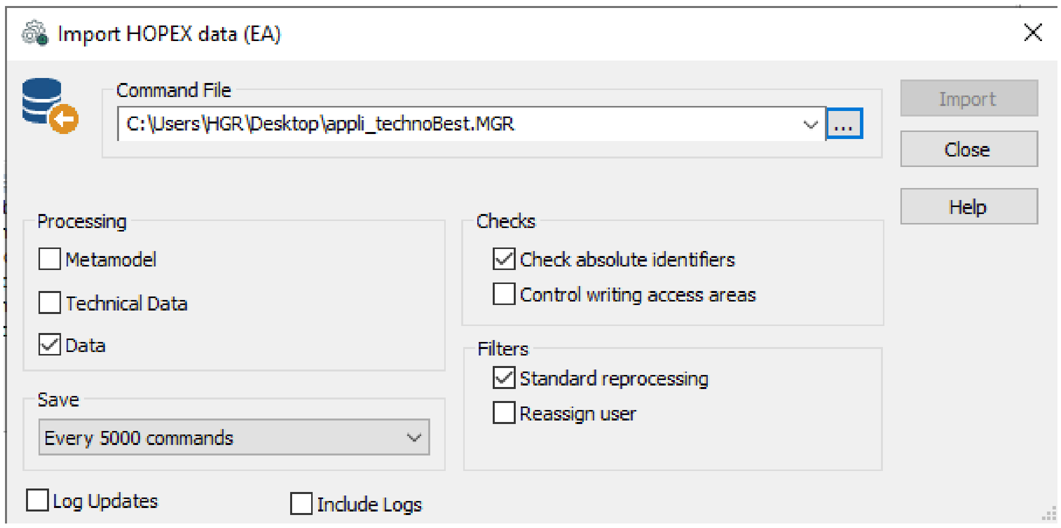
2. Right-click the repository concerned and select **Object Management > Import**.
   The **Import Hopex Data** dialog box opens.

   ☛ *To import a command file from **Hopex**, select **File > Import > Hopex File**.*

3. In the **Command File** pane, click **Browse** ... to browse the folders and select the backup file.

4. Select the types of **Processing** to be executed:
   You can update:
   - the **Metamodel** (repository structure)
   - the **Data** (most frequent case)
   - the **Technical Data** (*descriptions*, *requests*, as well as *users*).

     ☛ *If the file includes commands that do not match the type you have selected, these commands are ignored.*



Import HOPEX data (EA)

Command File: C:\Users\HGR\Desktop\appli_technoBest.MGR

Processing:
- [ ] Metamodel
- [ ] Technical Data
- [x] Data

Save: Every 5000 commands

Checks:
- [x] Check absolute identifiers
- [ ] Control writing access areas

Filters:
- [x] Standard reprocessing
- [ ] Reassign user

- [ ] Log Updates
- [ ] Include Logs

5. Select the **Save** frequency of the modifications.

   ☛ *Note that there is no optimal save frequency:*
   - ***Standard*** *frequency saves at each "Validate" command in the command file and at the end of the file. This type of frequency is useful when the command file has been written by a user.*
   - ***At end*** *is generally sufficient if the file is not very large.*
   - ***At end if no reject encountered*** *saves the changes only if no rejects were encountered.*
   - ***Never*** *is used to carry out tests before the effective update, for example for syntax checking.*
   - ***Every 5000 commands****: each save is quite long. You can speed things up or slow them down by saving* ***every 100, 200, 500, 1000*** *or* ***5000 commands****.*

   ☛ *Large files may cause memory problems when updating. To avoid such problems, you should decrease the intervals between saves.*

6. In the **Checks** pane, the checks to be carried out are selected automatically, based on the file extension:
   - **Check Absolute Identifiers** is not selected in the case of a command file that does not come from a **Hopex** repository.
   - **Control writing access areas** is selected when the **Hopex Power Supervisor** technical module is available on the site, ensuring that the user who executed the update has the corresponding writing access in the repository.

     ☛ *For command files with the MGR extension (repository backup), absolute identifiers are included in the imported objects and writing access levels are maintained.*

     ☛ *For command files with the MGL extension (log extraction or backup logfile), the absolute identifiers are included in the imported* *objects. The writing access levels are maintained if the updates are consistent with the writing access diagram for the environment.*

☛ *These controls are not carried out if the user level is "Administrator", this enables the data restorations.*

7. In the **Filters** pane, select the import behavior to be applied:
   - **Standard Reprocessing** changes creation of an already existing object into a modification, or into creation of an object of the same name preceded by a number if their absolute identifiers are different.
   - **Reassign User** ignores the writing accesses contained in the imported file. All elements in the imported file are given the same writing access level as the user executing the import. This is useful when you have the **Hopex Power Supervisor** technical module. The creator and modifier names are replaced with the name of the user executing the import.

     ☛ *It is recommended that you enable this option when the import file comes from an environment where the writing access diagram is not the same as the one for the environment where this file is being imported.*
8. (Optional: **Windows Front-End**) Select the **Log Updates** option if you want to update the *repository log*, if this log will be exported to another workstation without the file being imported.

   This option is an advanced operation, Bizzdesign recommends that you contact Bizzdesign support before selecting this option.
9. Selecting **Include Object Logs** allows you to also import object histories.

   ☛ *For more information on object logs, see Viewing Object History.*

   The options you select are controlled by the software, based on the file extension and the standard processing to be applied. If your choices are not consistent with the file extension, a message box informs you of this fact and its possible consequences.

   ☛ *For more details on the main causes of rejects, see Dispatch Conflicts and Rejects When Dispatching.*
10. Click **Import**.
    The report window appears showing the import progress.
    The **Processing** pane details the number of commands accepted and rejected.

When the import contains errors:

- a reject report file is generated.

  ☛ *See Viewing rejects.*
- an execution report file is available.

  ☛ *See Viewing the import execution report file.*

Viewing rejects

To view the rejects (or errors) recorded during the import of the command file:

1. Import the command file

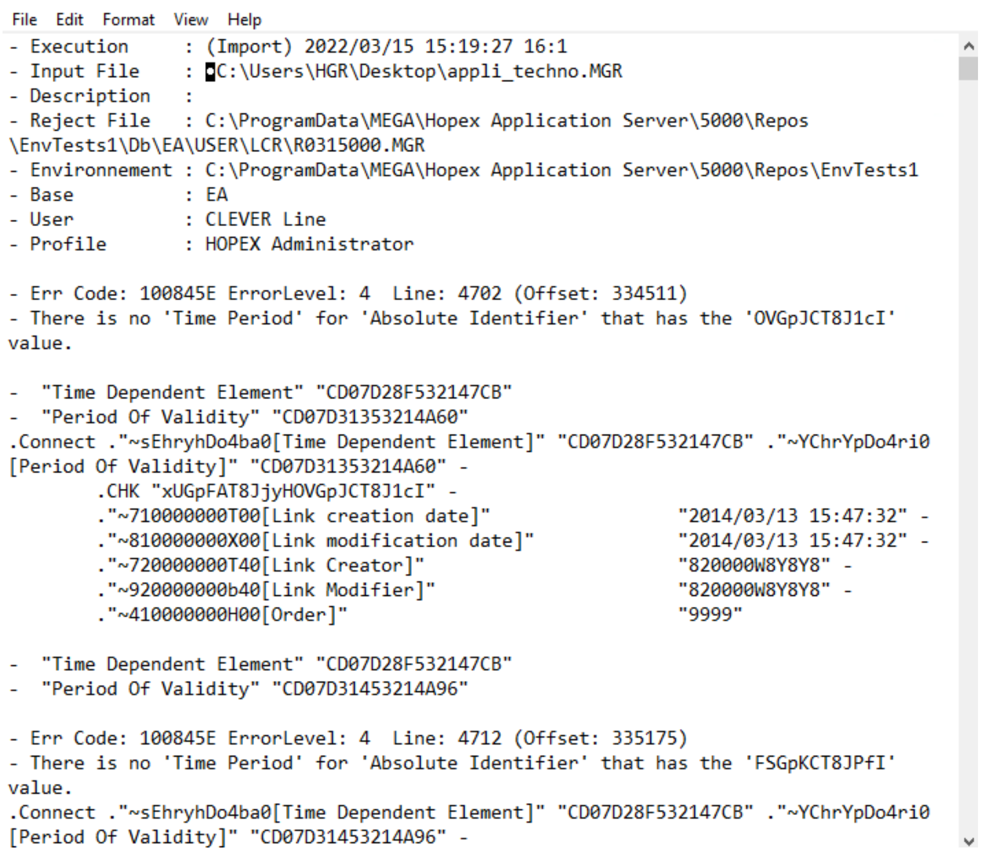
   ☛ *See Importing command files.*

2. Click the **Report File** button.

   ☛ *The contents of the report file depend on import options. For more details on importing a command file, see Options.*

Case of a text file import (MGR, MGL)

The report file appears and details all the rejects.



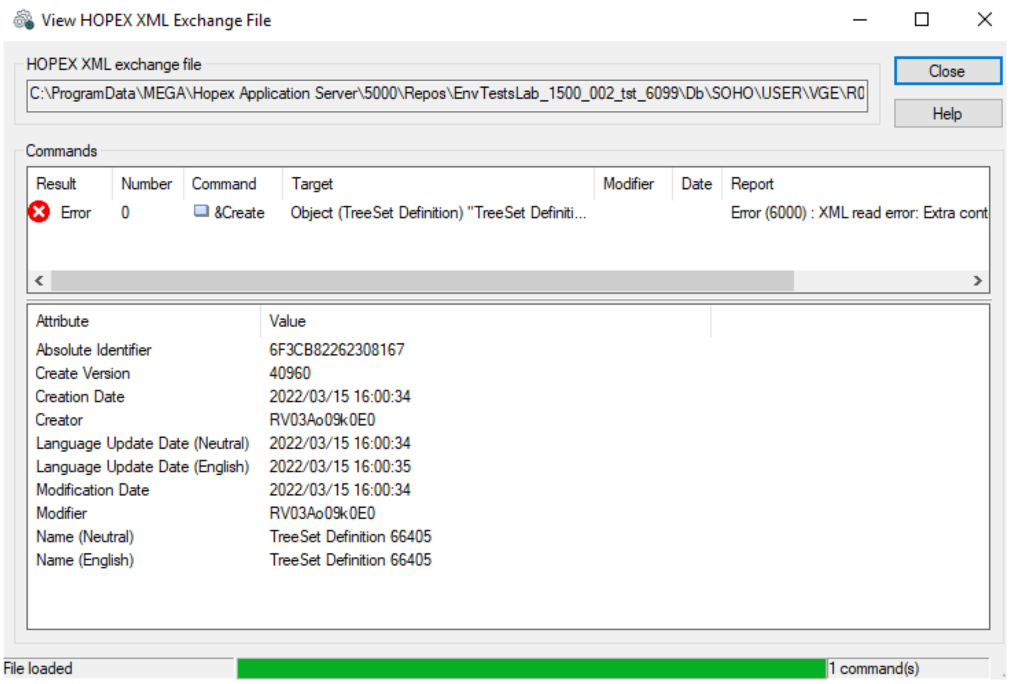
```
- Execution     : (Import) 2022/03/15 15:19:27 16:1
- Input File    : C:\Users\HGR\Desktop\appli_techno.MGR
- Description   :
- Reject File   : C:\ProgramData\MEGA\Hopex Application Server\5000\Repos
\EnvTests1\Db\EA\USER\LCR\R0315000.MGR
- Environnement : C:\ProgramData\MEGA\Hopex Application Server\5000\Repos\EnvTests1
- Base          : EA
- User          : CLEVER Line
- Profile       : HOPEX Administrator

- Err Code: 100845E ErrorLevel: 4  Line: 4702 (Offset: 334511)
- There is no 'Time Period' for 'Absolute Identifier' that has the 'OVGpJCT8J1cI'
value.

-  "Time Dependent Element" "CD07D28F532147CB"
-  "Period Of Validity" "CD07D31353214A60"
.Connect ."~sEhryhDo4ba0[Time Dependent Element]" "CD07D28F532147CB" ."~YChrYpDo4ri0
[Period Of Validity]" "CD07D31353214A60" -
        .CHK "xUGpFAT8JjyHOVGpJCT8J1cI" -
        ."~710000000T00[Link creation date]"              "2014/03/13 15:47:32" -
        ."~810000000X00[Link modification date]"          "2014/03/13 15:47:32" -
        ."~720000000T40[Link Creator]"                    "820000W8Y8Y8" -
        ."~920000000b40[Link Modifier]"                   "820000W8Y8Y8" -
        ."~410000000H00[Order]"                           "9999"

-  "Time Dependent Element" "CD07D28F532147CB"
-  "Period Of Validity" "CD07D31453214A96"

- Err Code: 100845E ErrorLevel: 4  Line: 4712 (Offset: 335175)
- There is no 'Time Period' for 'Absolute Identifier' that has the 'FSGpKCT8JPfI'
value.
.Connect ."~sEhryhDo4ba0[Time Dependent Element]" "CD07D28F532147CB" ."~YChrYpDo4ri0
[Period Of Validity]" "CD07D31453214A96" -
```

Example of reject file at MGR file import

Case of a MEGA XML import (Windows Front-End)

The view window for the report file appears and details the **Commands** contained in the imported file and the **Result** of execution of each command:

- ***Accepted***: the command is accepted
- ***Error***: the command is rejected
- ***Warning***: the command is accepted, but contains anomalies


View HOPEX XML Exchange File

HOPEX XML exchange file
C:\ProgramData\MEGA\Hopex Application Server\5000\Repos\EnvTestsLab_1500_002_tst_6099\Db\SOHO\USER\VGE\R0

Commands

Result	Number	Command	Target	Modifier	Date	Report
Error	0	&Create	Object (TreeSet Definition) "TreeSet Definiti...			Error (6000) : XML read error: Extra cont

Attribute	Value
Absolute Identifier	6F3CB82262308167
Create Version	40960
Creation Date	2022/03/15 16:00:34
Creator	RV03Ao09k0E0
Language Update Date (Neutral)	2022/03/15 16:00:34
Language Update Date (English)	2022/03/15 16:00:35
Modification Date	2022/03/15 16:00:34
Modifier	RV03Ao09k0E0
Name (Neutral)	TreeSet Definition 66405
Name (English)	TreeSet Definition 66405

File loaded
1 command(s)

Viewing the import execution report file

To view the import execution report:

1. Import the command file

   ☛ *See Importing command files.*

2. Click **Report**.
   The report details the number of commands analyzed, executed and rejected for each command type.

Viewing the Environment Report File

The environment report file, MegaCrdYYYYmm.txt (where YYYY and mm represent year and month of creation) indicates all administration operations (backup, export, restore, controls, etc.) carried out in the environment. The report file is stored in the user work folder associated with the environment system repository: SYSDB\USER\XXX\XXX.WRI where XXX is the user code.

After you have executed backup and restore operations on a repository, you can view the environment *report file*.

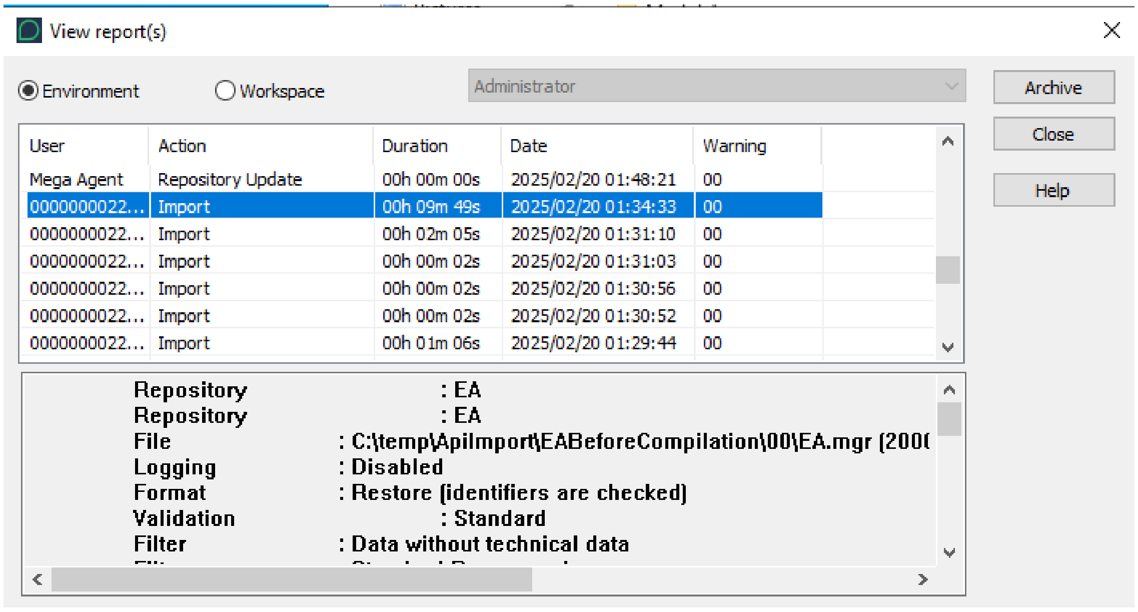
Viewing the environment report file

To open the environment report view window:

1. From **Hopex Administration**, connect to the environment.

   ☛ *See Connecting to an Environment.*

2. Right-click the environment and select **Reports > Open**.


View report(s)

Environment [✓] Workspace — Administrator

Archive / Close / Help

User	Action	Duration	Date	Warning
Mega Agent	Repository Update	00h 00m 00s	2025/02/20 01:48:21	00
0000000022...	Import	00h 09m 49s	2025/02/20 01:34:33	00
0000000022...	Import	00h 02m 05s	2025/02/20 01:31:10	00
0000000022...	Import	00h 00m 02s	2025/02/20 01:31:03	00
0000000022...	Import	00h 00m 02s	2025/02/20 01:30:56	00
0000000022...	Import	00h 00m 02s	2025/02/20 01:30:52	00
0000000022...	Import	00h 01m 06s	2025/02/20 01:29:44	00

```
Repository          : EA
Repository          : EA
File        : C:\temp\ApiImport\EABeforeCompilation\00\EA.mgr (200(
Logging     : Disabled
Format      : Restore (identifiers are checked)
Validation          : Standard
Filter      : Data without technical data
```

This dialog box also allows you to view report files of each user of the environment. Indicated for each action are:

- **User** who executed the operation.
- **Action** executed: dispatch, import, installation, translation, export, derivation, extraction, backup, update, user diagram extraction, user diagram import, repository translation, repository or environment check, repository creation, repository deletion, log initialization, object protection, etc.
- **Date** the operation began.
- **Duration** of the operation.
- return code in the **Warning** column:
  - Error level 0: no error.
  - Error level 2: minor errors such as attempts at creating already existing objects or links.
  - Error level 4: the actions involved nonexistent objects or links.
  - Error level 8: a system error was encountered during the update.
  - Error level 16: update aborted because of a problem such as insufficient disk space.

when you select an action, the lower frame displays:

- the repository concerned
- the file used
- details of the action.

Copying the environment report file

To copy and reinitialize the environment report file:

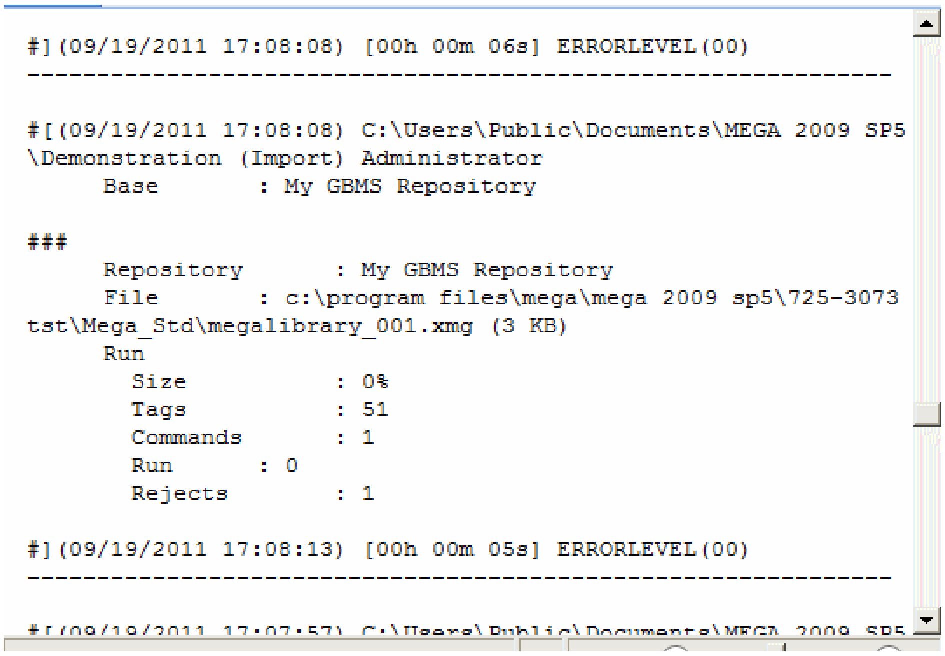
1. In the **View Report(s)** window (see Viewing the Environment Report File), click **Archive**.
   A confirmation request message appears.
2. Select **Yes**.

☛ *You should reinitialize this file (or archive with the network version) from time to time.*

Opening the environment report file

To open the "MegaCrdYYYYmm.txt" file with Wordpad:

1. From the explorer, localize the "MegaCrdYYYYmm.txt"" file in the environment folder.
2. Right-click the file and select **Open with > Wordpad**.



```
#](09/19/2011 17:08:08) [00h 00m 06s] ERRORLEVEL(00)
---------------------------------------------------------------

#[(09/19/2011 17:08:08) C:\Users\Public\Documents\MEGA 2009 SP5
\Demonstration (Import) Administrator
      Base        : My GBMS Repository

###
      Repository        : My GBMS Repository
      File        : c:\program files\mega\mega 2009 sp5\725-3073
tst\Mega_Std\megalibrary_001.xmg (3 KB)
      Run
        Size            : 0%
        Tags            : 51
        Commands        : 1
        Run       : 0
        Rejects         : 1

#](09/19/2011 17:08:13) [00h 00m 05s] ERRORLEVEL(00)
---------------------------------------------------------------

#[(09/19/2011 17:07:57) C:\Users\Public\Documents\MEGA 2009 SP5
```

Viewing User Process Error Trace Files

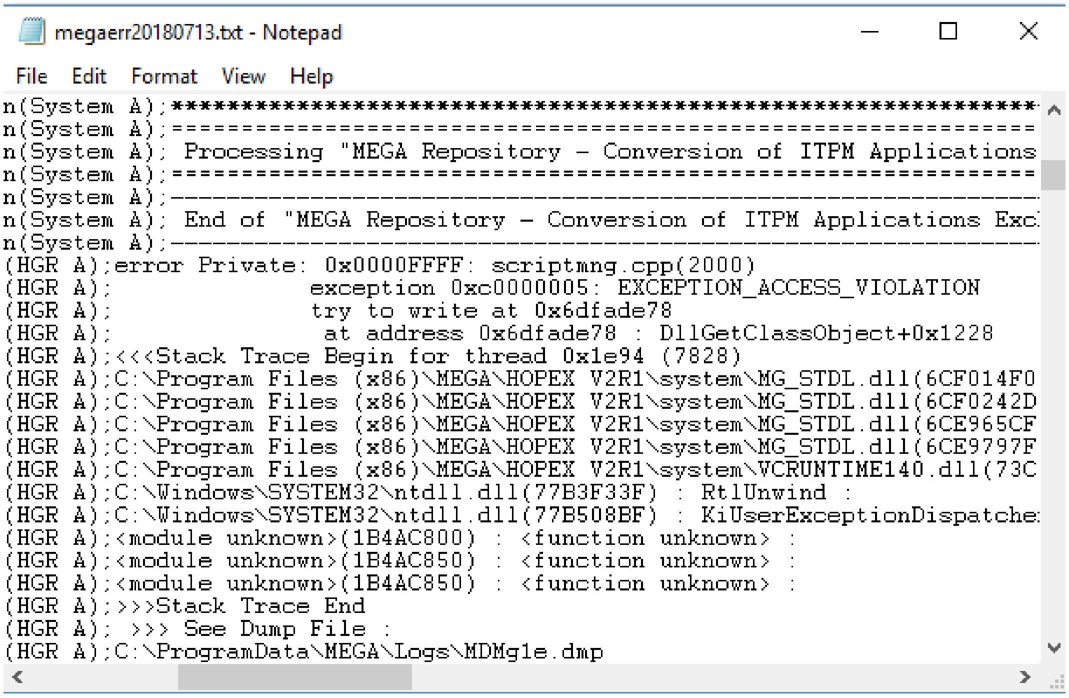
Trace files of errors in user processes contain information on operations executed and possible anomalies.

If there is a problem with a repository, this file will help the **Bizzdesign** Research Center to analyze it.

Each *trace file* has the following characteristics:

- format: *.txt
- name: megaerrYYYYmmDD
  where YYYYmmDD represent the year, month and day when the file was generated

  ```
  Example: file Megaerr20110204.txt was generated on 04
  February 2014).
  ```
- information on errors:
  - date and time of the error
  - action that produced the error
  - associated error message



```
n(System A);*************************************************************
n(System A);=============================================================
n(System A); Processing "MEGA Repository - Conversion of ITPM Applications
n(System A);=============================================================
n(System A);-------------------------------------------------------------
n(System A); End of "MEGA Repository - Conversion of ITPM Applications Exc
n(System A);-------------------------------------------------------------
(HGR A);error Private: 0x0000FFFF: scriptmng.cpp(2000)
(HGR A);              exception 0xc0000005: EXCEPTION_ACCESS_VIOLATION
(HGR A);              try to write at 0x6dfade78
(HGR A);               at address 0x6dfade78 : DllGetClassObject+0x1228
(HGR A);<<<Stack Trace Begin for thread 0x1e94 (7828)
(HGR A);C:\Program Files (x86)\MEGA\HOPEX V2R1\system\MG_STDL.dll(6CF014F0
(HGR A);C:\Program Files (x86)\MEGA\HOPEX V2R1\system\MG_STDL.dll(6CF0242D
(HGR A);C:\Program Files (x86)\MEGA\HOPEX V2R1\system\MG_STDL.dll(6CE965CF
(HGR A);C:\Program Files (x86)\MEGA\HOPEX V2R1\system\MG_STDL.dll(6CE9797F
(HGR A);C:\Program Files (x86)\MEGA\HOPEX V2R1\system\VCRUNTIME140.dll(73C
(HGR A);C:\Windows\SYSTEM32\ntdll.dll(77B3F33F) : RtlUnwind :
(HGR A);C:\Windows\SYSTEM32\ntdll.dll(77B508BF) : KiUserExceptionDispatche
(HGR A);<module unknown>(1B4AC800) : <function unknown> :
(HGR A);<module unknown>(1B4AC850) : <function unknown> :
(HGR A);<module unknown>(1B4AC850) : <function unknown> :
(HGR A);>>>Stack Trace End
(HGR A); >>> See Dump File :
(HGR A);C:\ProgramData\MEGA\Logs\MDMg1e.dmp
```

You can open the trace file from:

- **Hopex Administration**
- **Hopex Server Supervisor**
- **Hopex**

Opening the trace file from Hopex Administration

To open the trace file from **Hopex Administration**:

1. Connect to **Hopex Administration**.

   ☛ *See Accessing Hopex Administration.*

2. In the menu bar, select **Help > Technical Support > Error Logfile > Edit**.

Opening the trace file from the Hopex Server Supervisor tool

To open the trace file from the **Hopex Server Supervisor** tool:

1. (Prerequisite) The **Hopex Server Supervisor** tool is started, see Starting Hopex Server Supervisor.
2. In your workstation system tray, right-click **Hopex Server Supervisor** and select **Mega Logs > Open Daily Logs**.

Opening the trace file from Hopex

To open the trace file from **Hopex**:

1. Connect to **Hopex**.

   ☛ *See Connecting to Hopex Customizing Desktop.*

2. In the **Hopex** menu bar, select **Help > Technical Support > Error Logfile > Edit**.

   ☛ *Alternative: in the **Hopex** menu bar, select **Help > About Hopex**, then in **System Information**, select **Error Log > Edit**.*

Saving the Error Zip file for Diagnostics

The **Hopex Server Supervisor** tool allows you to save a zip file containing information required by the **Bizzdesign** Research Center to help in repository problem diagnostics.

This zip file of errors contains in particular the trace files of errors related to user processes (megaerrYYYYmmDD.txt).

Prerequisite: **Hopex Server Supervisor** is started, see Starting Hopex Server Supervisor.

To save the error zip file for diagnostics:

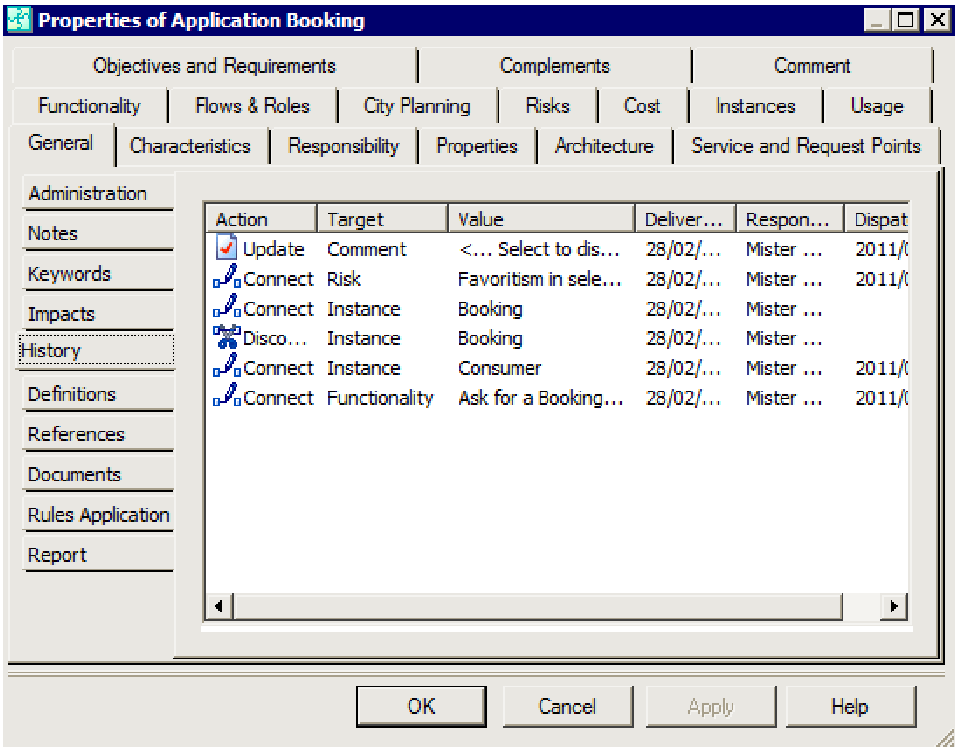
1. In your workstation system tray, right-click **Hopex Server Supervisor** and select **Logs > Daily Logs manager**.
2. Click **Zip**.
3. Specify a saving location and enter the zip file name.
4. Click **Save**.

Viewing Object History

The object history is a different view of the repository log: instead of browsing actions performed by a user in a dispatch, the object history shows actions carried out from an object for all dispatches and users.

☛ *The repository log must be enabled so that the object history can be supplied, see Repository log.*

In the properties dialog box of an object, the **History** subtab of the **General** tab gives an overview of actions on each object in the repository. At each update concerning the object (Create, Modify, Connect, Disconnect), the corresponding action is added to the list.



Action	Target	Value	Deliver...	Respon...	Dispat
Update	Comment	<... Select to dis...	28/02/...	Mister ...	2011/(
Connect	Risk	Favoritism in sele...	28/02/...	Mister ...	2011/(
Connect	Instance	Booking	28/02/...	Mister ...	
Disco...	Instance	Booking	28/02/...	Mister ...	
Connect	Instance	Consumer	28/02/...	Mister ...	2011/(
Connect	Functionality	Ask for a Booking...	28/02/...	Mister ...	2011/(

Viewing Dispatch Report

The report file can be accessed from **Hopex Administration**.

To view the report file:

1. Connect to **Hopex Administration**.

   ☛ *See Connecting to an Environment.*

2. Right-click the environment and select **Reports > Open**.
   A new window allows you to view report contents.

   ☛ *See Viewing the Environment Report File for more details on these dialog boxes.*

The error level is indicated in the dispatch report.

- Error level 0: no error.
- Error level 2: minor errors such as attempts at creating already existing objects or links.
- Error level 4: the actions involved nonexistent objects or links.
- Error level 8: a system error was encountered during the update.
- Error level 16: update aborted because of a problem such as insufficient disk space.

☛ *These error levels are the same as those used for manual file imports.*

☛ *When manually importing a file, rejects concerning the creation of already existing objects or links can be filtered out using the Reprocess option.*

Objects that were renamed are also listed in the report.

OPTIMIZING REPOSITORY ACCESS PERFORMANCE

To optimize **Hopex** performance, you must also remember to optimize your repository size. You must:

- reduce the log size

  ☛ *See Managing Log Size.*
- increase the cache size

  ☛ *See Increasing RDBMS cache size (memory).*
- delete temporary data and history data (RDBMS repository) regularly

  ☛ *See Following the Deletion of Repository Temporary Data and Historical Data.*
- perform regular maintenance tasks of RDBMS repositories

  ☛ *See Performing Repository Regular Maintenance Tasks.*
- reduce quantity of status indicators

  ☛ *See Managing Status Indicators.*
- clean up repository

  ☛ *See Cleaning up a Repository.*
- configure anti-virus actions

  ☛ *See Configuring the Anti-Virus According to Hopex Data.*
- reorganize repository

  ☛ *See Reorganizing an RDBMS Repository.*

Managing Log Size

☛ *Managing the log size is only necessary if you have enabled the repository log, see Repository log.*

To reduce the log size, you can:

- delete all the log commands earlier than a selected date

  ☛ *See Deleting a log or reducing the log size.*
- (SQL Server) select and delete log elements, earlier than a selected date

  ☛ *See Deleting log elements to reduce the log size.*
- modify MetaClass loggability

  ☛ *See Modifying MetaClass loggability.*

Log size management frequency

You must reduce the log size:

- every month, for configurations of less than 50 users.
- every week, for configurations of more than 50 users.

Deleting a log or reducing the log size

Prerequisite: before deleting your log (complete or partial deletion), Bizzdesign recommends that you back up it.

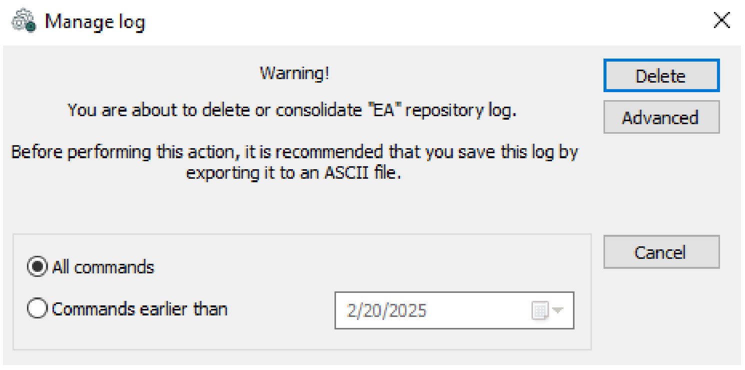
☛ *see Backing up the repository log.*

To delete a log or to reduce its size:

1. Connect to **Hopex Administration** and select the repository concerned.

   ☛ *See Accessing Repositories.*

2. Right-click the repository and select **Repository Log > Manage Repository and Object Log**.
   The **Manage Log** window opens.



Manage log

Warning!

You are about to delete or consolidate "EA" repository log.

Before performing this action, it is recommended that you save this log by exporting it to an ASCII file.

(●) All commands
( ) Commands earlier than 2/20/2025

3. Define the commands to be deleted:
   - **All commands** or
   - **Commands earlier than** the date you select using the drop-down menu calendar.
4. Click **Delete** to delete all the commands contained within the selected time interval.

   ☛ *For a more specific deletion, see Deleting log elements to reduce the log size.*

Backing up the repository log

To back up the repository log you have to export the repository log in a file and save it.

This file can be exported in:

- **logfile text** (.mgl).
  Name format of the exported file is "LOGmmdd.mgl", where "mmdd" represents logfile export date month and day.
- **XML MEGA** (.xmg)
  The exported file is in the form of an XML file containing commands or data (objects and links).

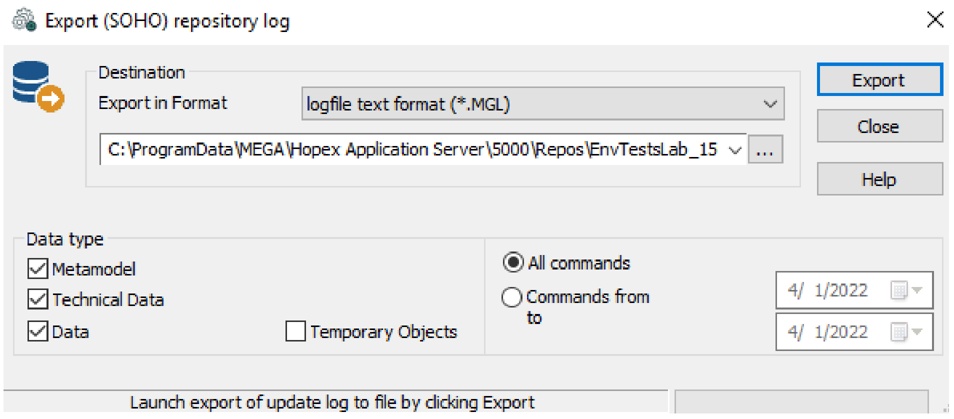
To avoid backup duplications and save backup time and size, select **Commands from** "selected date" **to** "selected date" and select a starting date corresponding to the end date of your last backup, and select the ending date you require.

To backup the repository log in the form of command file:

1. Connect to **Hopex Administration** and select the repository concerned.

   ☛ *See Accessing Repositories.*

2. Right-click the repository and select **Repository Log > Export**.
   The **Export repository log** dialog box opens.



Export (SOHO) repository log

Destination
Export in Format: logfile text format (*.MGL)
C:\ProgramData\MEGA\Hopex Application Server\5000\Repos\EnvTestsLab_15

Data type
☑ Metamodel
☑ Technical Data
☑ Data
☐ Temporary Objects

(●) All commands
( ) Commands from 4/ 1/2022
to 4/ 1/2022

Launch export of update log to file by clicking Export

3. Select export format.
4. (If needed) By default the file is saved in **<environment folder>\db\<repository name>\WORK** folder. Modify the default name and folder to save the export file. The **Browse** button ... allows you to browse the folder tree and select the folder in which the file will be saved.
5. In the **Data type** pane, select the type of exported modifications:
   - **Metamodel**: to extract the metamodel from the system repository. This is useful if the standard metamodel has been modified.
   - **Technical Data**: to export, from the system repository, the changes made to data such as descriptors and queries.
   - **Data**: to export changes made to repository data, particularly the workspaces.
   - **Temporary Objects** - these objects are created when executing requests, consulting objects (stored in the history), etc. Usually you do not need to export these objects.
6. To export the commands that correspond to a defined period only, select **Commands from** / **to** and specify the relevant dates.
7. Click **Export**.
   The **Execution Report** appears.
8. Click **OK**.
   The file is exported and saved in the specified folder.

Deleting log elements to reduce the log size

You can delete log elements more specifically: you can delete only log elements relating to specific MetaClasses, earlier than a selected date.

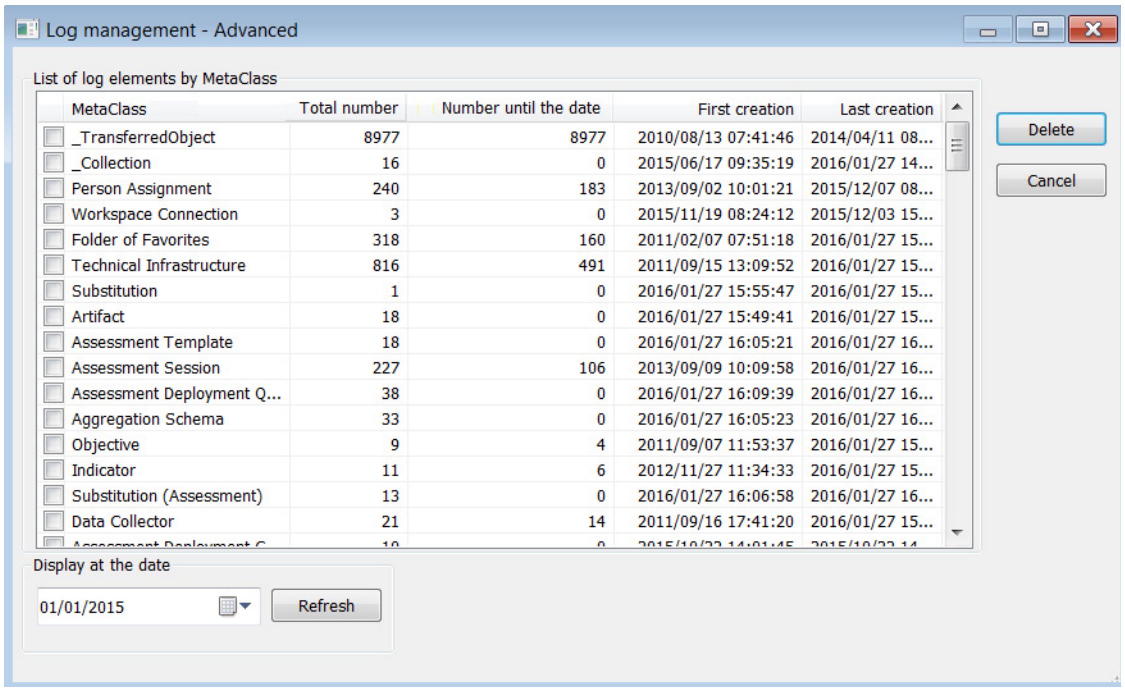
☛ *If you do not want to have to delete log elements you are not interested in, see Modifying MetaClass loggability.*

To delete log elements regarding specific MetaClasses:

1. Connect to **Hopex Administration** and select the repository concerned.

   ☛ *See Accessing Repositories.*

2. Right-click the repository and select **Repository Log > Manage Repository and Object Log**.
   The **Manage Log** dialog box opens.
3. Click **Advanced**.
   The **Log Management - Advanced** window shows the log element number by MetaClass and specifies the first and last creation date.



Log management - Advanced

List of log elements by MetaClass

MetaClass	Total number	Number until the date	First creation	Last creation
_TransferredObject	8977	8977	2010/08/13 07:41:46	2014/04/11 08...
_Collection	16	0	2015/06/17 09:35:19	2016/01/27 14...
Person Assignment	240	183	2013/09/02 10:01:21	2015/12/07 08...
Workspace Connection	3	0	2015/11/19 08:24:12	2015/12/03 15...
Folder of Favorites	318	160	2011/02/07 07:51:18	2016/01/27 15...
Technical Infrastructure	816	491	2011/09/15 13:09:52	2016/01/27 15...
Substitution	1	0	2016/01/27 15:55:47	2016/01/27 15...
Artifact	18	0	2016/01/27 15:49:41	2016/01/27 15...
Assessment Template	18	0	2016/01/27 16:05:21	2016/01/27 16...
Assessment Session	227	106	2013/09/09 10:09:58	2016/01/27 16...
Assessment Deployment Q...	38	0	2016/01/27 16:09:39	2016/01/27 16...
Aggregation Schema	33	0	2016/01/27 16:05:23	2016/01/27 16...
Objective	9	4	2011/09/07 11:53:37	2016/01/27 15...
Indicator	11	6	2012/11/27 11:34:33	2016/01/27 15...
Substitution (Assessment)	13	0	2016/01/27 16:06:58	2016/01/27 16...
Data Collector	21	14	2011/09/16 17:41:20	2016/01/27 15...
Assessment Deployment C...	10	0	2015/10/22 14:01:45	2015/10/22 14...

Display at the date

01/01/2015

4. In the **Display at the date** pane, select the date until which you want to delete the log elements.
5. Click **Refresh**.
   The **Number until the date** column shows for each MetaClass the log element number until the selected date.

   ☺ *Click the **Number until the date** column header to sort the MetaClasses, with the most populated at the top.*

6. In the **MetaClass** column, select the MetaClass for which you want to delete the log elements until the selected date.

   ☛ *You can select several MetaClasses.*

7. Click **Delete**.
   Log elements are deleted.

Modifying MetaClass loggability

To reduce the number of objects generated in logs, you can modify logging of MetaClasses you do not want to track.

☛ *See Configuring the logging.*

To modify MetaClass loggability:

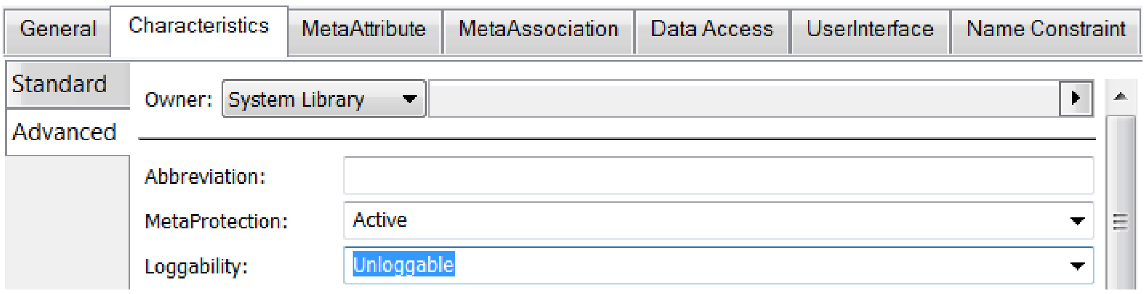
1. Connect to **Hopex**.

   ☛ *See Connecting to Hopex Customizing Desktop.*

2. Open the **MetaStudio** navigation window and expand folders **MetaClass > MetaModel**.

   ☛ *Alternative: click **Explore** and use explorer to find the **MetaClass** or **MetaAssociation** object.*

3. Open the properties dialog box of a **MetaClass** or **MetaAssociation**.
4. In the **Characteristics > Advanced** tab, for the loggability attribute, select "Unloggable" value.



Owner: System Library
Abbreviation:
MetaProtection: Active
Loggability: Unloggable

Occurrences created, updated or deleted in a private workspace are dispatched, but certain commands are not available in object histories or in repository activity.

Managing the Cache in RDBMS Environments

The RDBMS local cache avoid multiple requests when multiple users are on the same repository view. Access for the following users is speed up.

By default the RDBMS local cache is activated on all the repositories:

- You must configure your anti-virus accordingly.

  ☛ *See Configuring the Anti-Virus According to Hopex Data.*

- You can increase RDBMS cache size (memory).

Increasing RDBMS cache size (memory)

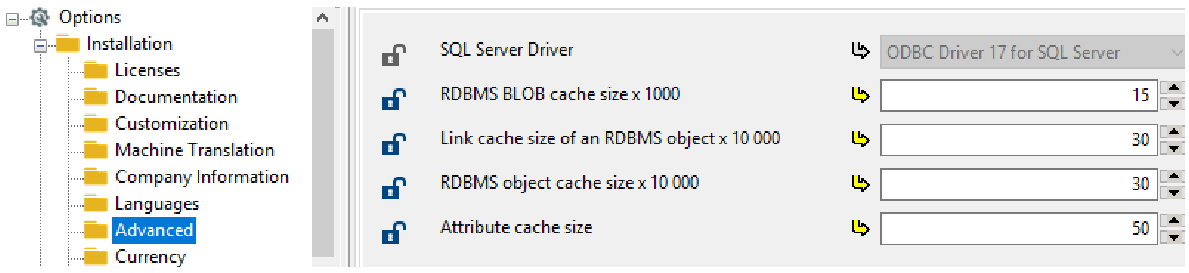
With repositories including a large amount of objects, we advise you to increase the size of RDBMS caches. The larger your cache space, the fewer the network exchanges and the better your **Hopex** performance.

To increase cache size:

1. Connect to **Hopex Administration**.

   ☛ *See Connecting to an Environment.*

2. Right-click the environment and select **Options > Modify**.
3. In the environment options, select **Options > Installation > Advanced**.



SQL Server Driver	ODBC Driver 17 for SQL Server
RDBMS BLOB cache size x 1000	15
Link cache size of an RDBMS object x 10 000	30
RDBMS object cache size x 10 000	30
Attribute cache size	50

4. Increase cache sizes according to your memory space.

Managing Status Indicators

Use of status indicators generates large quantities of queries on repositories. Select only those indicators you require.

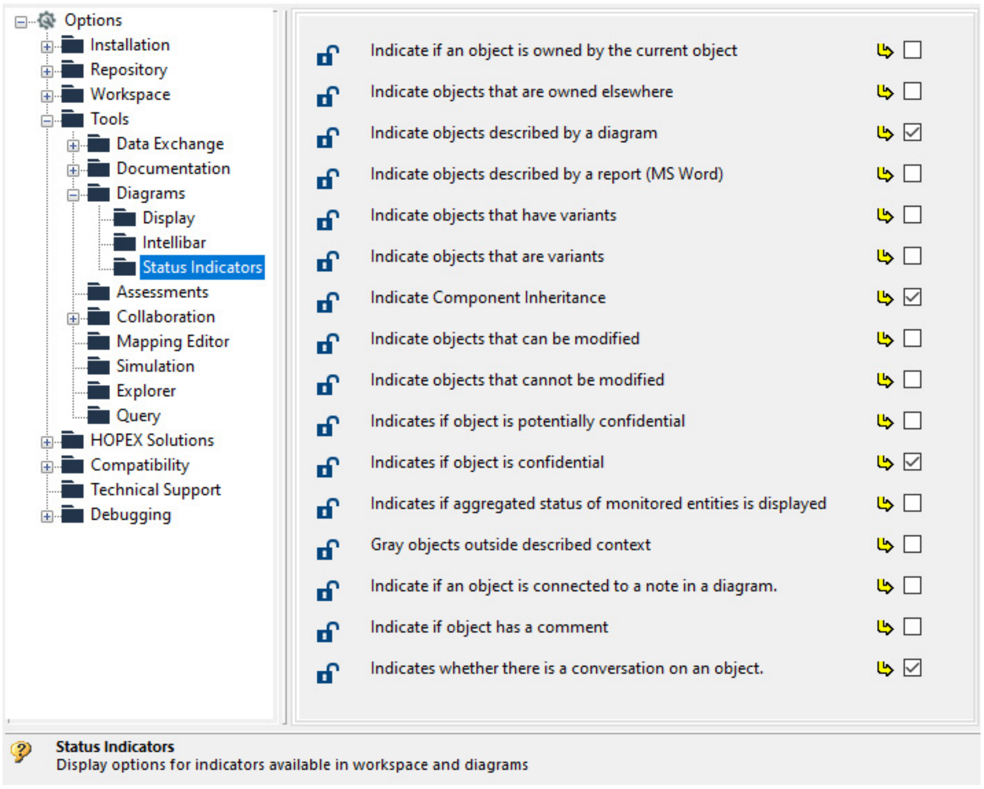
To modify indicator backup selection:

1. Connect to **Hopex Administration** and select the environment in which the repository is referenced.

   ☛ *See Connecting to an Environment.*

2. Right-click the environment and select **Options > Modify**.

3. In the **Environment Options** dialog box, select **Tools > Diagrams > Status Indicators**.



Indicate if an object is owned by the current object	☐
Indicate objects that are owned elsewhere	☐
Indicate objects described by a diagram	☑
Indicate objects described by a report (MS Word)	☐
Indicate objects that have variants	☐
Indicate objects that are variants	☐
Indicate Component Inheritance	☑
Indicate objects that can be modified	☐
Indicate objects that cannot be modified	☐
Indicates if object is potentially confidential	☐
Indicates if object is confidential	☑
Indicates if aggregated status of monitored entities is displayed	☐
Gray objects outside described context	☐
Indicate if an object is connected to a note in a diagram.	☐
Indicate if object has a comment	☐
Indicates whether there is a conversation on an object.	☑

Status Indicators
Display options for indicators available in workspace and diagrams

4. In the right pane, select only those indicators you require.
5. Click **OK**.

Following the Deletion of Repository Temporary Data and Historical Data

To prevent repository size increase and keep optimized performances, data of the completed private workspaces of **Hopex** users are regularly deleted through the ***Hopex SQL Server Maintenance Procedures Job*** scheduled job.

This includes the following procedures:

> ☛ *These procedures are detailed in* ***RDBMS Repository Installation Guide*** *deployment guide.*

- SP_CLEAN_MEGA_DATABASE

> ☛ *To consult the date of last private workspace cleanup (last execution of this procedure), see Consulting and Modifying Repository Properties.*

- SP_CONSOLIDATE_MEGA_DATABASE

> ☛ *To consult the date of last consolidation (last execution of this procedure), see Consulting and Modifying Repository Properties.*

Performing Repository Regular Maintenance Tasks

With RDBMS repositories you must include a regular maintenance plan. You (or your database administrator) should, for each repository (SystemDb repository included), perform the following maintenance tasks regularly (at least once a week):

- rebuild indexes
- update the statistics
- (optional) shrink the logs regarding the policies

Cleaning up a Repository

During modeling work, handling different objects results in creation of temporary objects (temporary queries, etc.). It is probable that these objects will subsequently be unnecessary. To avoid unnecessarily increasing repository size, you can delete these temporary objects.

To clean up a repository:

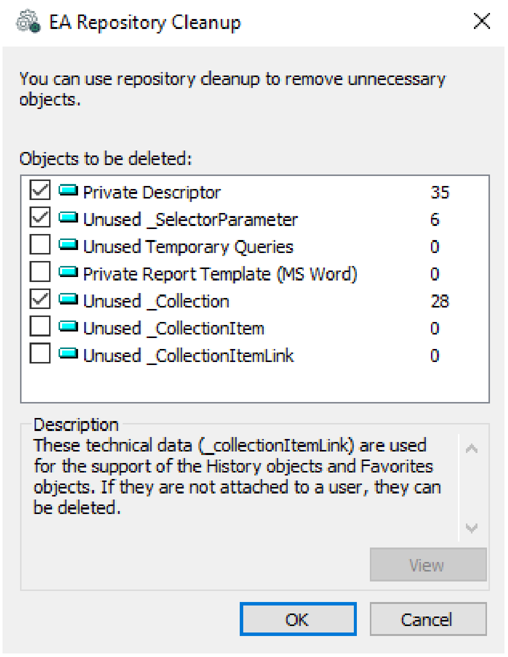
1. Connect to **Hopex Administration** and select the repository concerned.

> ☛ *See Accessing Repositories.*

2. Right-click the repository concerned and select **Object Management > Repository cleanup**.

   ☛ *Alternatively, connect to **Hopex** and in the menu bar, select **File > Properties**. In the properties of the repository to which you are connected, select **Characteristics**.*

   The **Repository Cleanup** dialog box opens.



EA Repository Cleanup

You can use repository cleanup to remove unnecessary objects.

Objects to be deleted:

☑	Private Descriptor	35
☑	Unused _SelectorParameter	6
☐	Unused Temporary Queries	0
☐	Private Report Template (MS Word)	0
☑	Unused _Collection	28
☐	Unused _CollectionItem	0
☐	Unused _CollectionItemLink	0

Description
These technical data (_collectionItemLink) are used for the support of the History objects and Favorites objects. If they are not attached to a user, they can be deleted.

3. (Optional) To view the content of an object group, click the object group name and then click **View**.
4. Select the object groups to be deleted and click **OK**.

   ☛ *By default, all the object groups that include objects are selected.*

   Selected object groups are deleted from the repository.

Configuring the Anti-Virus According to Hopex Data

To optimize anti-virus activity and to avoid unnecessary slowdown, you must configure what should be managed or not by the anti-virus.

You must exclude the following files from anti-virus scanning:

	Access mode	Location	File type
Program files external to Hopex Files used by **Hopex** to display user interface elements in Web or Windows interfaces	***Read***	**<ProgramData>\MEGA\ Hopex Application Server** and sub-folders	Proprietary format: ***.mgs** (vectorial shapes in diagrams) Public formats: ***.gif**, ***.ico** and ***.bmp**
Full-text search indexes Files used by the workstation or server executing the search	***Reading*** by the workstation or node executing the search ***Writing*** by the Administration station and by the server programming index creation		Proprietary format: ***.ix** (indexes) Public formats: ***.log** and ***.dat**
Data cache Files used by **Hopex** to improve its performance	***Read/Write***		Public format: ***.mgc**
Import/export files Files generated or read during import/export/logical backup processes	***Reading*** by the import function ***Writing*** by export and logical backup functions	Selected by the user	Proprietary formats: ***.mgr, *.mgl,** or ***.xmg**
MUST license Note: if the anti-virus does not allow such a configuration, allow ***Reading/Writing*** for all of the files on the license oath and sub folders.	***Reading/Writing*** of the files on MUST license path and under this path	In the **HAS console > Modules > Module Settings> MegasiteSettings**, the pane to update the **megasite.ini** gives the MUST License path: [must license] path=	Public format: ***.ini** Proprietary formats: ***.tnk*** and ****.usr***
	Reading of the files on MUST license path and under this path		Proprietary format: ***.must**

REFERENCING AND UNREFERENCING A REPOSITORY

Referencing and unreferencing a repository is carried out via the drop-down menu of the repository.

See successively:

- Referencing a Repository
- Unreferencing a Repository
- Protecting the Referencing of a Repository

Referencing a Repository

If you have moved or copied a repository without using Hopex move or restore commands, you will have to reference the repository so that the environment recognizes it.

☛ *To reference a repository in another environment, the two metamodels must be identical.*

To reference a repository:

1. Connect to **Hopex Administration** and select the environment in which you want to reference the repository.

   ☛ *See Connecting to an Environment.*

2. Right-click the **Repositories** folder and select **Create reference**.
3. (If the repository is password-protected) Enter the repository password and click **OK**.

   ☛ *See Protecting the Referencing of a Repository.*

4. Select **a repository from SqlServer**.

   ☛ *For more details on **Hopex** repository server type, see the **Hopex** deployment guide.*

   The Hopex **Repository Selection** dialog box opens. This dialog box allows you to create a reference for a new repository in the environment.

5. Indicate where the repository .EMQ file is located. The repository name is automatically indicated and cannot be modified.

The repository is accessible exactly as repositories created in the normal way.

☛ *you must save and then restore a repository to move it from one environment to another. The users and metamodel of the two environments must be defined identically so that transfer occurs without reject.*

☛ *To also copy these objects, import the missing part of the metamodel. One way of doing this is to upgrade the site or the environment. Then, import the rejected commands if you used the backup-restore procedure.*

☛ *Check that the repository is not simultaneously referenced in two different environments. Compatibility errors may occur if the environments are not identical.*

Unreferencing a Repository

You can delete a repository reference from an environment. This action does not delete the repository.

> **If you delete a reference of a password-protected repository. you must know this password to reference the repository again.**

To delete a repository reference:

1. Connect to **Hopex Administration** and select the repository concerned.

   > ☛ *See Accessing Repositories.*

2. Right-click the repository of which you want to delete the reference and select **Delete Reference**.
   A message requests confirmation.

   > **If the repository is password-protected, you must know this password to reference the repository again.**

3. Click **OK** to delete the repository reference.
   The repository reference is deleted, and can be created in another environment on condition that the metamodel is identical.

   > ☛ *A repository must be referenced in only one environment. It is important to check that the reference for your repository was deleted in its original environment.*

Protecting the Referencing of a Repository

You can password-protect the referencing of a repository.

See:

- Adding a referencing password to a repository
- Modifying/Canceling a repository password

Adding a referencing password to a repository

To password-protect the referencing of a repository:

1. Connect to **Hopex Administration** and select the repository concerned.

   > ☛ *See Accessing Repositories.*

2. Right-click the repository and select **Protect with a password**.
   The **Password Protect Repository** dialog box appears.
3. In the **Password** field, enter a password and confirm it.

   > ☛ *The password must contain at least eight characters.*

4. In the **Security question** pane:
   - enter a **Question**.
   - enter the **Answer** to the question.
5. Click **OK**.
   The repository password is requested on creation of a repository referencing.

Modifying/Canceling a repository password

To modify or cancel a repository password:

1. Connect to **Hopex Administration** and select the repository concerned.

   ☛ *See Accessing Repositories.*

2. Right-click the repository concerned and select **Password Protect**.
   The verification window for the password appears.
3. In the **Enter the password of the current repository** field, enter the repository password.
4. Click **OK**.
5. In the **Password** pane:
   - To modify the existing password, enter a password and confirm it.

     ☛ *The password must contain at least eight characters.*

   - To cancel the password, leave the fields empty.
6. Click **OK**.

Environments

When multiple users are working together across a network, it may be useful to create several work environments. Another reason for creating a new environment is to provide the administrator with an environment where report templates (MS Word), queries, etc. can be modified and tested without interfering with users' work.

Environment management functions are used when several environments are available and they need to exchange data or repositories.

See:

- ✓ Using Environments
- ✓ Customizing Environments

USING ENVIRONMENTS

See:

Environment Structure

An environment groups a set of users, the repositories on which they can work, and the system repository. It is where user private workspaces, users, system data, etc. are managed.

A site includes a single environment by HAS (Hopex Application Server) instance.

☛ *For detailed information regarding HAS, see "HAS Installation guide" Installation and Deployment Technical Article.*

To work with several environments (for example a development environment and a production environment) you must create the additional environment on another HAS instance.

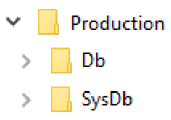
☛ *All users must have full access rights to the resources of the environments and repositories (creation and deletion of files and folders).*

An *environment* includes:

- a configuration file (MegaEnv.ini), available in the environment folder: **<HAS instance repository>> Repos > <environment name>**
- a system repository

  ☛ *See System Repository (SystemDb).*
- a “Db” folder for the data repositories used in this environment

The folder structure of an environment is as follows:



- ***Db***: contains the environment data repositories

  ☛ *Each repository is located in a folder carrying its name. See Creating a Repository.*
- ***SysDb***: the system repository

Creating an Environment

To create an environment, see **PLATFORM - Installation and deployment > RDBMS Repository Installation Guide** documentation.

You can also create an environment using modules available in Hopex store, for example to start with a blank environment or with a set of demo data.

Creating an environment using a module

If your SQL server and Hopex are installed on the same machine, you can create an environment using one of the **Hopex Databases backup** modules available in Hopex store:

- "Hopex Databases backup – Starter" module matching your Hopex version.
  This backup contains no data, you can use it to create a blank environment.
- "Hopex Databases backup - Demo - <Hopex version>" module to get the set of demo data, useful to test Hopex features.

To create a blank environment using a module from Hopex store:

1. Create an environment (i.e. an HAS instance):
   - Directly from the installation machine (or in RDP access), in Chrome go to http://localhost:30100/.
   - Connect to **HAS instance**.
   - In the **Instances** menu, click **New instance**.
   - Configure the instance

     ☛ *See **HAS Installation > Creating HAS instance** documentation.*
   - Click **Start**.
2. Add the "Hopex Databases backup – Starter" module to your new environment:
   - Log into the **HAS console** of your new environment.
   - In the navigation menus, select **Modules** > **Modules List**.
   - In the right pane, display the **Add new** page.
   - Download the "Hopex Databases backup – Starter" module version matching your Hopex version.

Moving and Referencing an Environment

During administration of **Hopex**, you may need to move an environment. When an environment is moved, you must create a new reference for it to be able to use it.

When you delete an environment, it is recommended that you delete its reference. It will then no longer appear as the environment of the corresponding **Hopex** site.

Moving an environment

When you need to move an environment, such as when you have to place it on a different drive:

1. Copy the root folder of the environment in its new location.
2. Delete the reference for the old environment.
3. Create a reference for it at its new location.
4. Carry out the same operations for each of the repositories not hosted in the environment structure.
5. Check that the reports (MS Word) and Web sites of each repository point to the correct files.

Precautions to be taken:

- Verify that there is enough free space in the destination folder.
- It is not necessary to dispatch private workspaces before moving the environment. They will be moved with the environment.
- No user should be connected during movement of environments. No private workspace should be active.

Referencing an environment

When an environment has been moved by the user, it is necessary to create a reference for this environment so the site will recognize it.

To create a reference to an environment:

1. Connect to **Hopex Administration**.

   ☛ *See Accessing Hopex Administration.*
2. In the navigation tree, right-click **Environments** and select **Create Reference**.
3. In the dialog box that opens, select the folder in which the environment to be referenced is located.
4. Click **OK** to validate.
   The new environment reference is created and appears as the environment of the site.

Deleting a reference to an environment

To delete a reference to an environment:

1. Connect to **Hopex Administration**.

   ☛ *See Accessing Hopex Administration.*
2. In the navigation tree, right-click the desired environment and select **Delete Reference**.
   A confirmation request appears.
3. Select **Yes**.
   The environment reference is deleted and no longer appears as the environment of the site.

Updating an Environment

You must update your environment at each upgrade or when installing a module.

☛ *To install a module and automatically update the environment, see Importing a Module into HOPEX.*

The update includes the following actions:

- **Environment update**
  Import of files for version upgrade, bundle upgrade (SP/CU), or additional modules.

  SP: Service Pack, CU: Cumulative Update.
- **RDBMS technical alignments**
  SQL processing: alignment of SQL Tables with the metamodel.
- **Metamodel and data conversions**
  Conversion of data in the System repository and the data repositories.

  `E.g.: conversion of names, deprecated links, profiles.`
- **Metamodel and technical data compilation**
  Compilation of the metamodel and the technical data.
- **Permission compilation**
  Compilation of permissions.

Duration:

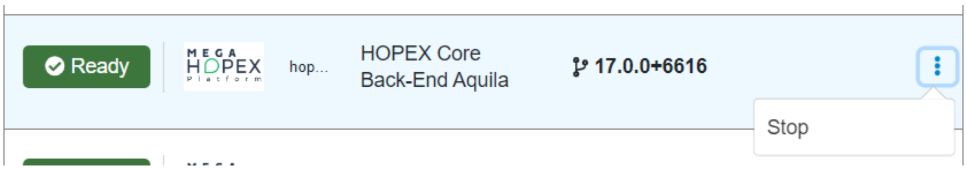
The process duration depends on the migration path, the infrastructure, and the data volume. As the process may takes several hours, check that the machine may not be shut down (turn to sleep or hibernate mode).

Indicative durations:

- CU, SP, or product installation: from 10 min to one or two hours
- major version change: more than four hours

To update an environment:

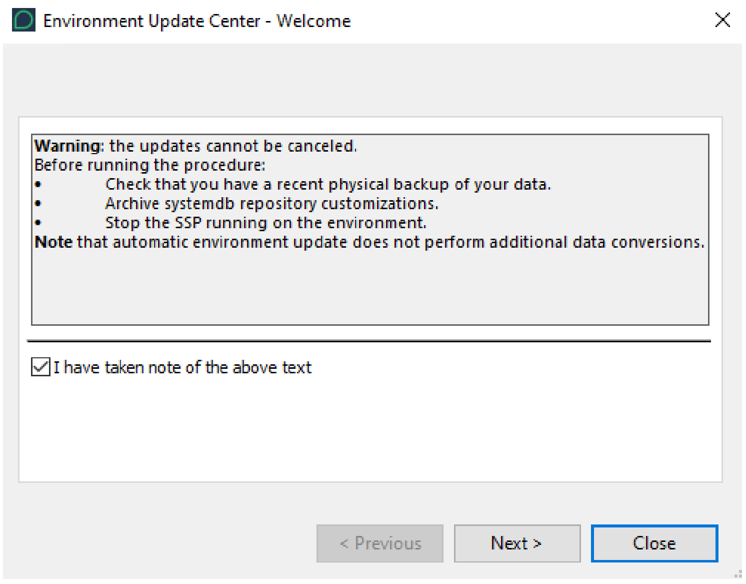
1. Stop the **Hopex Core Back-End** module.
   Make sure to perform this action when users are not connected.
   In the **HAS console**, **Cluster** navigation menu, **Modules** tab: on **Hopex Core Back-End** module, click **Plus** ⋮ **> Stop**.



2. Connect to **Hopex Administration**.

   ☛ *See Accessing Hopex Administration.*
3. Right-click the environment and select **Environment automatic update**.

4. In the **Environment Update Center** window, read and validate the ***Warning***.

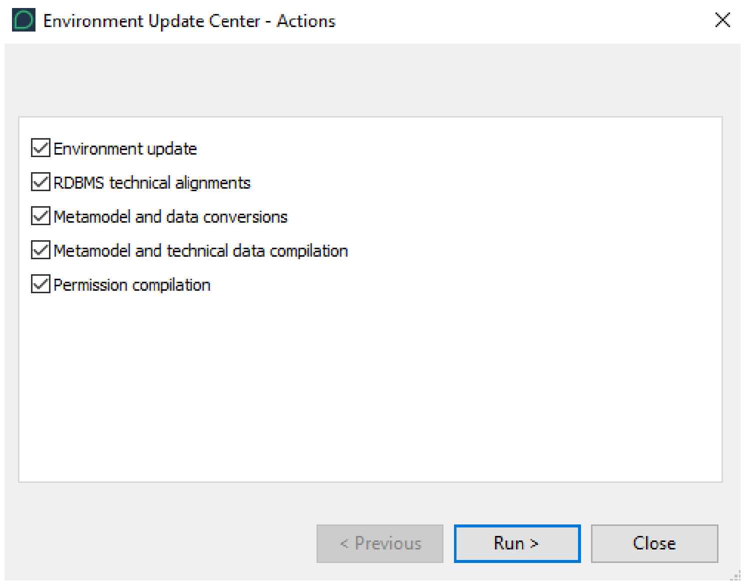


Warning: the updates cannot be canceled.
Before running the procedure:
- Check that you have a recent physical backup of your data.
- Archive systemdb repository customizations.
- Stop the SSP running on the environment.

Note that automatic environment update does not perform additional data conversions.

☑ I have taken note of the above text

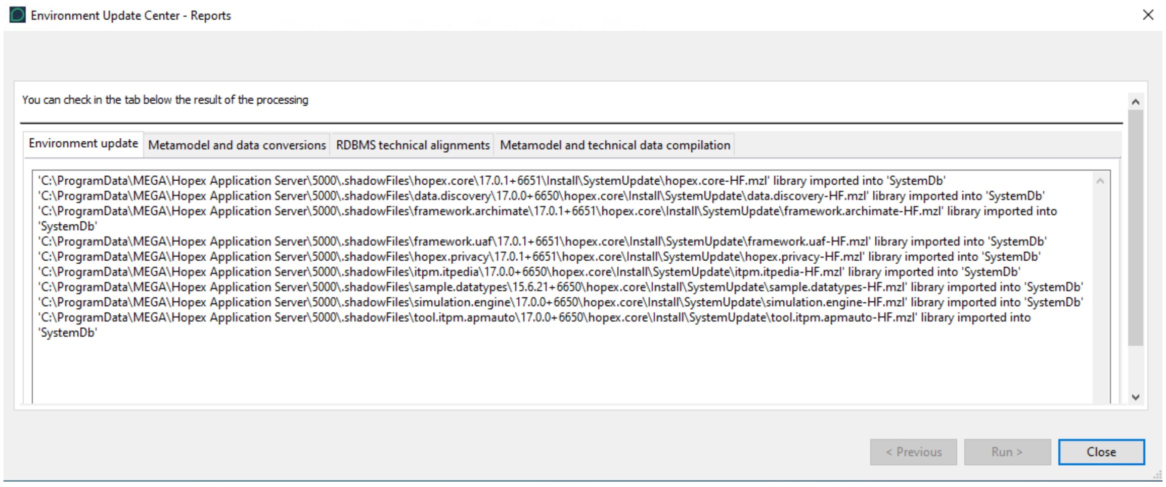
5. Click **Next**.



☑ Environment update
☑ RDBMS technical alignments
☑ Metamodel and data conversions
☑ Metamodel and technical data compilation
☑ Permission compilation

6. Keep all the selections.
If you are not in a Production environment, you can clear **Permission compilation**.
7. Click **Run**.
The update reports are displayed in four tabs (the last one cumulates both compilation steps).

8. Read each report, especially if an error was detected (error message displayed).


Environment Update Center - Reports
Environment update
Metamodel and data conversions
RDBMS technical alignments
Metamodel and technical data compilation
< Previous
Run >
Close

You can check in the tab below the result of the processing

```
'C:\ProgramData\MEGA\Hopex Application Server\5000\.shadowFiles\hopex.core\17.0.1+6651\Install\SystemUpdate\hopex.core-HF.mzl' library imported into 'SystemDb'
'C:\ProgramData\MEGA\Hopex Application Server\5000\.shadowFiles\data.discovery\17.0.0+6650\hopex.core\Install\SystemUpdate\data.discovery-HF.mzl' library imported into 'SystemDb'
'C:\ProgramData\MEGA\Hopex Application Server\5000\.shadowFiles\framework.archimate\17.0.1+6651\hopex.core\Install\SystemUpdate\framework.archimate-HF.mzl' library imported into
'SystemDb'
'C:\ProgramData\MEGA\Hopex Application Server\5000\.shadowFiles\framework.uaf\17.0.1+6651\hopex.core\Install\SystemUpdate\framework.uaf-HF.mzl' library imported into 'SystemDb'
'C:\ProgramData\MEGA\Hopex Application Server\5000\.shadowFiles\hopex.privacy\17.0.1+6651\hopex.core\Install\SystemUpdate\hopex.privacy-HF.mzl' library imported into 'SystemDb'
'C:\ProgramData\MEGA\Hopex Application Server\5000\.shadowFiles\itpm.itpedia\17.0.0+6650\hopex.core\Install\SystemUpdate\itpm.itpedia-HF.mzl' library imported into 'SystemDb'
'C:\ProgramData\MEGA\Hopex Application Server\5000\.shadowFiles\sample.datatypes\15.6.21+6650\hopex.core\Install\SystemUpdate\sample.datatypes-HF.mzl' library imported into 'SystemDb'
'C:\ProgramData\MEGA\Hopex Application Server\5000\.shadowFiles\simulation.engine\17.0.0+6650\hopex.core\Install\SystemUpdate\simulation.engine-HF.mzl' library imported into 'SystemDb'
'C:\ProgramData\MEGA\Hopex Application Server\5000\.shadowFiles\tool.itpm.apmauto\17.0.0+6650\hopex.core\Install\SystemUpdate\tool.itpm.apmauto-HF.mzl' library imported into
'SystemDb'
```

The environment is updated.

9. Click **Close**.
10. Restart the **Hopex Core Back-End** module:
    In the **HAS console**, **Cluster** navigation menu, **Modules** tab: on **Hopex Core Back-End** module, click **Plus** ⋮ **> Start**.

Warnings

Some warning examples you might get at environment upgrade:

- "Upgrade Database Version": "Perform SQL conversion on the repository"
  When migrating, and at each major version change, you must convert SQL Server table formats.
  ☛ *See Converting a Repository.*
- "The indexing format changed"
  You must reindex your repositories to be able to use the full-text search.
  ☛ *See Indexing a repository manually.*
- "Reading/Writing access diagram is not compiled"
  You must compile the reading/writing access diagram
  ☛ *see Compiling the Writing Access Diagram or Compiling the Reading Access Diagram.*

CUSTOMIZING ENVIRONMENTS

Administration tasks may involve you in modifying configuration of an environment so that it meets your particular requirements.

See:

- Backing Up Environment Customizations
- Restoring Environment Customizations
- Compiling an Environment

Backing Up Environment Customizations

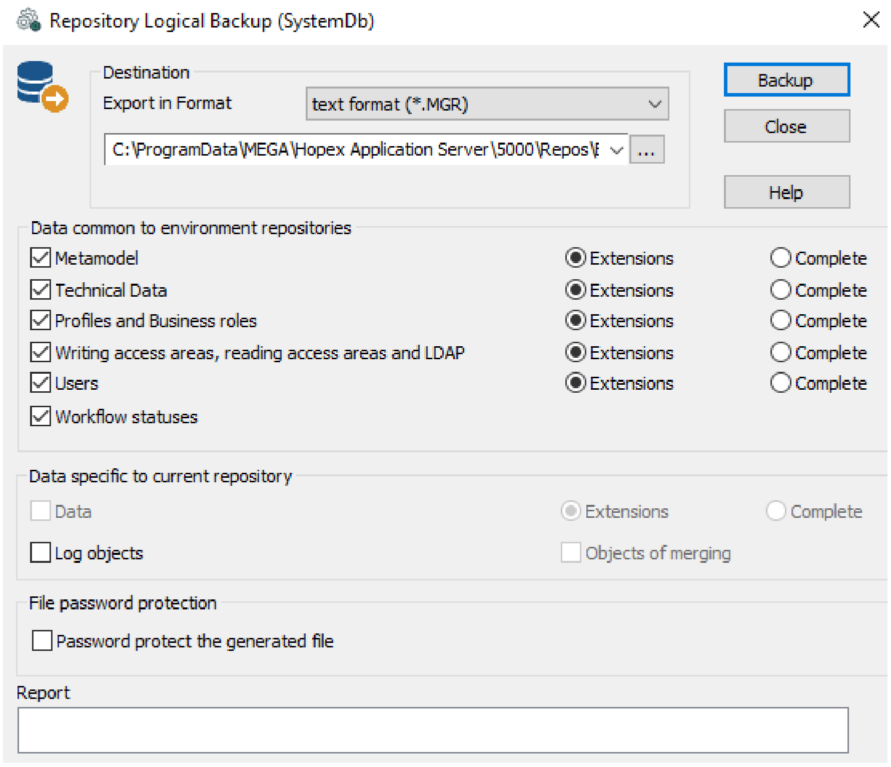
The following operations allow you to back up modifications made to the standard environment so that these can be imported into a new environment. This backup contains modifications you have made to *report templates (MS Word)*, Web site templates, *descriptors*, *queries* etc. It also contains the list of users and their configuration if this exists. You can also save any extensions to the metamodel.

To back up customizations:

1. From **Hopex Administration**, connect to the environment.

   ☛ *See Connecting to an Environment.*

2. In the **Repositories** folder, right-click the system repository (SystemDb) and select **Logical Backup**.
   The **Repository Logical Backup** dialog box opens.


Repository Logical Backup (SystemDb)

Destination
Export in Format: text format (*.MGR)
C:\ProgramData\MEGA\Hopex Application Server\5000\Repos\

Data common to environment repositories

☑ Metamodel	● Extensions	○ Complete
☑ Technical Data	● Extensions	○ Complete
☑ Profiles and Business roles	● Extensions	○ Complete
☑ Writing access areas, reading access areas and LDAP	● Extensions	○ Complete
☑ Users	● Extensions	○ Complete
☑ Workflow statuses		

Data specific to current repository

☐ Data	● Extensions	○ Complete
☐ Log objects	☐ Objects of merging	

File password protection
☐ Password protect the generated file

Report

3. In the **Destination** pane, select the export format of the backup file:
   - **text** (.MGR)
   - **XML MEGA** (.XMG)

     ☛ *This format cannot be used to extract the metamodel or technical data. This format is reserved for exchange of data between **Hopex** and other applications. It includes commands or data (objects and links).*

4. (Optional) In the **Destination** frame, click **Browse** ... to browse the folder tree and modify the name and/or location of the backup file.

   ☛ *By default, the backup is saved in the "SystemDb.mgr" file in the \SysDb\WORK\ folder of the repository.*

5. Select the type of data you want to save.

   ☛ *The **Extensions** button corresponds to data created by users.*

   Carry out a complete backup only if technical support asks you to do so.

   ☺ *It is recommended that you save the extensions of the metamodel and technical data in different files.*

   In the **Data common to environment repositories** pane, select the data type of the system repository to be saved:

   - **Metamodel** if you want to extract the metamodel from the system repository. This is useful if the standard metamodel has been modified.
   - **Technical Data** allows extraction from the system repository of data such as *descriptors*, *queries*, and *report templates (MS Word)*.
   - **Profiles and Business roles** allows extraction of created profiles and business roles (those not provided by MEGA).
   - **Writing access areas and reading access areas** allows extraction of created writing and reading access areas (those not provided by MEGA).
   - **Users** allows extraction of created users (persons, person groups, logins).
   - **Workflow statuses** enables extraction of workflows (workflow instances, transitions and statuses, tasks, validations, requests for change).

   In the **Data specific to current repository** pane:

   - **Log objects** allows you to also save object logs.

   ☛ *For more information on histories, see Deleting a Repository and Viewing Object History.*

6. (If needed) In the **File password protection** pane, select **Password protect the generated file**.

   ☛ *The password is asked at file import.*

7. Click **Backup** to start backup.
   A series of messages keeps you informed of backup progress.

   The file is saved in the selected format (*.mgr or *.xmg) or in *.mgz format if you have selected the file protection option.

Restoring Environment Customizations

You can restore environment customizations that you have backed up.

☛ *See Backing Up Environment Customizations.*

To restore environment customizations:

1. From **Hopex Administration**, connect to the environment.

   ☛ *See Connecting to an Environment.*

2. Under the **Repositories** folder, right-click the repository concerned and select **Object Management > Import**.

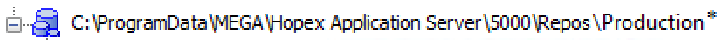
   ☛ *See Updating a Repository.*

Compiling an Environment

The metamodel and technical data must be compiled after migration or customization. This is to check configuration of the environment. When compilation has been completed, processing for all users of this environment is speeded up.

Hopex can operate in "interpreted" (not compiled) mode but with reduced performance.

In the **Hopex Administration** navigation tree, an asterisk after the environment name indicates that the metamodel or/and technical data (excluding permissions) of this environment is/are in "interpreted" (not compiled) mode.


C:\ProgramData\MEGA\Hopex Application Server\5000\Repos\Production*

Metamodel compilation includes in parallel translation in the current language. You can also translate the metamodel into another language.

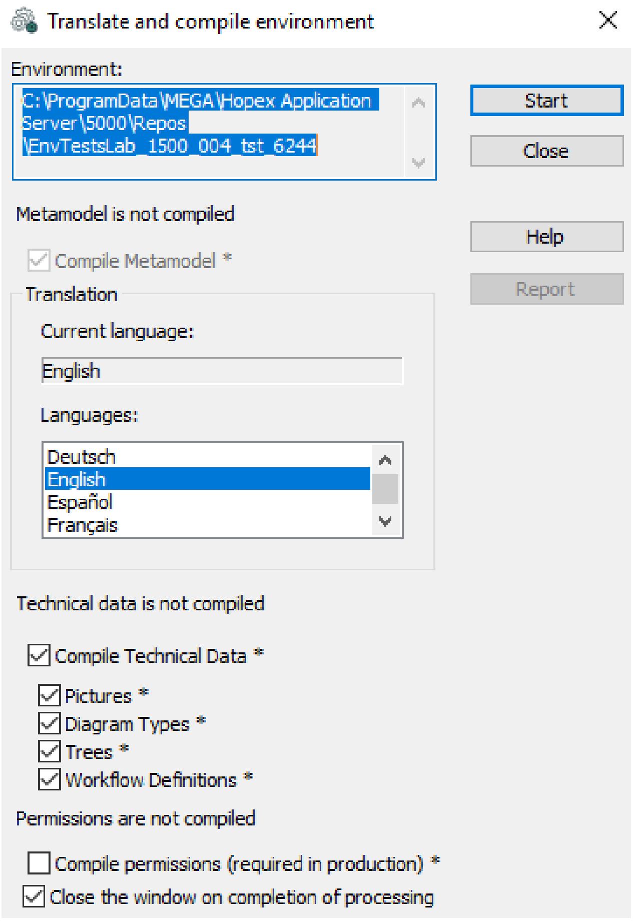
☺ *You can perform the compilation while users are still connected and working.*

To translate and compile the metamodel and/or compile technical data:

1. From **Hopex Administration**, connect to the environment.

   ☛ *See Connecting to an Environment.*

2. In the navigation tree, right-click the environment and select **Metamodel > Translate and Compile**.
   The **Translate and compile environment** dialog box opens.



Translate and compile environment

Environment:
C:\ProgramData\MEGA\Hopex Application Server\5000\Repos\EnvTestsLab_1500_004_tst_6244

Metamodel is not compiled

[✓] Compile Metamodel *

Translation

Current language:
English

Languages:
Deutsch
English
Español
Français

Technical data is not compiled

[✓] Compile Technical Data *

[✓] Pictures *
[✓] Diagram Types *
[✓] Trees *
[✓] Workflow Definitions *

Permissions are not compiled

[ ] Compile permissions (required in production) *
[✓] Close the window on completion of processing

* : the asterisk indicates interpreted (not compiled) mode.
In the **Translation** frame, the **Current Language** field indicates the current language of the system repository.

3. If the metamodel is not compiled, keep **Compile Metamodel*** selected.
4. (If **Compile Metamodel*** is selected) In the **Translation** frame, in the **Languages** list of the system repository, select the target translation language.

   ```
   Example: "English"
   ```

5. If technical data is not compiled, keep **Compile Technical Data*** selected.

   If technical data and metamodel are not both compiled, you must also keep Compile Metamodel* selected.

   By default, all technical data types (pictures, diagram types, trees, workflow definitions) are selected.

6. If you are in a production environment, keep the **Compile Permissions*** option selected; otherwise, you can clear the selection. This compilation improves **Hopex** loading times.

   ☛ *Compilation of permissions can take some time (more than an hour) and is recommended only in a production environment. The fact of not compiling permissions (permissions interpreted mode) has no impact on correct operation of* ***Hopex****.*

7. Keep **Close the window on completion of processing** option selected.

   ☛ *This option enables automatic closing of the* ***Translate and Compile Environment*** *window when compilation is completed, allowing* ***Hopex (Windows Front-End)*** *users to resume their work.*

8. Click **Start** to run compilation and translation.

   ☛ *If* ***Hopex (Windows Front-End)*** *users have remained connected, they are blocked during processing.*

   Metamodel and/or technical data compilation (excluding permissions) takes several minutes.

   If you selected metamodel compilation with a different target language, after execution the system repository is available in the new language.

SCHEDULING (SCHEDULER)

The **Hopex Scheduler** feature enables to create and schedule jobs.

☛ *For detailed information on the scheduler, see the* ***Hopex Customization (Windows) - Using the Scheduler*** *technical article.*

The following points are covered here:

INTRODUCTION TO THE SCHEDULER

Concepts

The Scheduler feature enables to perform tasks defined by MEGA or by an Hopex Administrator at defined dates, times, and frequencies so as to avoid overloading **Hopex** at user working hours.

Job

A job is a process. It includes:

- a macro to be executed
- a context, which gives the information required to execute the macro: **Job Context** as a character string.

Scheduler

The Scheduler enables to schedule job execution:

- execution date and time
- frequency

Trigger

A Trigger is associated with a job to define the job execution date:

- the Trigger is based on a Trigger Definition. This definition consists of a job which includes the macro that the Trigger will execute.
- the Scheduler enables to define when (date and time) to execute the job and at which frequency.

Defining your Local Time (Time Zone)

In the Scheduler, by default the time format is hh:mm:ss (UTC). To facilitate the configuration, you can change this UTC format for a (user or server) local time format.

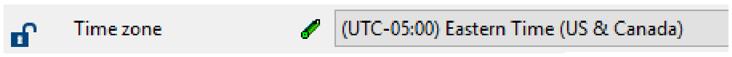
☛ *See Defining the Execution Time Zone.*

To define your user local time:

1. Access the site (or environment) level options.
2. Expand the **Installation** folder and select **Web Application**.

3. In the right pane, use the drop-down menu of the **Time zone** option to select your time zone.

    ```
    E.g.: select "(UTC-05:00) Eastern Time (US & Canada)" to
    configure times in New-York local time.
    ```

    Time zone (UTC-05:00) Eastern Time (US & Canada)

    When you configure your Triggers, their execution scheduling is defined in this time zone if you choose **User time zone** as execution time zone.

    If you configure your Triggers in the **UTC** or **Server time zone**, you can check their conversion in this local time zone if needed.

    ☛ *For example, see Defining the Execution Time.*

Managing Triggers

See:

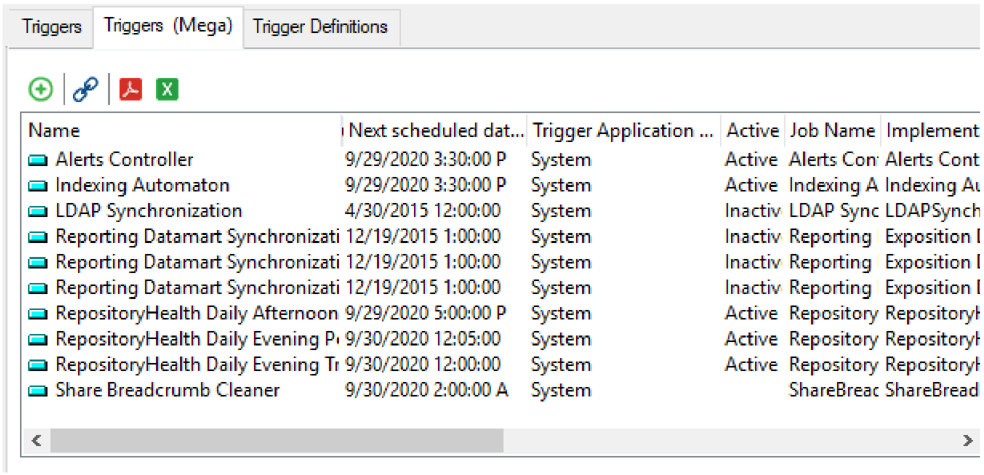
- Accessing Triggers
- Creating a Trigger
- Managing a Trigger
- Defining the Trigger Execution Context

Accessing Triggers

In the **Administration** application (Administration.exe), the Trigger Management window displays the following tabs:

- **Triggers**
  Triggers defined by a Hopex administrator. These Triggers can be defined on a data repository or on the System repository.
- **Triggers (Mega)**
  Triggers provided with Hopex and available in all the installations. These Triggers are defined on the System repository.
- **Trigger Definitions**
  Predefined Triggers available for you to create your Triggers.

```
Example: "Computation of OnDemand and expired OnUpdate
MetaAttributes”
```



Name	Next scheduled dat...	Trigger Application ...	Active	Job Name	Implement
Alerts Controller	9/29/2020 3:30:00 P	System	Active	Alerts Con	Alerts Cont
Indexing Automaton	9/29/2020 3:30:00 P	System	Active	Indexing A	Indexing Au
LDAP Synchronization	4/30/2015 12:00:00	System	Inactiv	LDAP Sync	LDAPSynch
Reporting Datamart Synchronizati	12/19/2015 1:00:00	System	Inactiv	Reporting	Exposition I
Reporting Datamart Synchronizati	12/19/2015 1:00:00	System	Inactiv	Reporting	Exposition I
Reporting Datamart Synchronizati	12/19/2015 1:00:00	System	Inactiv	Reporting	Exposition I
RepositoryHealth Daily Afternoon	9/29/2020 5:00:00 P	System	Active	Repository	Repositoryh
RepositoryHealth Daily Evening P	9/30/2020 12:05:00	System	Active	Repository	Repositoryh
RepositoryHealth Daily Evening Tı	9/30/2020 12:00:00	System	Active	Repository	Repositoryh
Share Breadcrumb Cleaner	9/30/2020 2:00:00 A	System		ShareBreac	ShareBread

For each scheduled Trigger, the list indicates in particular:

- its name
- its next execution date and time
- its status (active or inactive)
- the name of the executed job
- the name of the job implementing macro

To access the Trigger management:

1. From **Hopex Administration**, connect to the environment.

   ☛ *See Connecting to an Environment.*

2. In the **Repositories** folder, expand the repository concerned.
3. Right-click the **Scheduler** folder and select **Manage Triggers**.

Creating a Trigger

A Trigger is based on a Trigger Definition. This definition consists of a job which includes the macro that the Trigger will execute.

Trigger Definitions are available in the **Trigger Definitions** tab.

To create a Trigger:

1. Access the Trigger management window.
2. In the **Triggers** tab, click **New** ➕.

   ☛ *If you create a Trigger from the **Triggers (Mega)** tab, this Trigger is automatically moved to the **Triggers** tab.*

3. Select the **Trigger Definition**.
4. Click **Next**.
   The job definition window opens.
5. Enter the **Name** of the job.
6. In the **Job Context** pane, define the job execution context.

   ☛ *See Defining the Trigger Execution Context.*

7. Click **Finish**.
   The Trigger is created.
   By default, the Trigger is active.
   You can execute the Trigger to test it.

   ☛ *See Managing a Trigger.*

Managing a Trigger

You can:

- update the Trigger scheduling
  To modify the job execution dates, times, and frequencies.
  ☛ *See Configuring the Trigger Scheduling.*
- activate/deactivate a Trigger
  By default, a Trigger is active.
  To temporarily suspend the job execution, you can temporarily deactivate its Trigger.
- execute a Trigger
  To immediately execute the job associated with the Trigger (outside its scheduling).
  `For example, to test a job.`
- delete a Trigger
  If you want to reuse the Trigger later, instead of deleting the Trigger you can deactivate it.
- display the Trigger properties
  - The **Scheduling** tab details the scheduling definition and lists all the next executions of the trigger
  - The **System Job** tab details the job executed by the Trigger (especially the macro and execution context).
  - The **Characteristics** page enables to keep the Trigger in the list after its last execution and to modify its retention period (by default 15 days).

To manage a Trigger:

1. Access the Trigger management.
   ☛ *See Accessing Triggers.*
2. Right-click the Trigger concerned and select:
   - **Update Scheduling**
     ☛ *See Configuring the Trigger Scheduling.*
   - **Activate**/**Deactivate**
   - **Execute**
   - **Delete**
   - **Properties**
     ☛ *See Defining the Trigger Execution Context.*

Defining the Trigger Execution Context

A Trigger is triggered on the objects defined in the associated job macro.

To define on which MetaClass the Trigger applies:

1. Access the Trigger management.
   ☛ *See Accessing Triggers.*

2. Right-click the Trigger concerned and select **Properties**.
3. Select the **System Job** tab.
4. In the **Context** pane define the Trigger execution context, i.e. the objects on which the job applies.

   Attention: do not add any break line in the character string.

CONFIGURING THE TRIGGER SCHEDULING

To configure the Trigger scheduling, you must define:

- its execution time zone for all its scheduling time definitions
  - ☛ *See Defining the Execution Time Zone.*
- the date and time of its first execution
  - ☛ *In a recurrence case, the first execution date is not mandatory.*
  - ☛ *See Defining the First Execution Date of the Trigger.*
- its frequency
  The execution can be unique or recurrent.
  - ☛ *See Defining the Trigger Frequency.*

  If the execution is recurrent:
  - its execution time.
    - ☛ *See Defining the Trigger execution time.*
  - if needed, you can define a recurrence on the execution time, i.e. execute the Trigger several times the scheduled day.
    - ☛ *See Defining a time-based recurrence on the Trigger execution.*
  - the date of its last execution (defined or with no end)
    - ☛ *See Defining the Last Execution Date.*

Defining the Execution Time Zone

To facilitate scheduling time definition, you can modify the time zone in which you define the scheduling times:

- **UTC** (default), to define times in UTC format
- **User time zone** to define times in the user time zone
- **Server time zone** (STZ), to define times in the time zone of the server executing the Trigger

Attention: if you change the time zone a posteriori, times are not automatically adapted.

To define the execution time zone:

1. Access the Triggers.
   - ☛ *See Accessing Triggers.*
2. Right-click the Trigger concerned and select **Update Scheduling**.
3. In the **Time zone for all the scheduling time definitions**, select the time zone.
4. If you select **User time zone**, you must define your time zone.
   - ☛ *Voir Defining your Local Time (Time Zone).*

Defining the First Execution Date of the Trigger

You must define the first execution date of the Trigger. It can be:

- absolute

  `E.g.: on the 04/18/2020 at 18:30:15.`
- relative (relative to a reference date)

  ☛ *In a recurrence case, the first execution date is not mandatory.*

  ☛ *In a non recurrence case, the first execution date is the unique execution date.*

Defining the first execution date (or unique execution)

To define the first execution date of a Trigger:

1. Access the Triggers.

   ☛ *See Accessing Triggers.*
2. Right-click the Trigger concerned and select **Update Scheduling**.
3. In the **Start** section:
   - Use the calendar of the **Start date (absolute)** field to select the first execution date of the Trigger.

     Select **Today**, if you want to define the current day.
   - In the **Start time** field, set the triggering time of the Trigger.

     By default the time is set to 00:00:00 (hh:mm:ss format) in the time zone defined (see Defining the Execution Time Zone).

     ☛ *If you are in the UTC time zone, to facilitate the check of your settings, see Defining your Local Time (Time Zone).*

Defining a relative date for the first execution

You can define a relative date for the first execution, i.e. define the first execution date as:

- (by default) immediately after the Trigger creation, or
- at a later date:
  - a specific number of days after the reference date
  - a specific day of the week after the reference date
  - a specific day of the moth after the reference date

To define the first execution date of a Trigger:

1. Access the Triggers.

   ☛ *See Accessing Triggers.*
2. Right-click the Trigger concerned and select **Update Scheduling**.
3. In the **Start** section, select **Relative Date**.
   By default **Reference date/time as soon as possible** is selected: the execution is triggered right after the Trigger creation.

4. To configure a later relative date, clear **Reference date/time as soon as possible**, then in the **Start date (relative)** click ⋯ and define:
   The **Day of relative date**:
   - In the **Days from reference**, enter the number of days after the reference date, or
   - Select the **Day of week** and use the drop-down list to select the chosen day, or

     ```
     E.g.: Tuesday, the Trigger is executed on the first Tuesday
     following the reference date.
     ```
   - Select the **Day of month** and use the drop-down list to select the day.

     ```
     E.g.: 15th, the Trigger is executed on the 15th of the month
     following the reference date.
     ```

   The **Month of relative date**:
   - In the **Months from reference**, enter the number of months after the reference date, or

     ```
     E.g.: 2, the Trigger is executed a couple of months after
     the reference date.
     ```
   - Select the **Month of year** and use the drop-down list to select the chosen month, or

     ```
     E.g.: June, the Trigger is executed in June following the
     reference date.
     ```

Defining the Trigger Frequency

A Trigger can be executed uniquely or on a regular basis.

Whatever the frequency chosen, you can perform a first execution as defined in the **Start** section.

Frequency:

- ***daily***
  By default, the Trigger is executed every day at the time set for the first execution.
  You can execute the Trigger every N days (N to be defined)
- ***weekly***, you must define:
  - the day of the week

    `E.g.: Monday, Tuesday,..., Sunday`

  You can select several days.
  - the frequency

    `E.g.: every two weeks (N=2)`
- ***monthly***, you must define:
  - the day of the month, or the day of the week (day of the week and week of the month to be defined)

    `E.g.: 1,2,...,31, last day of the month`

  You can select several days.

    `E.g.: every Sunday of the last week of the month, i.e. the last Saturday of the month.`

  You can select several days and several weeks.
  - the frequency: every N months (N to be defined) or a specific month every year

    `E.g.: every 2 months (N=2) or in April every year.`

  You can select several months.

To define the Trigger execution frequency:

1. Access the Triggers.

   ☛ *See Accessing Triggers.*
2. Right-click the Trigger concerned and select **Update Scheduling**.
3. In the **Date Recurrence** section, use the drop-down menu of the **Recurrence Type** field to select the frequency.

   `Example: Daily, Monthly, Once, Weekly.`
4. Configure the frequency.
5. (If you want to first execute the Trigger as defined in the **Start** section) Select **Execute at Start date time**.

Defining the Last Execution Date

By default, the Trigger scheduling is endless.

You can define the Trigger last execution date, via:

- an end date, or
- a defined repeat number

To define the Trigger last execution date:

1. Access the Triggers.

   ☛ *See Accessing Triggers.*

2. Right-click the Trigger concerned and select **Update Scheduling**.
3. In the **Recurrence End** section, use the drop-down menu of the **Recurrence End Type** to select the end type.

   `Example: End Date or Repeat Number.`
4. (If you selected End Date) Define the last execution day and time.
   - Use the calendar of the **End date (absolute)** field to select the last execution date of the Trigger.
   - In the **End time** field, set the triggering time of the Trigger.

     `By default the time is set to 00:00:00 (hh:mm:ss format) in the time zone define (see Defining the Execution Time Zone).`

     ☛ *If you are in the UTC time zone, to facilitate the check of your settings, see Defining your Local Time (Time Zone).*

Defining the Execution Time

In case the **Recurrence Type** is "Daily", "Weekly", or "Monthly", you must define the Trigger execution time:

- once: you need to define the execution time only
- several times a day, you must define:
  - the scheduling period (in hours):
  - the start time
  - the end time

Times are defined in the defined time zone (see Defining the Execution Time Zone) in hh:mm:ss format.

☛ *If you are in the UTC time zone, to facilitate the check of your settings, see Defining your Local Time (Time Zone).*

Defining the Trigger execution time

You can define a unique execution time each scheduled execution day.

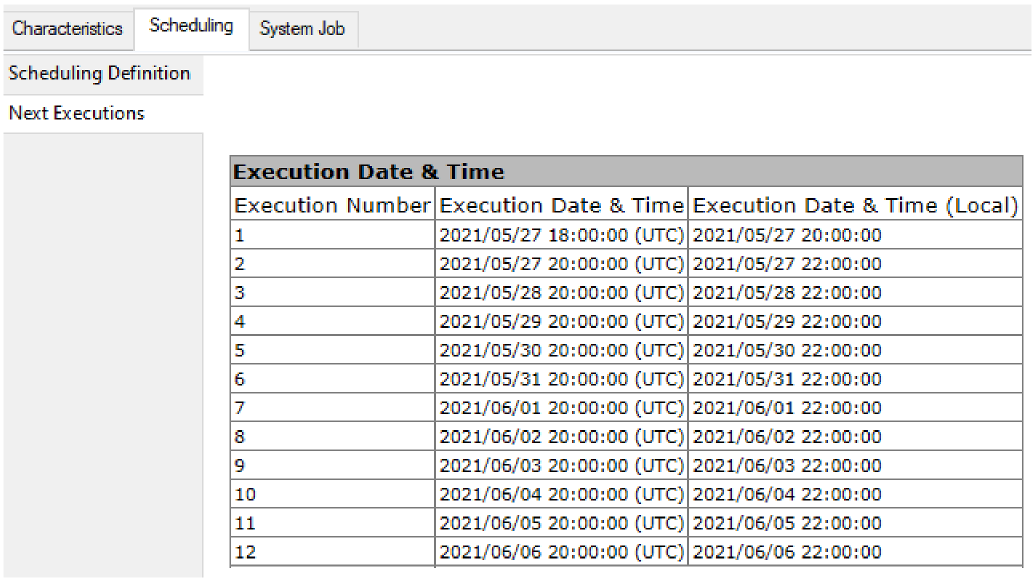
To set the Trigger execution time:

1. Access the Triggers.

   ☛ *See Accessing Triggers.*
2. Right-click the Trigger concerned and select **Update Scheduling**.
3. In the **Time scheduling (for date recurrence)** section:
   - in the **Scheduling type**, keep "Once".
   - in the **Single trigger time** set the time at which you want to execute the Trigger.

     `E.g.: 22:00:00, by default 04:00:00 (in the time zone defined).`
4. Click **OK**.

5. Check the scheduling of your execution time.
   - Close and re-open the Trigger Management window.
   - Right-click the Trigger and select **Properties**.
   - In the **Scheduling** tab, select the **Next Executions** sub-tab.



Execution Date & Time		
Execution Number	Execution Date & Time	Execution Date & Time (Local)
1	2021/05/27 18:00:00 (UTC)	2021/05/27 20:00:00
2	2021/05/27 20:00:00 (UTC)	2021/05/27 22:00:00
3	2021/05/28 20:00:00 (UTC)	2021/05/28 22:00:00
4	2021/05/29 20:00:00 (UTC)	2021/05/29 22:00:00
5	2021/05/30 20:00:00 (UTC)	2021/05/30 22:00:00
6	2021/05/31 20:00:00 (UTC)	2021/05/31 22:00:00
7	2021/06/01 20:00:00 (UTC)	2021/06/01 22:00:00
8	2021/06/02 20:00:00 (UTC)	2021/06/02 22:00:00
9	2021/06/03 20:00:00 (UTC)	2021/06/03 22:00:00
10	2021/06/04 20:00:00 (UTC)	2021/06/04 22:00:00
11	2021/06/05 20:00:00 (UTC)	2021/06/05 22:00:00
12	2021/06/06 20:00:00 (UTC)	2021/06/06 22:00:00

The **Execution Date & Time** table indicates the first execution date and time and the following ones with their corresponding local time.

☛ *To define your local time, see Defining your Local Time (Time Zone).*

```
E.g.: in this daily scheduling, after the first execution
date (here 20:00:00 Paris local time i.e. 18:00:00 UTC), the
Trigger is executed once a day at 22:00:00 Paris local time
i.e. 18:00:00:00 (UTC).
```

Defining a time-based recurrence on the Trigger execution

You can schedule the Trigger execution several times each scheduled execution day. You then must define:

- the scheduling period (in hours):
  Default period: every 4 hours (04:00:00) each scheduled day.
- the start time of the time-based scheduling
- the end time of the time-based scheduling

To define a time-based recurrence on the Trigger execution:

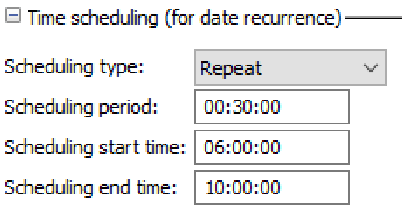
1. Access the Triggers.

   ☛ *See Accessing Triggers.*

2. Right-click the Trigger concerned and select **Update Scheduling**.
3. In the **Time scheduling (for date recurrence)** section, use the drop-down menu of **Scheduling type** field to select "recurrent".

4. Define the recurrence (period, start time and end time of the time-based scheduling):
   - **Scheduling period** (hh:mm:ss format)
   - **Scheduling start time** (hh:mm:ss format)
   - **Scheduling end time** (hh:mm:ss format)

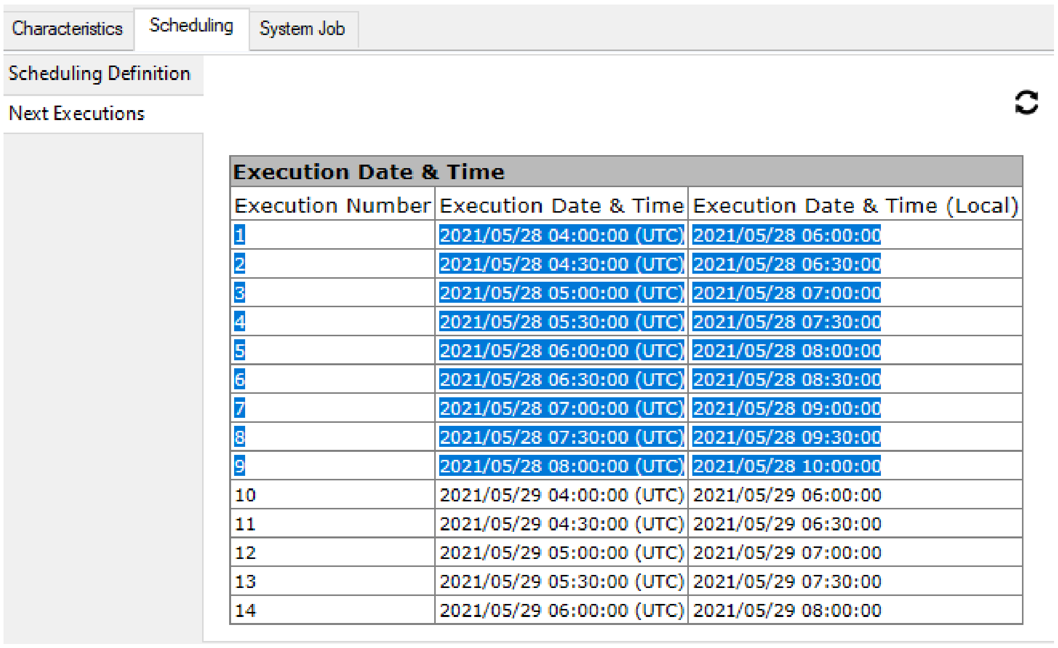
   ```
   E.g.: schedule the Trigger every 30 minutes from 6am to 10am
   in the defined time zone.
   ```



⊟ Time scheduling (for date recurrence)

Scheduling type:	Repeat
Scheduling period:	00:30:00
Scheduling start time:	06:00:00
Scheduling end time:	10:00:00

5. Click **OK**.
6. Check your scheduling configuration regarding the execution time recurrence:
   - Close and re-open the Trigger Management window.
   - Right-click the Trigger and select **Properties**.
   - In the **Scheduling** tab, select the **Next Executions** sub-tab.



Execution Date & Time		
Execution Number	Execution Date & Time	Execution Date & Time (Local)
1	2021/05/28 04:00:00 (UTC)	2021/05/28 06:00:00
2	2021/05/28 04:30:00 (UTC)	2021/05/28 06:30:00
3	2021/05/28 05:00:00 (UTC)	2021/05/28 07:00:00
4	2021/05/28 05:30:00 (UTC)	2021/05/28 07:30:00
5	2021/05/28 06:00:00 (UTC)	2021/05/28 08:00:00
6	2021/05/28 06:30:00 (UTC)	2021/05/28 08:30:00
7	2021/05/28 07:00:00 (UTC)	2021/05/28 09:00:00
8	2021/05/28 07:30:00 (UTC)	2021/05/28 09:30:00
9	2021/05/28 08:00:00 (UTC)	2021/05/28 10:00:00
10	2021/05/29 04:00:00 (UTC)	2021/05/29 06:00:00
11	2021/05/29 04:30:00 (UTC)	2021/05/29 06:30:00
12	2021/05/29 05:00:00 (UTC)	2021/05/29 07:00:00
13	2021/05/29 05:30:00 (UTC)	2021/05/29 07:30:00
14	2021/05/29 06:00:00 (UTC)	2021/05/29 08:00:00

The **Execution Date & Time** table indicates the first execution date and time and the following ones with their corresponding local time.

☛ *To set your local time, see Defining your Local Time (Time Zone).*

```
This example shows a daily scheduling, the Trigger is
executed every day from 06:00:00 to 10:00:00 (Paris local
time) i.e. from 04:00:00 to 08:00:00 (UTC).
```

5

SUPERVISION AND EVENTS

Hopex provides a supervision tool (**Hopex Server Supervisor**) that enables management of events.

The following points are covered here:

To access the list and description of the supervision events, see **Hopex Administration > Supervision Event Description** Technical Article.

Introduction to Supervision

The **Hopex Server Supervisor** supervision tool is used to collect messages from **Hopex** applications (**Windows Front-End** and **Web Front-End**). These messages include indicators to ensure that **Hopex** is operating correctly.

`Example: information or error messages.`

A message corresponds to a supervision event, which has been previously coded in the executable. A client cannot create a new event.

The following points are covered here:

- Supervision Event
- Supervision Files
- Supervision Configuration

Supervision Event

A supervision event can include about fifty pieces of information.

```
Example: "Event infos" information is a json (or a text)
that can provide details regarding the event context. The
json structure depends on each event.
```

To consult and analyze a supervision event, see Consulting a Supervision Event File.

Event types

Events are sorted by type

- A (Action): user action
- W (Warning): alert
- E (Error): error
- S (Snapshot): process snapshot

A type A event is characterized by:

- a start, which corresponds to its creation.
- an end, which corresponds to the moment the message is actually sent. Only one message is sent at the end. It summarizes the indicators of the process for the event duration.

Type W and type E events are immediate.

Type S events summarize the indicators of the process since the last sent snapshot.

Supervision Files

Supervision files include supervision events.

Each supervision file represents a day.

Supervision file name format: sspsprvs<YYYYmmDD>.txt.

```
Example: SSPSPRVS20210719.TXT
(supervision file for the 19th of July 2021)
```

To find the supervision file location in **Hopex Server Supervisor**, see Finding the Supervision File Location.

Supervision Configuration

Configuration of some of the supervision behaviors is performed in the **HAS console**: **Modules > Module Settings > MegasiteSettings** in the [Supervision] section.

```
[Supervision]
StateInterval=<time interval>
Filter=<Filter>
```

Megasite-Settings	**Description**
[Supervision]	Supervision parameter section
StateInterval	Time interval (in millisecond) between two supervision events of snap-shot type. Minimum value: 1000 By default this parameter value corresponds to 3' (180000) (do not modify this parameter)
Filter	Enables definition of executables to be supervised. Other executable supervision is deactivated. If not specified, there is no filtering and all of **Hopex** executables are supervised. Example: `Filter=AM` Displays only **Hopex Windows Front-End** (code A) and its Administration (code M) events. See Executable code. To modify the filter, see Modifying the Processes to Supervise

Executable code

Each application is associated with a code:

Windows Front-End:

- A: Administration
  (mgwmapp.exe /DesktopAppGbm.Administration)
- M: Hopex
  (mgwmapp.exe)
- N: Automation
  (API mgwmapp.exe/Automation)

Web Front-End:

- R: Session holder
  (HAS.Hopex.BackEnd.exe)
- O: Environment holder
  (HAS.Hopex.BackEnd.exe)
- J: Job holder
  (HAS.Hopex.BackEnd.exe)

SUPERVISION TOOL: HOPEX SERVER SUPERVISOR

The **Hopex Server Supervisor** tool:

- enables reading supervision files.
- gives access to calculated/filtered views of supervision events.

You can activate/deactivate the supervision of certain processes to filter messages. Only selected executables send messages.

> ☛ *See Supervision Configuration.*

See:

- Starting Hopex Server Supervisor
- Extending Hopex Server Supervisor Functionalities
- Modifying the Processes to Supervise
- Finding the Supervision File Location
- Modifying the Supervision File Location

Starting Hopex Server Supervisor

To start the **Hopex Server Supervisor** tool:

1. Access the HAS instance repository.
   Default path: C:\ProgramData\MEGA\Hopex Application Server\ <name of the HAS instance>

   ```
   E.g.: C:\ProgramData\MEGA\Hopex Application Server\5000
   ```

2. Right-click **Supervisor** and select **Execute as administrator**.

   The **Hopex Server Supervisor** icon appears in the system tray of your workstation.

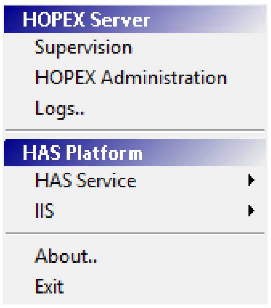
   > ☛ *The green button on the icon indicates that the HAS instance is started and its health status is ok, else the button is red: .*

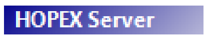
Extending Hopex Server Supervisor Functionalities

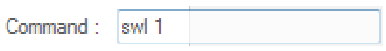
By default, at **Hopex** installation, **Hopex Server Supervisor** menus are minimum.

To extend access to **Hopex Server Supervisor** functionalities:

1. In your workstation system tray, right-click **Hopex Server Supervisor** .

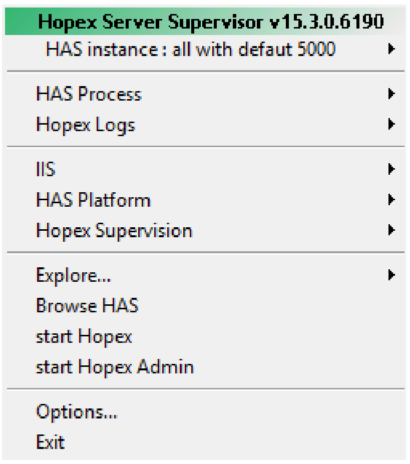


2. Click  .
   The **Command** field is displayed.
3. Enter "swl 1" and press "Enter".



All of **Hopex Server Supervisor** functionalities are available.

☛ *To return to a minimum display, perform step 2, enter 'swm" and press "Enter".*

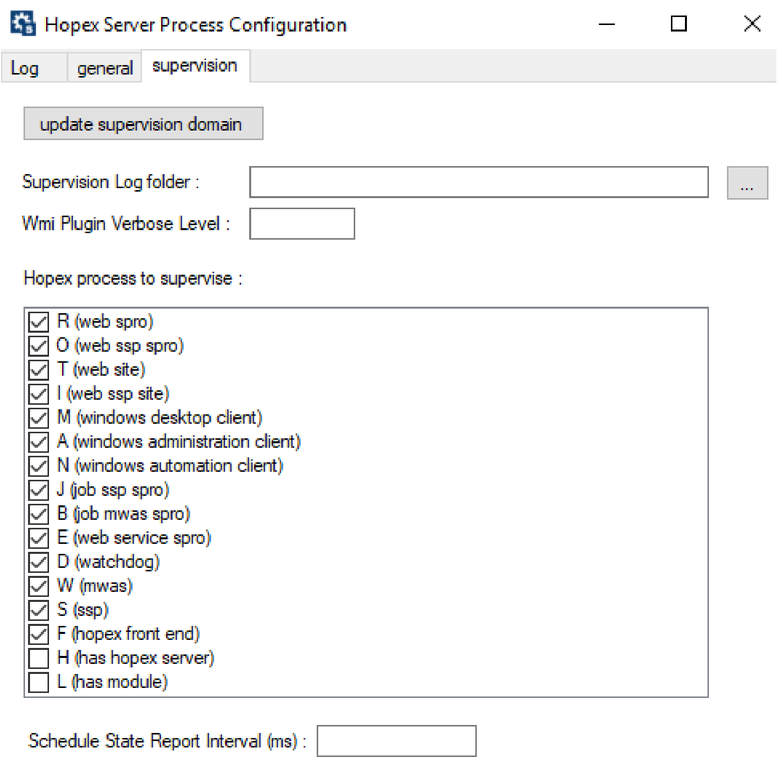


Modifying the Processes to Supervise

Hopex Server Supervisor enables to configure the processes to supervise.

To configure the processes to supervise:

1. In your workstation system tray, right-click **Hopex Server Supervisor** and select **Hopex Supervision > Supervision configuration**.
2. From **Hopex Server Process Configuration** window, **Supervision** tab, select the processes you want to supervise.



Hopex Server Process Configuration

Log | general | supervision

update supervision domain

Supervision Log folder :

Wmi Plugin Verbose Level :

Hopex process to supervise :

- [x] R (web spro)
- [x] O (web ssp spro)
- [x] T (web site)
- [x] I (web ssp site)
- [x] M (windows desktop client)
- [x] A (windows administration client)
- [x] N (windows automation client)
- [x] J (job ssp spro)
- [x] B (job mwas spro)
- [x] E (web service spro)
- [x] D (watchdog)
- [x] W (mwas)
- [x] S (ssp)
- [x] F (hopex front end)
- [ ] H (has hopex server)
- [ ] L (has module)

Schedule State Report Interval (ms) :

Finding the Supervision File Location

To open the folder where the supervision files are stored:

1. Open **Hopex Server Supervisor** in extended configuration.

   ☛ *See Extending Hopex Server Supervisor Functionalities.*

2. From **Hopex Server Supervisor**, select **Hopex Logs > Daily Logs Manager**.
3. Right-click the daily log row (sspsprvs<yyyymmdd>.txt) and select **Open Folder**.

   ☛ *To modify the supervision file location, see Modifying the Supervision File Location.*

Modifying the Supervision File Location

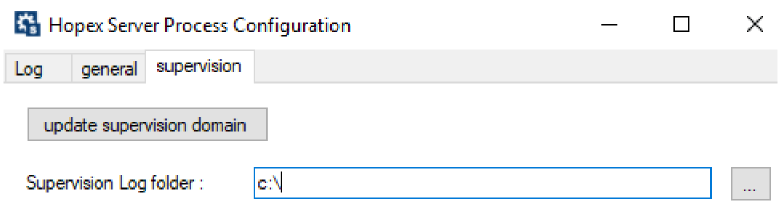
To modify the supervision file location:

1. Create the directory where you want the supervision files to be stored.

   `Example: c:\log`

2. From **Hopex Server Supervisor**, select **Hopex Supervision > Supervision configuration**.
3. Select the **Supervision** tab.
4. In the **Supervision Log folder** field, use the browse button **...** to define the directory path where the supervision files are stored.

   `Example: c:\log`



Supervision Log folder : c:\

These modifications are immediately taken into account. Supervision files are stored in the specified directory (e.g.: c:\log).

Supervising Events

Event supervision is performed from the **Supervision** tool of **Hopex Server Supervisor**.

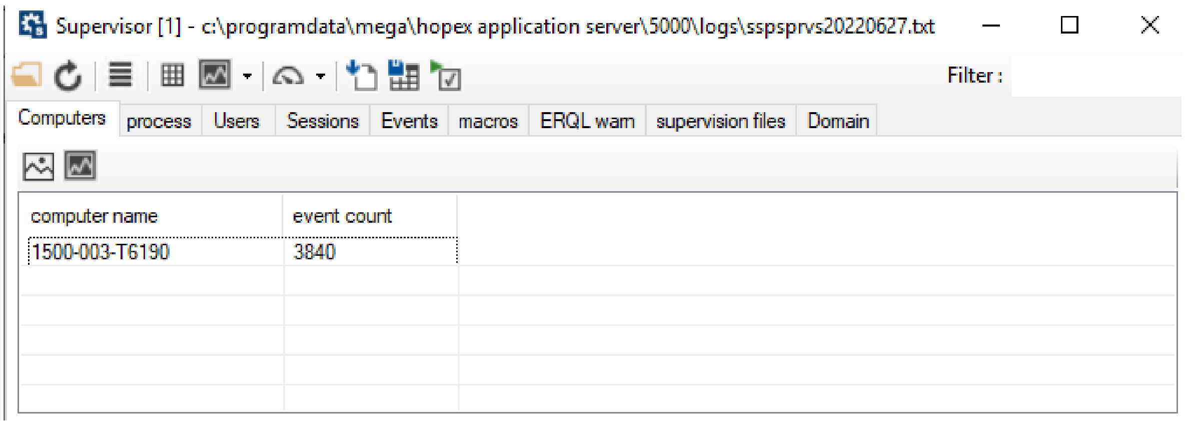
See:

- Supervision Tool
- Consulting a Supervision Event File
- Actions from an Event Supervision Window

Supervision Tool

The **Supervision** tool of **Hopex Server Supervisor** enables to open and analyze **Hopex** event files.

☛ *To launch the **Supervision** tool, see Consulting a Supervision Event File.*


Supervisor [1] - c:\programdata\mega\hopex application server\5000\logs\sspsprvs20220627.txt
Filter :
Computers
process
Users
Sessions
Events
macros
ERQL warn
supervision files
Domain

computer name	event count
1500-003-T6190	3840

For detailed information on event files, see:

- Supervision Event
- Supervision Files.

Supervision tool toolbar

From the **Supervision** tool toolbar, click:

- to open one or several specific supervision files
- to refresh calculated view data of the current supervision file
- to open the set of snapshots that have been created on the different servers.
  A consolidated snapshot gives an application calculated view over the last three minutes.
- to view the load state, object consumption on the set of supervised processes.
- to load a supervision domain (reference domain - Standard) so that comparisons can be made.
- to export data in CSV format.
- to check for incidents.

Supervision tool tabs

In the **Supervision** tool, events are grouped by calculated view that gives access to the corresponding list of prefiltered events. Views correspond to the following tabs:

- **Computers**
  This tab shows the list of servers used and its associated event number.
- **process**
  This tab shows the set of supervised processes of the set of servers.
- **Users**
  This tab shows the list of users who logged on the application.
- **Sessions**
  This tab shows the list of current or past sessions of all the servers or workstations supervised.
- **Events**
  This tab shows the list of Hopex events that have been recorded.
- **macros**
  This tab enables to view macros, for which execution time is high.
- **ERQL warn**
  This tab shows the list of ERQLWarning events that have been recorded.
- **supervision file**
  This tab shows analyzed supervision files.
- **Domain**
  This tab enables to show activity synthesis according to the supervision domain

Consulting a Supervision Event File

To consult an event of a supervision file:

1. From your workstation system tray, right-click **Hopex Server Supervisor** and select **Supervision**.

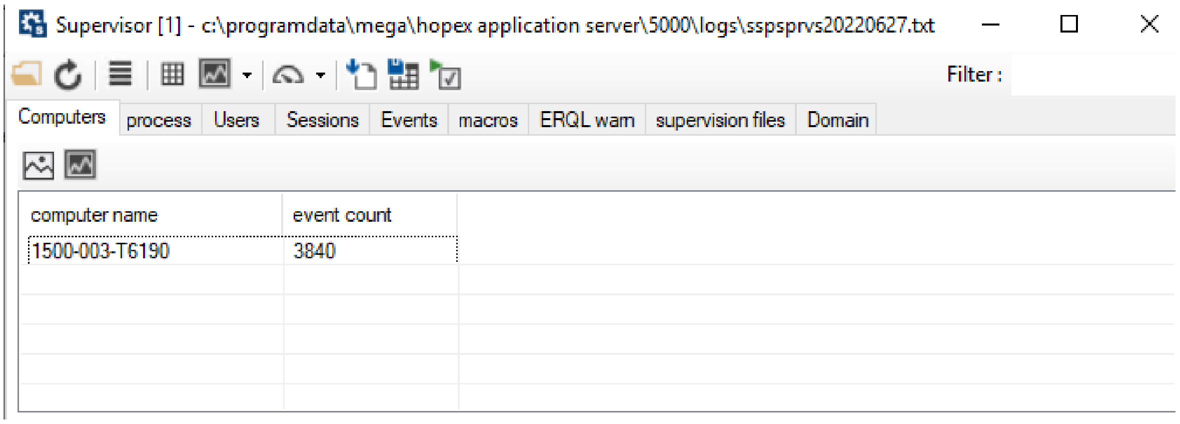
   ☛ *If **Hopex Server Supervisor** is in extended configuration (see Extending Hopex Server Supervisor Functionalities), select **Hopex Supervision > Supervision**.*

   The current supervision file opens.

   ☛ *To open another (or several) supervision file , if **Hopex Server Supervisor** is in extended configuration select **Hopex Supervision > Supervision from file** and select the files.*

   *Alternatively, from **Supervision chart** toolbar, click **Open Supervision data file** and select the files.*

   See Finding the Supervision File Location.


Supervisor [1] - c:\programdata\mega\hopex application server\5000\logs\sspsprvs20220627.txt
Filter :
Computers
process
Users
Sessions
Events
macros
ERQL warn
supervision files
Domain

computer name	event count
1500-003-T6190	3840

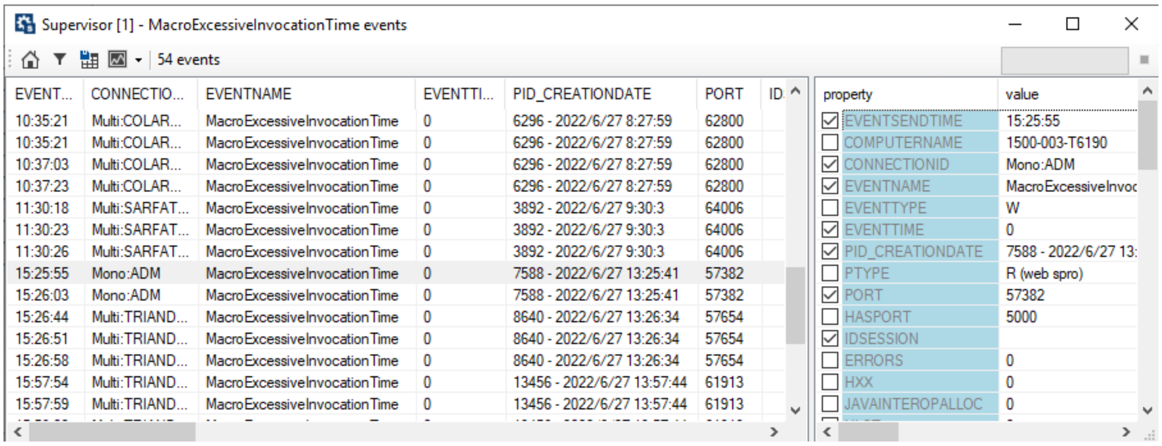
2. (optional) If you are on the current supervision file, click **Refresh** anytime you want to immediately refresh the view data.

   ☛ *See Supervision tool tabs.*

3. Click the tab regarding the view you want to consult:

   ☛ *See Supervision tool tabs.*

4. Double-click the line of the view you want to consult the events.
   Events regarding this view are displayed in table format.



Supervisor [1] - MacroExcessiveInvocationTime events

54 events

EVENT...	CONNECTIO...	EVENTNAME	EVENTTI...	PID_CREATIONDATE	PORT	ID
10:35:21	Multi:COLAR...	MacroExcessiveInvocationTime	0	6296 - 2022/6/27 8:27:59	62800	
10:35:21	Multi:COLAR...	MacroExcessiveInvocationTime	0	6296 - 2022/6/27 8:27:59	62800	
10:37:03	Multi:COLAR...	MacroExcessiveInvocationTime	0	6296 - 2022/6/27 8:27:59	62800	
10:37:23	Multi:COLAR...	MacroExcessiveInvocationTime	0	6296 - 2022/6/27 8:27:59	62800	
11:30:18	Multi:SARFAT...	MacroExcessiveInvocationTime	0	3892 - 2022/6/27 9:30:3	64006	
11:30:23	Multi:SARFAT...	MacroExcessiveInvocationTime	0	3892 - 2022/6/27 9:30:3	64006	
11:30:26	Multi:SARFAT...	MacroExcessiveInvocationTime	0	3892 - 2022/6/27 9:30:3	64006	
15:25:55	Mono:ADM	MacroExcessiveInvocationTime	0	7588 - 2022/6/27 13:25:41	57382	
15:26:03	Mono:ADM	MacroExcessiveInvocationTime	0	7588 - 2022/6/27 13:25:41	57382	
15:26:44	Multi:TRIAND...	MacroExcessiveInvocationTime	0	8640 - 2022/6/27 13:26:34	57654	
15:26:51	Multi:TRIAND...	MacroExcessiveInvocationTime	0	8640 - 2022/6/27 13:26:34	57654	
15:26:58	Multi:TRIAND...	MacroExcessiveInvocationTime	0	8640 - 2022/6/27 13:26:34	57654	
15:57:54	Multi:TRIAND...	MacroExcessiveInvocationTime	0	13456 - 2022/6/27 13:57:44	61913	
15:57:59	Multi:TRIAND...	MacroExcessiveInvocationTime	0	13456 - 2022/6/27 13:57:44	61913	

	property	value
☑	EVENTSENDTIME	15:25:55
☐	COMPUTERNAME	1500-003-T6190
☑	CONNECTIONID	Mono:ADM
☑	EVENTNAME	MacroExcessiveInvoc
☐	EVENTTYPE	W
☑	EVENTTIME	0
☑	PID_CREATIONDATE	7588 - 2022/6/27 13:
☐	PTYPE	R (web spro)
☑	PORT	57382
☐	HASPORT	5000
☑	IDSESSION	
☐	ERRORS	0
☐	HXX	0
☐	JAVAINTEROPALLOC	0

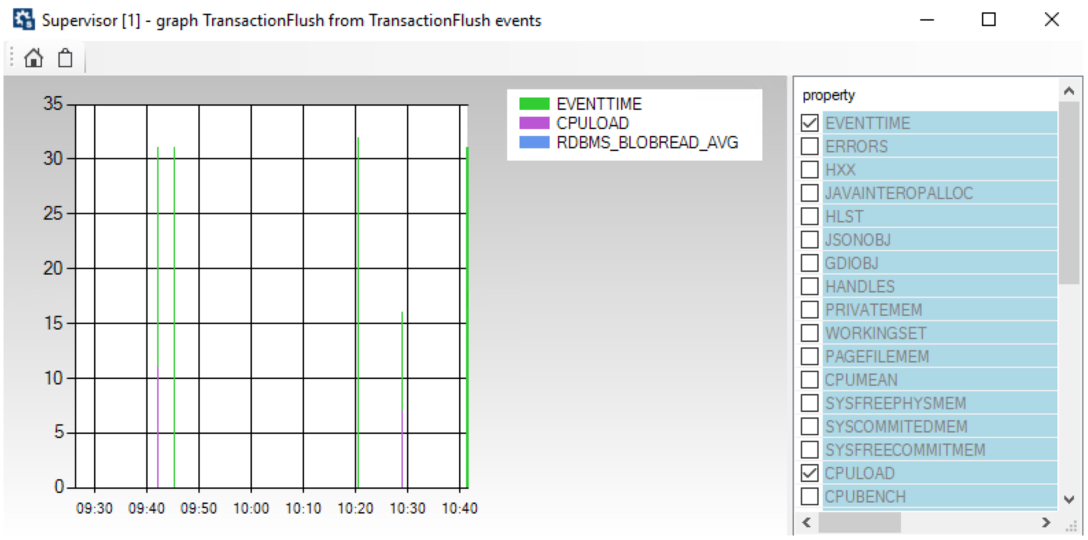
Each line represents an event of which characteristics are detailed.

☛ *See Actions from an Event Supervision Window.*

5. (optional) In the **property** pane, select the indicators (columns) you want to be displayed in a table column.
6. (optional) Right-click the event for which you want to display a graph and select **show graph for <event name>**.

☛ *You cannot get a graph for an Error type or Warning type event.*

The graph is displayed.



7. In the **property** pane, select the indicators you want to be displayed in the graph.
   The graph is calculated on the set of the prefiltered original view, taking into account the event selected.

8. (optional) Click **copy graph image to clipboard** to copy the graph image in the clipboard.

Actions from an Event Supervision Window

From the supervision window, which lists the events in table format, you can:

- access the window of the source **Supervision chart** tool.

  Click **Supervisor Home** .
- create filters to filter displayed events.

  Click **Filters** .
- export current events in .txt format.

  ☛ *Example: to import events in Excel.*

  Click **Export the current filter view as supervision format** .
- display the graph of event count by time interval defined.
  Click **Graph event count by time** .
- interrupt data downloading.

  ☛ *Example: when experiencing long downloading times.*

  Click **Interrupt loading** .

OBJECTS

The following points are covered here:

To work with certain Hopex Solutions, you might need to import modules into Hopex, see **Modules > Importing a Module into Hopex** documentation.

EXPORTING HOPEX OBJECTS

Export of an object with propagation enables creation of a consistent set allowing transfer of part of the repository to another repository. For example, export of **Hopex** objects from a library includes objects present in the library and their dependent objects.

The following points are detailed here:

Export

You can export common modeling objects as well as configuration and parameterization objects.

```
Examples: report templates (MS Word), queries, metamodel
extensions, users.
```

To access these objects, you must select the extended metamodel options in your configuration. This option is available in user options, from the **Repository** icon.

☛ *See Options.*

Export uses the propagation mechanism, which can be configured using perimeters.

☛ *For detailed information on propagation mechanism, see the **Hopex Power Studio - Perimeters** technical article.*

Propagation steps in organizational process (major) export:

1. For an organizational process (major) you also export its operations (minor).
2. For an operation you export the event messages or result messages (minor) of the operation (major)
3. Propagation continues step by step until all links have been explored.

☛ *Export takes account of link types: for certain link types, search in depth of other links stops.*

All of the tags are also exported.

Exporting Objects

You can export **Hopex** objects from:

- **Hopex Administration**
- **Hopex**

☛ *To export objects from the **Administration** desktop (Web Front-End) see the **Hopex Administration Web** guide.*

You can export objects in the following formats:

- **plain text**
  The exported file is in the form of an .MGR file.

  ☛ *For more details on .MGR file syntax, see Command File Syntax.*
- **XML Bizzdesign**
  The exported file is in the form of an .XMG file containing commands or data (objects and links).

  ☛ *For more details on MEGA XML data exchange format, see technical article "MEGA Data Exchange XML Format 70".*
- **Excel**

  ☛ *See the **Hopex Common Features** guide, "Exchanging Data With Excel" chapter.*

You can password-protect the export file generated. The exported file has the .mgz format and can only be imported by entering the password you defined.

See:

- Exporting Hopex objects
- Exporting a Hopex object from the object

Exporting Hopex objects

You can export objects from **Hopex Administration** or from your **Hopex** desktop.

To export **Hopex** objects:

1. Access the repository from which you want to export objects.

   ☛ *See Accessing Repositories.*
2. Right-click the repository concerned and select **Object Management > Export Objects**.

   ☛ *Alternatively, from your **Hopex** desktop, select **File > Export > MEGA objects**.*

   The **Export Hopex Objects** dialog box opens.
3. In the **Export in Format** field, select the export file format.
   Several options enable object export configuration.

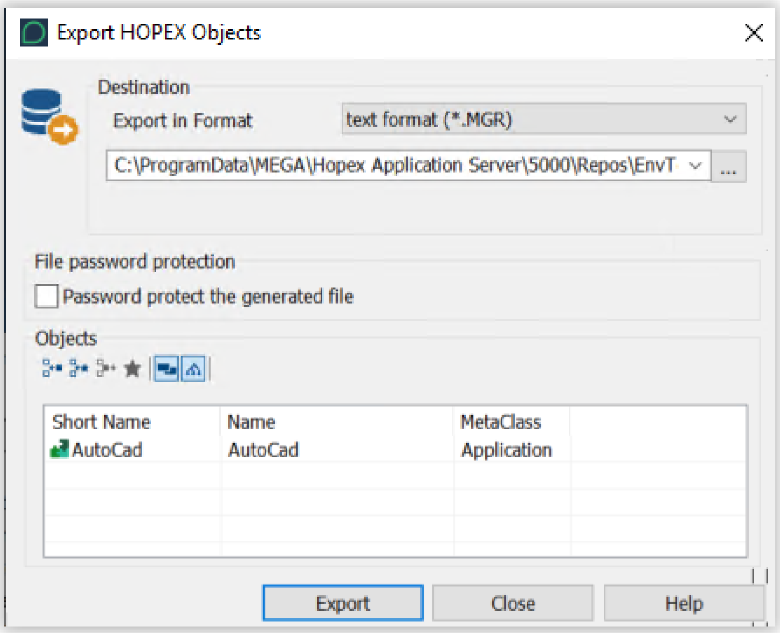
   ☛ *See Options.*
4. (Optional) Enter a different name and folder if the default values are not suitable.

   ☛ ***Browse*** ... *enables you to browse the folder tree and select where the export file will be located.*
5. (If needed) In the **File password protection** pane, select "Password protect the generated file".
6. In the **Objects** frame, click **Add Objects to List** .

   ☺ *To simplify the query, click **Add objects to list from favorites** .*

   The query dialog box appears.
7. Start the query and select the appropriate objects in the result window.

8. Click **OK**.
The selected objects are added to the **Export Hopex Objects** dialog box list, preceded by their type.



Export HOPEX Objects

Destination
Export in Format: text format (*.MGR)
C:\ProgramData\MEGA\Hopex Application Server\5000\Repos\EnvT

File password protection
Password protect the generated file

Objects

Short Name	Name	MetaClass
AutoCad	AutoCad	Application

You can carry out this procedure several times, allowing you for example to export objects of different types.

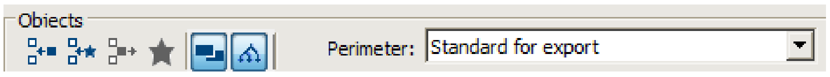
☛ *In the event of an error, click **Remove objects from list** to delete an object from the list.*

9. In the **Objects** group box, by default two export configuration options are proposed:
   - **Include Objects of Merging**, which allows you to export technical objects resulting from merging objects (_TransferredObject).

     ☛ *For further information on merging objects, see Merging Two Objects.*
   - **Propagate**, which allows you to export listed objects together with their dependent objects.

     ☛ *To view objects before export, see Viewing Objects Before Export*

10. (Optional) By default, the export perimeter is as defined in the properties of the Export tool. To modify the export default perimeter, you must have previously activated export perimeter selection, see Activating the export perimeter selection option.
In the **Objects** frame, select the **Perimeter** of export using the drop-down menu.

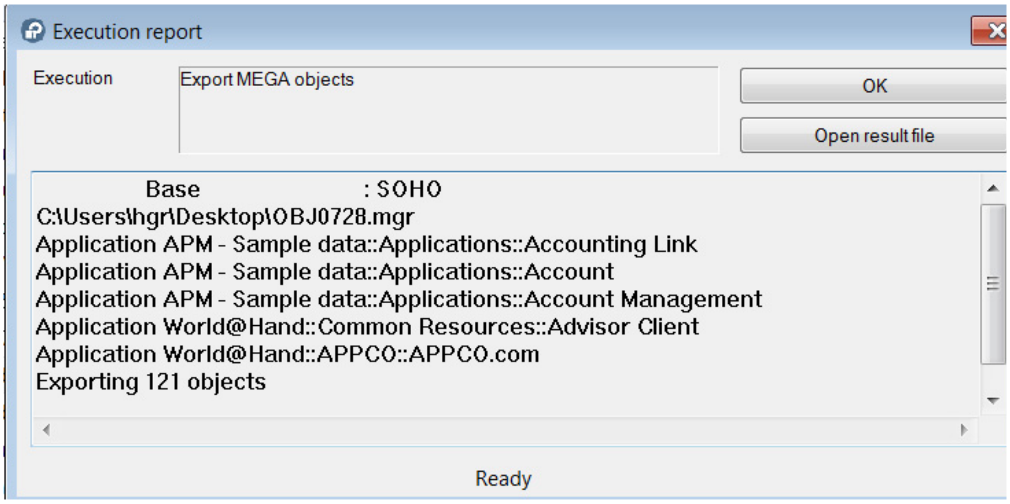


Objects
Perimeter: Standard for export

11. When selection is complete, click **Export**.
12. (If step 5 you selected the password protect file option) In the **File protection** dialog box, enter a password, confirm it and click **OK**.
    The export process begins.

    ☛ *To interrupt export during execution, click **Cancel**.*

    During export, the name and type of the objects being exported appear at the bottom of the dialog box, together with duration of export. On completion of export, the **Execution Report** displays the number of exported objects.



```
            Base                    : SOHO
C:\Users\hgr\Desktop\OBJ0728.mgr
Application APM - Sample data::Applications::Accounting Link
Application APM - Sample data::Applications::Account
Application APM - Sample data::Applications::Account Management
Application World@Hand::Common Resources::Advisor Client
Application World@Hand::APPCO::APPCO.com
Exporting 121 objects
```

☛ *See Viewing an export file.*

Exporting a Hopex object from the object

You can export an object from your **Hopex** desktop.

To export a **Hopex** object from the **Hopex** desktop:

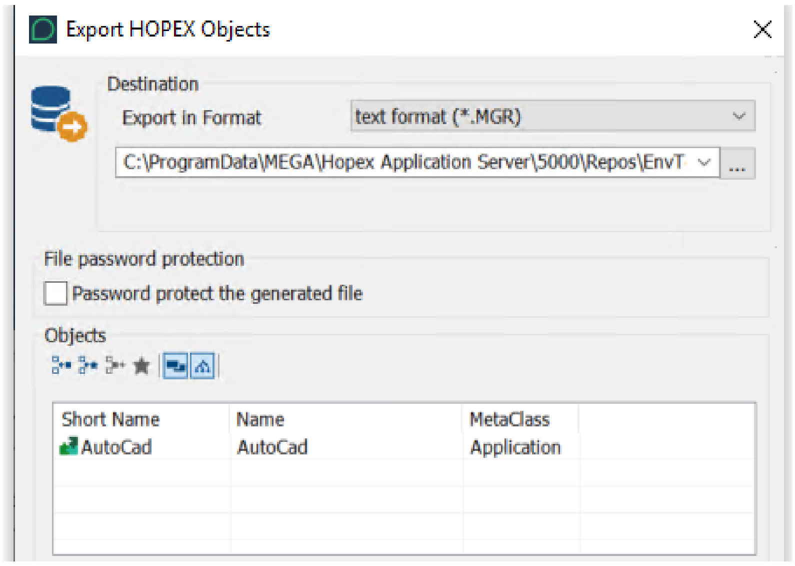
1. Connect to **Hopex**

    ☛ *See Connecting to Hopex Customizing Desktop.*

2. In the **Main Objects** navigation window, right-click the object you want to export and select **Manage > Export**.

   ☛ *You can also export **Hopex** objects by selecting **File > Export > Hopex Objects** in your workspace.*

   The **Export Hopex Objects** dialog box opens.



Export HOPEX Objects

Destination

Export in Format: text format (*.MGR)

C:\ProgramData\MEGA\Hopex Application Server\5000\Repos\EnvT

File password protection

☐ Password protect the generated file

Objects

Short Name	Name	MetaClass
AutoCad	AutoCad	Application

   In the **Objects** frame, the object to be exported is already selected.

3. Refer to the procedure Exporting Hopex objects.

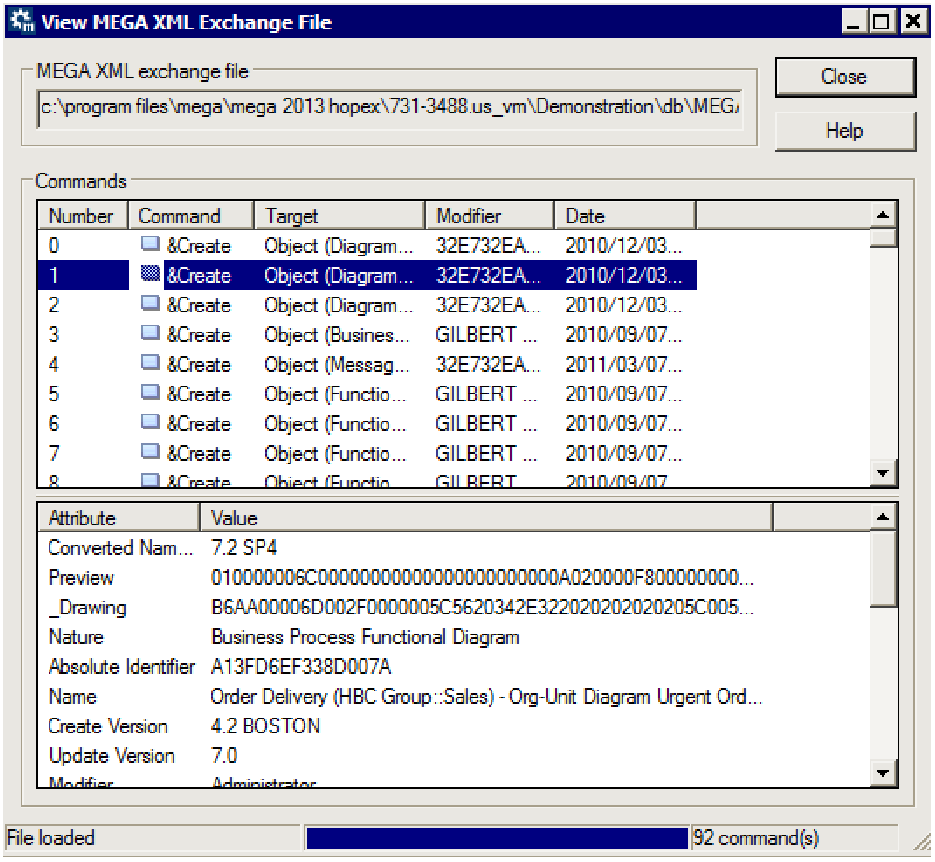
Viewing an export file

To view the exported file:

1. Export objects.

   ☛ *See Exporting Objects.*

2. In the export **Execution Report** dialog box, click **Open result file**
   In the case of:
   - an export in **MEGA** XML format, the view MEGA XML exchange file dialog box appears.



View MEGA XML Exchange File

MEGA XML exchange file

c:\program files\mega\mega 2013 hopex\731-3488.us_vm\Demonstration\db\MEG/

Commands

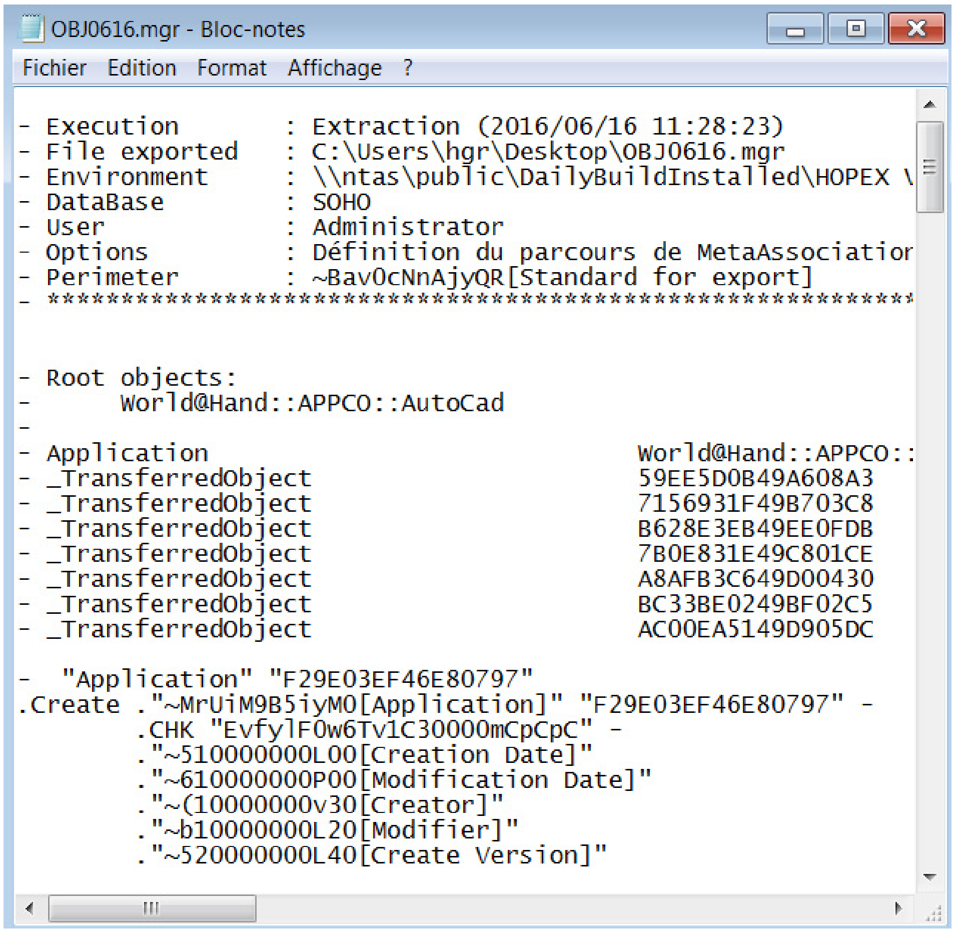
Number	Command	Target	Modifier	Date
0	&Create	Object (Diagram...	32E732EA...	2010/12/03...
1	&Create	Object (Diagram...	32E732EA...	2010/12/03...
2	&Create	Object (Diagram...	32E732EA...	2010/12/03...
3	&Create	Object (Busines...	GILBERT ...	2010/09/07...
4	&Create	Object (Messag...	32E732EA...	2011/03/07...
5	&Create	Object (Functio...	GILBERT ...	2010/09/07...
6	&Create	Object (Functio...	GILBERT ...	2010/09/07...
7	&Create	Object (Functio...	GILBERT ...	2010/09/07...
8	&Create	Object (Functio...	GILBERT ...	2010/09/07

Attribute	Value
Converted Nam...	7.2 SP4
Preview	010000006C0000000000000000000000A020000F800000000...
_Drawing	B6AA00006D002F0000005C5620342E322020202020205C005...
Nature	Business Process Functional Diagram
Absolute Identifier	A13FD6EF338D007A
Name	Order Delivery (HBC Group::Sales) - Org-Unit Diagram Urgent Ord...
Create Version	4.2 BOSTON
Update Version	7.0
Modifier	Administrator

File loaded | 92 command(s)

This file is presented in the form of a table showing a list of commands. Corresponding to each command ("Create", "Connect", etc.) there is a an object, object modification date and name of the last modifier.

- export in text format, the Notepad dialog box appears:



```
- Execution        : Extraction (2016/06/16 11:28:23)
- File exported    : C:\Users\hgr\Desktop\OBJ0616.mgr
- Environment      : \\ntas\public\DailyBuildInstalled\HOPEX \
- DataBase         : SOHO
- User             : Administrator
- Options          : Définition du parcours de MetaAssociatior
- Perimeter        : ~BavOcNnAjyQR[Standard for export]
- ************************************************************


- Root objects:
-      World@Hand::APPCO::AutoCad
-
- Application                              World@Hand::APPCO::
- _TransferredObject                       59EE5D0B49A608A3
- _TransferredObject                       7156931F49B703C8
- _TransferredObject                       B628E3EB49EE0FDB
- _TransferredObject                       7B0E831E49C801CE
- _TransferredObject                       A8AFB3C649D00430
- _TransferredObject                       BC33BE0249BF02C5
- _TransferredObject                       AC00EA5149D905DC

-  "Application" "F29E03EF46E80797"
.Create ."~MrUiM9B5iyM0[Application]" "F29E03EF46E80797" -
        .CHK "EvfylF0w6Tv1C30000mCpCpC" -
        ."~510000000L00[Creation Date]"
        ."~610000000P00[Modification Date]"
        ."~(10000000v30[Creator]"
        ."~b10000000L20[Modifier]"
        ."~520000000L40[Create Version]"
```

This file lists all objects in text format.

The exported file can then be imported into another repository.

☛ *For more details on updating a repository by command file import, see Updating a Repository.*

Activating the export perimeter selection option

☛ *This option is not available in Web Front-End.*

At the time of export, to be able to select export perimeter, you must activate the **Activate export perimeter selection** option:

1. From your **Hopex** workspace, select the **Tools > Options** menu.
2. In the Hopex **Data Exchange > Export > Files** option group: **Generic Options** select the **Activate export perimeter selection** option.

Viewing Objects Before Export

☛ *Viewing of objects is available with the **Hopex Power Supervisor** technical module.*

Viewing models with a perimeter allows preview of the operation result before its execution, and modification of the operation if required. You can therefore:

- see default impact of the applied perimeter, see Viewing objects.
- modify behavior of the perimeter according to the different types of links browsed from the root object, see the **MetaStudio** guide.

In the case of object export, other objects (connected to the root object) are also exported - they are determined by behavior of the "***Standard for export***" perimeter related to links existing around the root object.

☺ *Before exporting one or several **Hopex** objects, you may find it useful to view all objects that will be exported. These can be objects you have selected as well as those deduced by the propagation mechanism.*

☛ *For detailed information on perimeters, see the **Hopex Power Studio** - **Perimeters** technical article.*

Enabling the view option

To view objects that will be exported, you must enable the **View objects before export** option:

1. From your **Hopex** workspace, select **Tools > Options**.
2. In the Hopex **Data Exchange > Export > Files** option group: **Generic options** select **View objects before export** option.

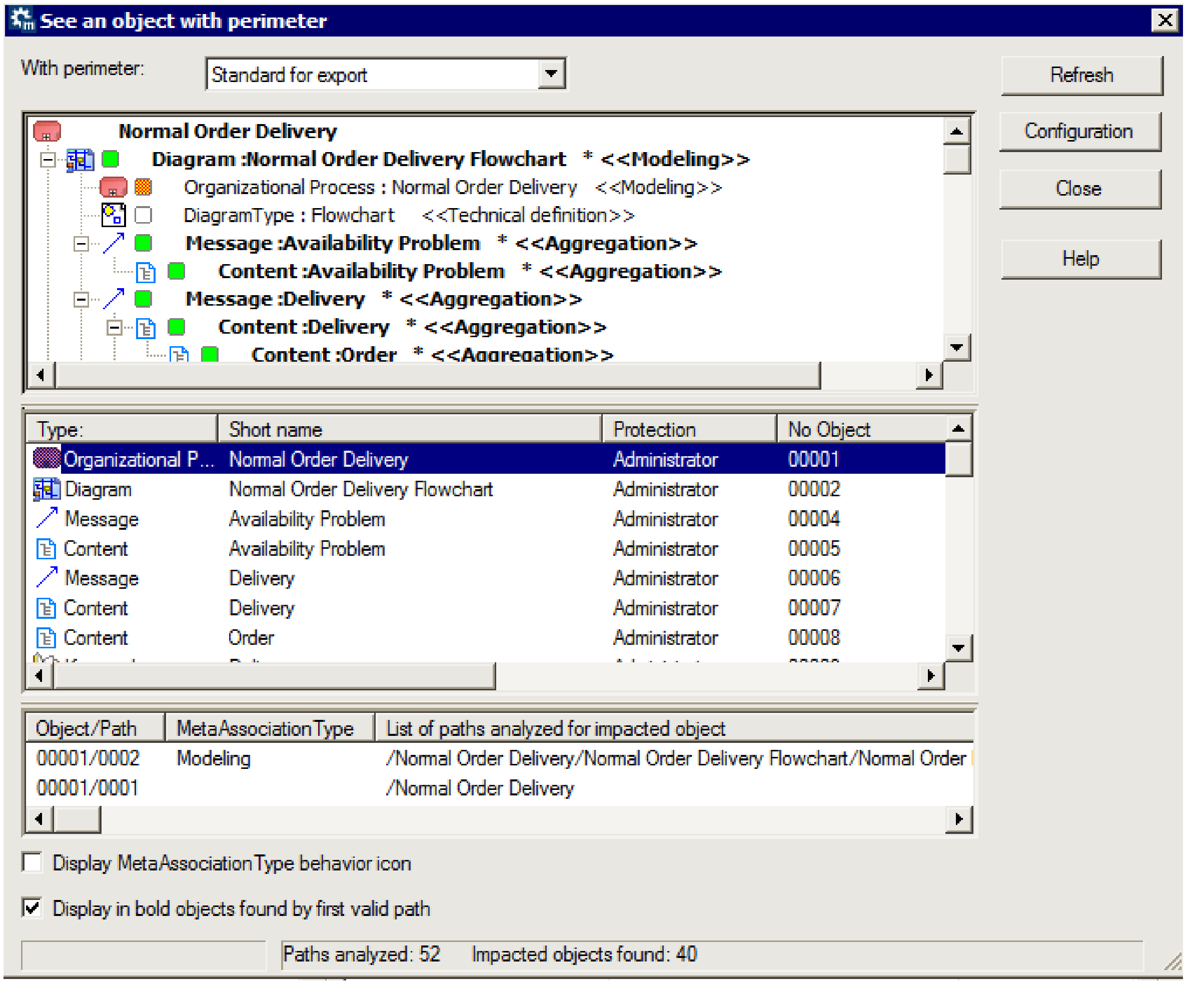
Viewing objects

To view objects that will be exported:

1. Select menu **Tools > Options**.
2. In the Hopex **Data Exchange > Export > Files** option group: select the **View objects before export** option.

   The **View** button appears in the **Objects** frame.

3. In the **Export Hopex Objects** dialog box, select an object in the list and click **View** .
The object detailed view window appears.



With perimeter: Standard for export

Normal Order Delivery
Diagram :Normal Order Delivery Flowchart * <<Modeling>>
Organizational Process : Normal Order Delivery <<Modeling>>
DiagramType : Flowchart <<Technical definition>>
Message :Availability Problem * <<Aggregation>>
Content :Availability Problem * <<Aggregation>>
Message :Delivery * <<Aggregation>>
Content :Delivery * <<Aggregation>>
Content :Order * <<Aggregation>>

Type:	Short name	Protection	No Object
Organizational P...	Normal Order Delivery	Administrator	00001
Diagram	Normal Order Delivery Flowchart	Administrator	00002
Message	Availability Problem	Administrator	00004
Content	Availability Problem	Administrator	00005
Message	Delivery	Administrator	00006
Content	Delivery	Administrator	00007
Content	Order	Administrator	00008

Object/Path	MetaAssociationType	List of paths analyzed for impacted object
00001/0002	Modeling	/Normal Order Delivery/Normal Order Delivery Flowchart/Normal Order
00001/0001		/Normal Order Delivery

☐ Display MetaAssociationType behavior icon

☑ Display in bold objects found by first valid path

Paths analyzed: 52 Impacted objects found: 40

The ***See an object with perimeter*** window presents all exported objects in two ways (result of propagation applied to the root object):

☛ *To view the impact of other perimeters on the object concerned, select another **Perimeter** in the drop-down list.*

- The top frame presents, in tree form, the objects that will be exported with the root object. For each object it details:
  - the propagation behavior defined by the "***Standard for export***" perimeter for the link browsed.
    The behavior of this operator depends on the various link types and will determine the objects that will also be exported.
  - the corresponding link type (for example ***Modeling***).
  - the propagation type (identified by an icon) that will be executed on the object.

Table: Description of propagation behaviors

Icon	Value	Propagation description
🟩	***Deep***	***Recursive complete propagation***: Takes into account this link and the opposite object only. Propagation continues.
▨	***Standard***	***Simple propagation***: Takes into account this link and the opposite object only. Propagation stops.
☐	***Link***	***Limited propagation:*** Takes into account this link but not the opposite object. Propagation stops.
■	***Abort***	***No propagation:*** Does not take into account this link or the opposite object. No propagation.

You can customize display of these results by selecting:

- **Display MetaAssociationType behavior icon**, which presents propagation behavior defined for the MetaAssociationType.
- **Display in bold objects found by first valid path**.

- the middle table lists objects that will be exported with the root object. For each object it details:
  - the number of paths linked to the object.
  - the comment associated with the object.
  - the bottom table details all paths by which the object selected in the middle table has been found, together with the corresponding link type (MetaAssociationType).

To locate an object/path in the tree:

› From one of the tables, right-click the object/path you want to locate and select **Find in tree**.

The **Configuration** button accesses the perimeter configuration tool.

☛ *For detailed information, see how to configure a perimeter in the **Hopex Power Studio - Perimeters** guide.*

PROTECTING OBJECTS

The object protection function is available with the **Hopex Power Supervisor** technical module.

An object has the writing access level of the user who created it. The writing access level of an object can be modified:

- directly
- by its dependence on another object (project, process, etc.) by propagating the authorization level for that object. This dependence may be indirect.

☛ *See Viewing Objects Before Export for more details.*

See:

- Accessing the Object Protection Management Window
- Assigning a Writing Access Area to an Object
- Propagating Authorizations Between Linked Objects

Accessing the Object Protection Management Window

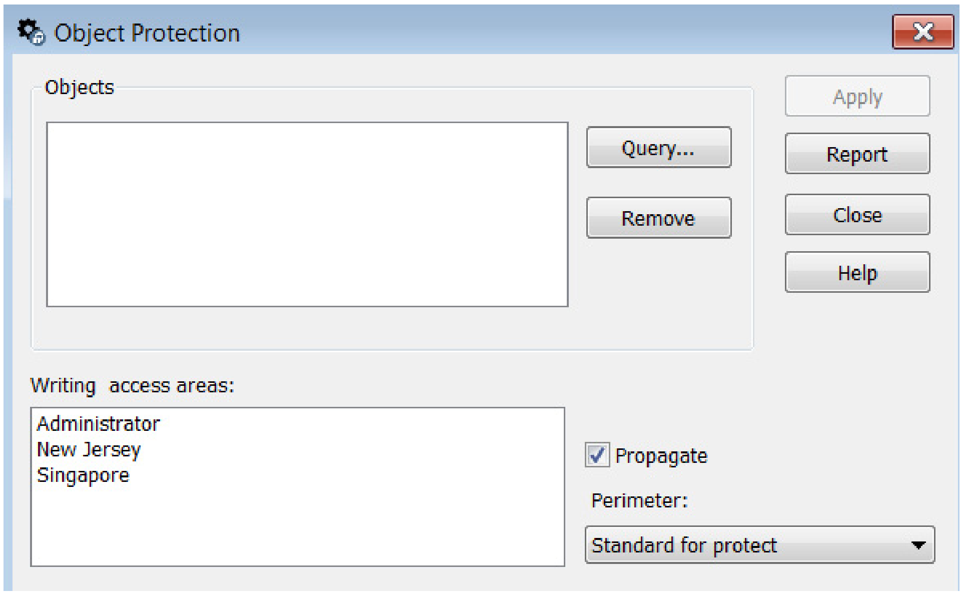
To access the object protection management window:

1. From **Hopex Administration**, connect to the environment.

   ☛ *See Connecting to an Environment.*

2. Expand the **Repositories** folder.

3. Right-click **SystemDb** and select **Object Management > Protect Objects**.
   The **Protect objects** dialog box opens.



Writing access areas: Administrator, New Jersey, Singapore

✓ Propagate

Perimeter: Standard for protect

The **Writing Access Areas** list presents the writing access area of the user accessing the dialog box, and if appropriate, its dependent access areas (it is not possible to assign writing access areas higher than its own writing access area).

Writing access areas are identified by level.

When you open this dialog box, its **Objects** list is empty.

> ☛ *See Assigning a Writing Access Area to an Object.*

> ☛ *See Propagating Authorizations Between Linked Objects.*

Assigning a Writing Access Area to an Object

To assign a writing access area to an object:

1. Access the **Object Protection** window.

   > ☛ *See Accessing the Object Protection Management Window.*

2. In the **Writing Access Areas** list, select a writing access area.
3. Click **Query** to start a query.
4. Run the query, select the appropriate objects from the results, and click **OK**.
   The selected objects are added to the **Protect objects** dialog box list. Their types are also displayed.

   > ☛ *You can carry out this procedure several times, allowing you for example to protect objects of different types.*

   > ☛ *In event of error, select the unwanted object in the list and click **Delete**.*

5. Select the objects in the list.
6. When selection is complete, click **Apply**.
   The objects are assigned the selected protection level.

   ☛ *You can select some objects in the list and assign one authorization to these, then select other objects and assign a different writing access level without executing a new query.*

Propagating Authorizations Between Linked Objects

You can automatically assign writing access level of an object to objects linked directly or indirectly to it.

1. Access the **Object Protection** window.

   ☛ *See Accessing the Object Protection Management Window.*

2. Click **Query** to select the objects concerned.
3. In the **Writing Access Areas** list, select a writing access area.
4. Select the **Propagate** check box.
5. Click **Apply**.
   Objects dependent on those for which propagation is requested are protected with the same writing access level.

   ☛ *For more details, see Viewing Objects Before Export.*

   ☛ *Note that this type of propagation may modify the authorizations for objects shared by several projects or diagrams.*

COMPARING AND ALIGNING OBJECTS BETWEEN REPOSITORIES

Hopex enables comparison and alignment of:

- two complete repositories
- objects in different repositories
- objects of the public repository with those of the current private workspace.
- two repository archived states

☛ *The objects compared must not be in the same private workspace.*

See:

- Compare and Align Principle
- Compare and Align Warnings
- Comparing and Aligning Two Repositories

Compare and Align Principle

The principle of comparing and aligning objects between repositories is as follows:

1. ***Extraction***
   The selected objects and any linked objects are extracted from the two repositories, browsing links according to **Hopex** principles of object extraction.

   Comparison

   The two sets of data are compared on the basis of *absolute identifiers* of the objects they contain.
2. ***Comparison result***
   A window displays the results of the comparison. You can also generate a report and a command file in this window.

   ☛ *The page showing differences displays a maximum of 1000 lines. If the list of differences is greater than 1000 lines, a message prompts you to either ignore this limit and display all the lines (in this case, the list may take some time to load) or not.*
3. ***Alignment***
   The upgrade command file is imported in the target repository.

Compare and Align Warnings

You must be aware of the following points before alignment and selection of the user executing alignment.

In case the compare and align includes a large amount of data, this action can take some time and slow down Hopex

performance. Remember to perform this action when Hopex users are cont connected.

Repository log

The repository log lists all modifications made in the repository. It gives users a better understanding of actions executed in a repository in private workspaces. Each time an action is executed, an occurrence of Change Item is created.

☛ *For more details on repository log, see Repository log.*

The repository log is not transferred from one repository to the other: a new log is created in the target repository. Object history is not therefore kept.

Users

The creator/modifier of an object in the target repository is the user executing the alignment.

The date of creation of an object is the date on which alignment was executed.

Reading (confidentiality) and writing access levels

Writing and reading access levels are taken into account during the comparison and during the alignment.

☛ *For more details on writing and reading access management, see Data Reading Access and Data Writing Access.*

To perform a comparison and an alignment, you must have reading access (if reading access management is activated) and maximum reading access for all objects in the repository.

☛ *Reject files are generated on completion of alignment. To delete files: in environment options **Options > Tools > Data Exchange > Exchange with third party tool > MEGA**, select the option **Delete files produced at compare/align on completion of processing**.*

Comparing and Aligning Two Repositories

☛ *Before comparing and aligning, see Compare and Align Warnings*

To compare and align two repositories:

1. Connect to **Hopex**

   ☛ *See Connecting to Hopex Customizing Desktop.*

   ☛ *You can also, from **Hopex Administration**, right-click the repository concerned and select **Compare and Align**.*

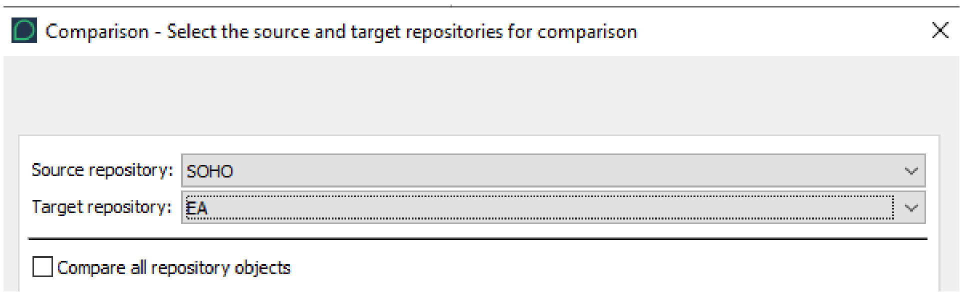
2. In the **Hopex** menu bar, select **Tools > Manage > Compare and Align**.

   ☛ *You can also right-click the object and select **Manage > Compare and Align**.*

   The object comparison wizard opens.

3. Indicate if you want to compare:
   - two repositories
   - two current repository archived states (RDBMS repository only)
4. Click **Next**.
5. Select:
   - the **Source repository**
   - the **Target repository**, which is the repository to be updated.

   ☛ *It can be a private workspace of the repository.*


Comparison - Select the source and target repositories for comparison

Source repository: SOHO
Target repository: EA

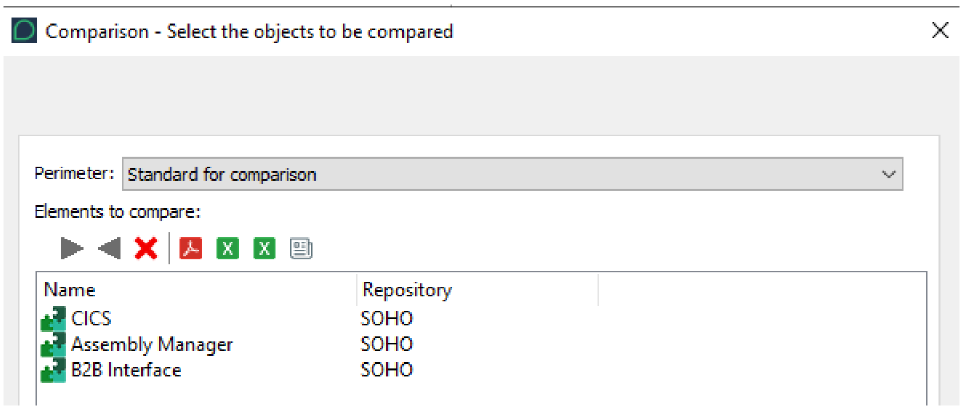
Compare all repository objects

6. (Optional) If required, you can choose to **Compare all repository objects**.

   ☛ *Processing of this option can take some time.*
7. Click **Next**.
   The dialog box for selection of objects to be compared opens.
8. In the **Perimeter** field, select the perimeter type (by default **Standard for Comparison**)

   ☛ *For detailed information on perimeters, see the **Hopex Power Studio - Perimeters** technical article.*
9. In the **Elements to compare** pane, select:
   - ▶ to add objects from the source repository, or
   - ◀ to add objects from the target repository.

   ☛ *If you have opened the comparison wizard from an object, this object is automatically added in the list of objects to be compared.*

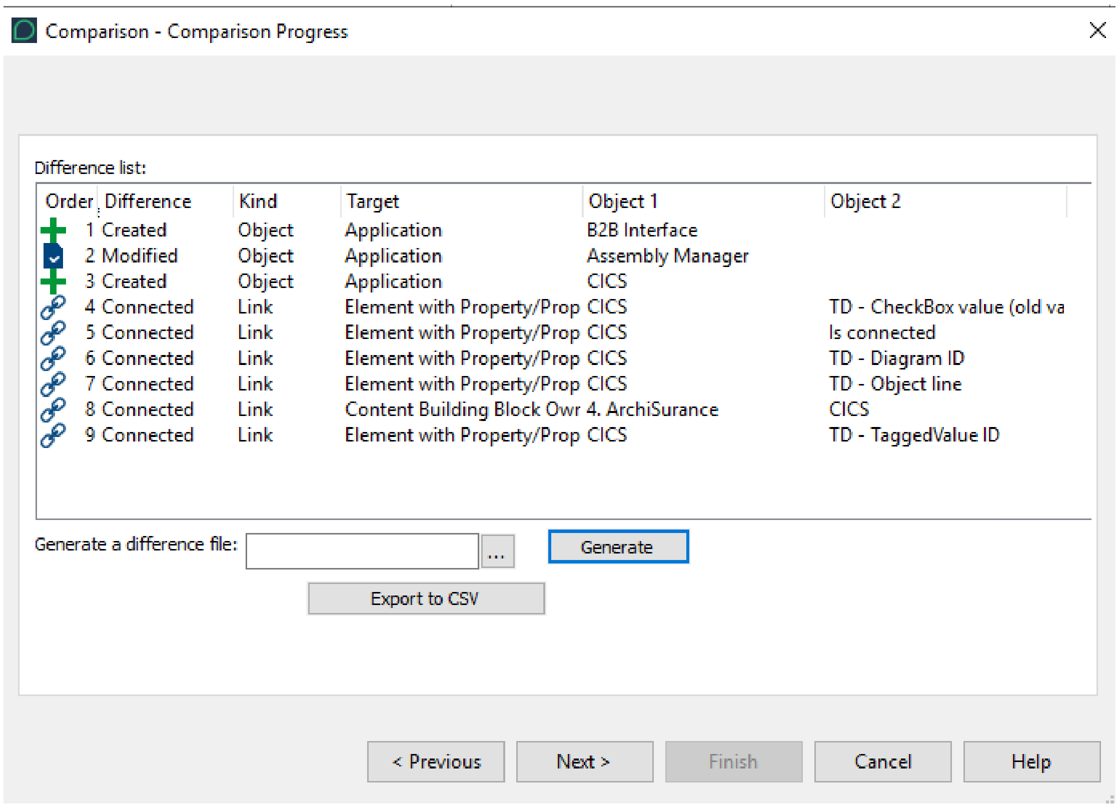

Comparison - Select the objects to be compared

Perimeter: Standard for comparison

Elements to compare:

Name	Repository
CICS	SOHO
Assembly Manager	SOHO
B2B Interface	SOHO

10. Click **Next**.
    The **Comparison Progress** window opens. It presents the differences between compared objects and their modifications.



Order	Difference	Kind	Target	Object 1	Object 2
1	Created	Object	Application	B2B Interface	
2	Modified	Object	Application	Assembly Manager	
3	Created	Object	Application	CICS	
4	Connected	Link	Element with Property/Prop	CICS	TD - CheckBox value (old va
5	Connected	Link	Element with Property/Prop	CICS	Is connected
6	Connected	Link	Element with Property/Prop	CICS	TD - Diagram ID
7	Connected	Link	Element with Property/Prop	CICS	TD - Object line
8	Connected	Link	Content Building Block Owr	4. ArchiSurance	CICS
9	Connected	Link	Element with Property/Prop	CICS	TD - TaggedValue ID

The **Difference** column presents differences by update category:

- ***Created***: objects not existing in the target repository.
- ***Deleted***: objects existing in the target repository but not in the source repository.

  ☛ *Deletion commands of compare and align can be generated in a separate file. To do this, activate the corresponding option in **Options > Data Exchange > Import/Export Synchronization > MEGA**.*

- ***Modified***: objects of which characteristics, including name, have been modified.
- ***Connected***: links, between two objects, that do not exist in the target repository.
- ***Disconnected***: links existing in the target repository but not in the source repository.
- ***Changed***: links for which a characteristic has been modified.

The **Kind** column presents differences by type.

11. In the **Generate a difference file** field (.mgr format):
    - Click ... and enter the name and location of the comparison file, then click **Save**.
    - Click **Generate** to generate the .mgr file that contains the list of differences detected.

12. (Optional) Click **Export CSV** to generate a CSV file of the differences detected.

    ☛ *The cmd-align-YYYY-MM-DD-hh-mm-ss file is stored in <environment name>\DB \<repository name>\USER\<user code>.*

13. Click **Next**.
    Differences are imported in the target repository.

    The target repository is aligned with the source repository.

    ☛ *An alignment file with the content of differences (align-YYYY-MM-DD-hh-mm_555.mgr) is automatically saved in folder <Environment Name>\Db\<Repository Name>\USER\<User Code>.*

    If the alignment contains rejects, click **Display Rejects** to open the alignment rejects file (.mgr,format).

    A rejects file is automatically saved in folder <Environment name>\Db\<Repository name>\USER\<User Code> (rejects file-reject-YYYY-MM-DD-hh-mm_555.mgr). This file is empty if alignment does not contain rejects.

14. Click **Finish**.

MERGING TWO OBJECTS

The object merge feature is available with the **Hopex Power Supervisor** technical module.

When you merge two objects, you obtain a single object by transferring the *characteristics* and *links* from one object to the other. The source object is deleted. It is therefore recommended that merges be done in a new private workspace, so you can discard the changes if the result is not satisfactory.

You must have administration rights to merge two objects.

Choice of the objects to be merged

The **Target** object is the reference object that will be merged with the **Source** object. By default:

- its characteristics are not modified
- merging proposes addition of source object links.

The **Source** object is the object of which:

- you want to reuse certain characteristics or certain links
- characteristics and links will be transferred to the **Target** object.

When the link is to be a unique link (e.g., for sub-typing where the type is unique), the link of the target object is kept by default.

When merging is completed, the source object is deleted.

☺ *You can **Explore** objects using the corresponding command. It can also be used to explore their links.*

Merging Two Objects

To merge two objects:

1. Connect to **Hopex**.

   ☛ *See Connecting to Hopex Customizing Desktop.*

2. In the menu bar, select **Tools > Manage > Merge Two Objects**.
   The first step of the **Merge objects** wizard appears: **Object selection**.

   ☛ *To have the right to merge two objects, you must have the right to delete objects.*

3. In the **Object** pane, select the object type to be merged.

   ☛ *See Choice of the objects to be merged.*

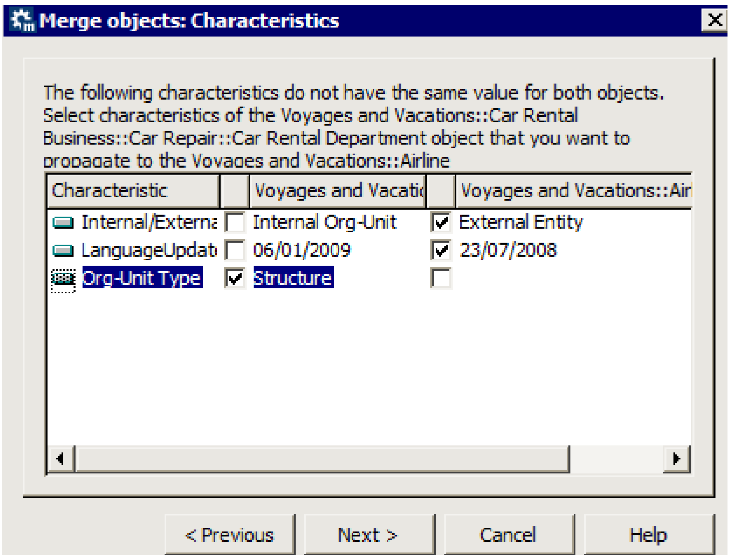
4. In the **Source** field, click the arrow and select the source object.
5. In the **Target** field, click the arrow and select the target object.

6. Click **Next**.

   ☛ *If the target and source objects are the same, the **Next** button is disabled.*

   The second step of the **Merge objects** wizard appears: **Characteristics**. It presents the differences found in characteristic values of both objects.

7. Select the source object characteristics you want to transfer to the target object. Characteristics that remain selected in the target object are kept.



The following characteristics do not have the same value for both objects. Select characteristics of the Voyages and Vacations::Car Rental Business::Car Repair::Car Rental Department object that you want to propagate to the Voyages and Vacations::Airline

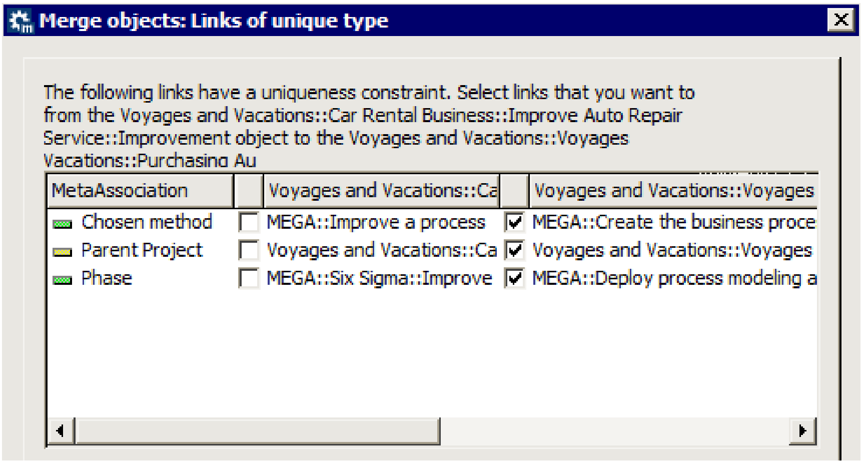
Characteristic		Voyages and Vacatio		Voyages and Vacations::Air
Internal/Externa	☐	Internal Org-Unit	✓	External Entity
LanguageUpdat	☐	06/01/2009	✓	23/07/2008
Org-Unit Type	✓	Structure	☐	

8. Click **Next**.

   The third step of the **Merge objects** wizard appears: **Unique links** (those that can only exist once for a given object).

   ```
   Example: a message can have only one super-type.
   ```

   ☛ *This step appears only if the objects to be merged have unique links connecting them to different objects.*

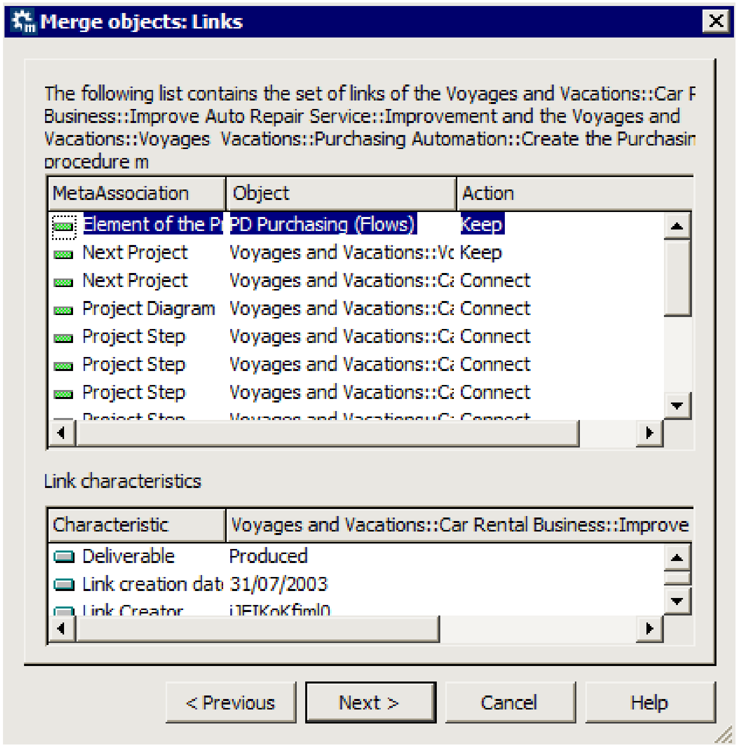


The following links have a uniqueness constraint. Select links that you want to from the Voyages and Vacations::Car Rental Business::Improve Auto Repair Service::Improvement object to the Voyages and Vacations::Voyages Vacations::Purchasing Au

MetaAssociation		Voyages and Vacations::Ca		Voyages and Vacations::Voyages
Chosen method	☐	MEGA::Improve a process	✓	MEGA::Create the business proce
Parent Project	☐	Voyages and Vacations::Ca	✓	Voyages and Vacations::Voyages
Phase	☐	MEGA::Six Sigma::Improve	✓	MEGA::Deploy process modeling a

9. Select the links to be transferred.

10. Click **Next**.
The **Merge objects** wizard shows: **Links**.



The following list contains the set of links of the Voyages and Vacations::Car F Business::Improve Auto Repair Service::Improvement and the Voyages and Vacations::Voyages Vacations::Purchasing Automation::Create the Purchasin procedure m

MetaAssociation	Object	Action
Element of the P	PD Purchasing (Flows)	Keep
Next Project	Voyages and Vacations::Vc	Keep
Next Project	Voyages and Vacations::Ca	Connect
Project Diagram	Voyages and Vacations::Ca	Connect
Project Step	Voyages and Vacations::Ca	Connect
Project Step	Voyages and Vacations::Ca	Connect
Project Step	Voyages and Vacations::Ca	Connect

Link characteristics

Characteristic	Voyages and Vacations::Car Rental Business::Improve
Deliverable	Produced
Link creation dat	31/07/2003

- When the link does not exist for the target object, the default is to connect the target object (**Action** : "Keep"). You can select the **Do not connect** command so as not to transfer the link.

You can **Keep** existing links, or **Disconnect** them.

- When the two objects are linked to the same object by the same link, it is possible to **Not change characteristics** of the link for the target object, or to **Copy characteristics** of the link for the source object. In this case, you can indicate for each characteristic whether to use the value of the source object, or keep the value of the target object.

11. Click **Next**.

12. Click **Finish** to start merging.
The gauge indicates progress of the operation.
When merging has been completed, the source object no longer exists and the selected *characteristics* and *links* have been transferred to the target object.

> ☛ *"_TransferredObject" temporary merge objects are created on this occasion. Merge objects of a repository can all be exported at export of **Hopex** objects.*

Managing Data Access Dynamically

Writing and reading access diagrams define data access statically. A person sees objects belonging to his/her reading access area, and can modify objects belonging to his/her writing access area.

☛ *See Data Writing Access, Data Reading Access.*

You can define dynamic rules for reading or writing data access.

Dynamic rule:

- applies to an object for given profiles
- is defined by a macro

Attention regarding confidentiality management

An object is associated with a confidentiality level and you must be careful while setting up dynamic data access rules.

- ***Static mode***:
  Confidentiality management is taken into account through reading and writing access diagrams, as they both manage data access statically.
- ***Dynamic mode***:
  Confidentiality management might not be always taken into account through data access rules, as they manage data access dynamically.

  When a user generates certain types of documentation (e.g.: Web site, report), this documentation is generated with the data access rules of the person who generates it. Once cached, this documentation might not take into account the confidentiality of the user who will read this documentation (e.g.: Web site, report), which might not follows the same data access rules.

Implementing a dynamic data access rule

☛ *See Use case: data access rule set up.*

A dynamic data access rule:

- defines for a person, his/her reading or writing access rights on a given object

  ☛ *The rule can be applied to several objects.*
- can be based on characteristics of an object, a person, or an object and a person
- can be called at object creation
- can be associated with one or several profiles

  ☛ *By default the rule is associated with all the profiles.*

To manage dynamic data access on an object, you must implement a permission rule:

1. Create the macro for the permission rule.

   ☺ *For information on writing the macro, see Hopex Studio > Using APIs: Optimizing the macro of a dynamic data access rule.*

2. Create the permission rule.

   ☛ *See Creating a permission rule (data access rule).*

3. (If needed) Define the profile to which the rule applies.
   By default the rule applies to all profiles.

   ☛ *See Associating a permission rule with a profile.*

4. Associate the permission rule with the object concerned by the rule.
   The rule may apply to several objects.

   ☛ *See Associating a permission rule with an object.*

Creating a permission rule (data access rule)

A permission rule is defined by a macro. A permission rule can define reading or writing access rights on an object.

To create a permission rule:

1. In **Hopex** (Windows Front-End), from the **Hopex** explorer, click **Create** ➕.
2. Select **Data Access Rule** and click **OK**.
3. In the **Creation of Data Access Rule** dialog box, enter a **Name** for the rule and click **OK**.
4. Access properties of the rule.
5. In the **Characteristics** tab, in the **Macro** field, click the arrow and connect the macro that manages the rule.
6. In the **Data Access Type** field, select the data access type (**Reading** or **Writing**).
   In the **User Profile** frame, if no profile is connected to the rule, the rule applies to all profiles.

   ☛ *See Associating a permission rule with a profile.*

7. (To call the data access rule at object creation) In the **Texts > _Settings** tab enter:

```
[General]
RelaxCreationTime=0
```

Associating a permission rule with a profile

☛ *To associate a dynamic permission rule with a profile provided at installation, see PLATFORM - Studio > Customizing the characteristics of an existing profile / Creating a profile from an existing profile documentation.*

To associate a permission rule with a profile:

1. Open permission rule properties.

   `Example: "Action Plan - Writing"`
2. Click the **Characteristics** tab.
3. In the **User Profile** frame, click **Connect** and select the profile with which you want to associate the permission rule.

   ☛ *You can connect several profiles.*

Associating a permission rule with an object

☛ *To associate a dynamic permission rule to an object, you must have rights to modify **Hopex** data, see Managing Hopex Data Customization.*

To associate a permission rule with an object:

1. Open object properties.

   `Example: MetaClass “Risk”.`
2. Select the **Data Access** tab.
3. In the **Data Access Rule** frame, click **Connect** and select the rule you want to associate with the object.

Use case: data access rule set up

The same permission rules have been set up for both MetaClasses:

- **Processing Activity**
- **Data Transfer**

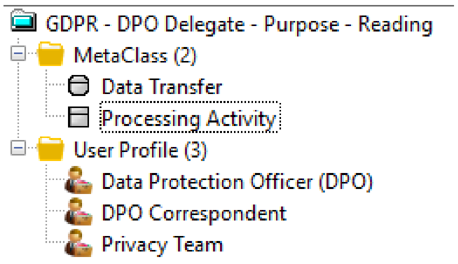
The visibility (access rights) of these MetaClasses is customized according to the user profile:

- **Data Protection Officer (DPO)**
  The Data Protection Officer (DPO) works independently to ensure that an entity is adhering to the policies and procedures set forth in the GDPR. The DPO edits processing activities, carries out pre-assessments as well as DPIAs.
- **DPO Correspondent**
  The DPO Correspondent (Privacy) plays the same role as the DPO but his tasks are restricted to a sub-set of the organization.
- **Privacy Team**
  The Privacy Team is made of operational people who carry out the instructions of the DPO or the Chief Privacy Officer.

The visibility (access rights) of these MetaClasses is managed through three data access rules.

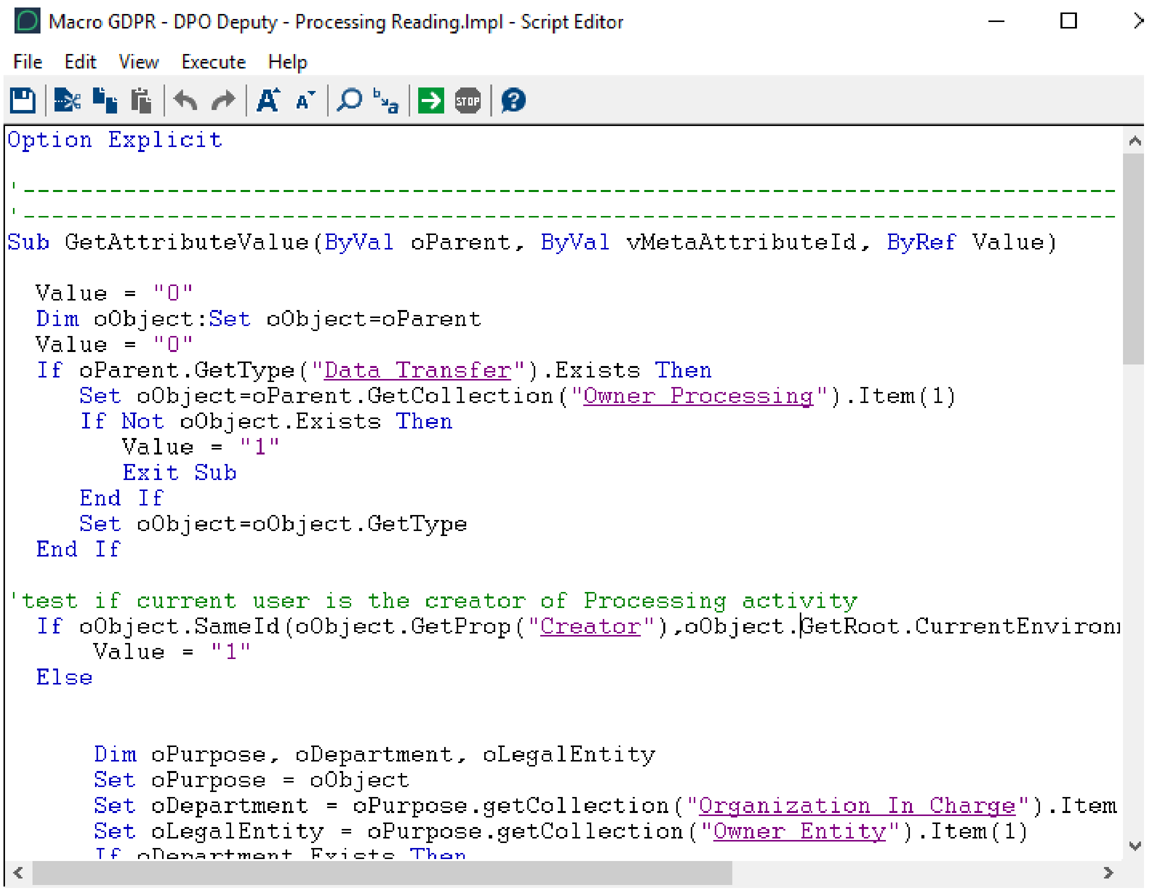
E.g.: the “GDRP - DPO Delegate - Purpose - Reading” data access rule applies to both **Data Transfer** and **Processing**

Activity MetaClasses for **Data Protection Officer (DPO), DPO Correspondent** and **Privacy Team** profiles.


GDPR - DPO Delegate - Purpose - Reading
MetaClass (2)
Data Transfer
Processing Activity
User Profile (3)
Data Protection Officer (DPO)
DPO Correspondent
Privacy Team

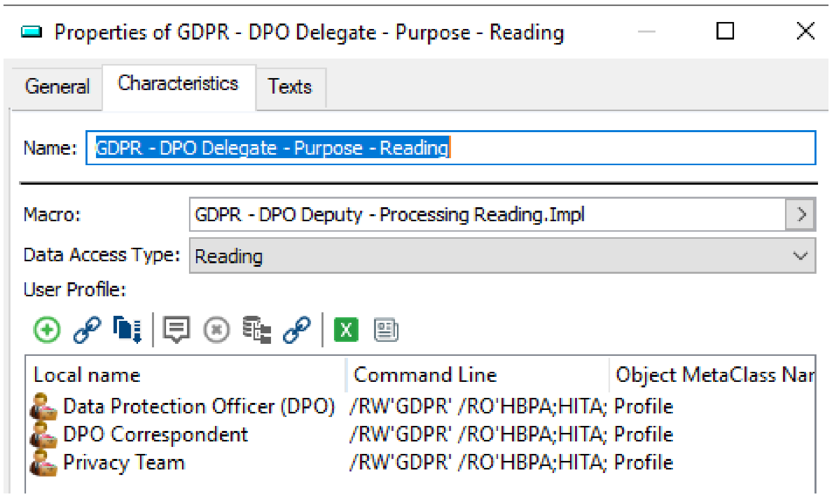
Principle of a permission rule setup on **Data Transfer** and **Processing Activity** MetaClasses:

1. Creation of the macros that manage the rules:
   - GDPR - Activity Owner PrAct - Readig.Implementation
   - GDPR - Purposes -App Owner - ReadingImplementation
   - GDPR - DPO Deputy - Processing Reading.Impl


Macro GDPR - DPO Deputy - Processing Reading.Impl - Script Editor
File Edit View Execute Help

```
Option Explicit

'----------------------------------------------------------------------------
'----------------------------------------------------------------------------
Sub GetAttributeValue(ByVal oParent, ByVal vMetaAttributeId, ByRef Value)

  Value = "0"
  Dim oObject:Set oObject=oParent
  Value = "0"
  If oParent.GetType("Data Transfer").Exists Then
     Set oObject=oParent.GetCollection("Owner Processing").Item(1)
     If Not oObject.Exists Then
        Value = "1"
        Exit Sub
     End If
     Set oObject=oObject.GetType
  End If

'test if current user is the creator of Processing activity
  If oObject.SameId(oObject.GetProp("Creator"),oObject.GetRoot.CurrentEnviron
      Value = "1"
  Else


      Dim oPurpose, oDepartment, oLegalEntity
      Set oPurpose = oObject
      Set oDepartment = oPurpose.getCollection("Organization In Charge").Item
      Set oLegalEntity = oPurpose.getCollection("Owner Entity").Item(1)
      If oDepartment.Exists Then
```

2. Creation of the data access rules associated with each macro:
   - **Data Access Type**: "Reading"
   - **Profiles** associated with the rule: **Data Protection Officer (DPO)**, **DPO Correspondent**, **Privacy Team**.



Properties of GDPR - DPO Delegate - Purpose - Reading

General | Characteristics | Texts

Name: GDPR - DPO Delegate - Purpose - Reading

Macro: GDPR - DPO Deputy - Processing Reading.Impl

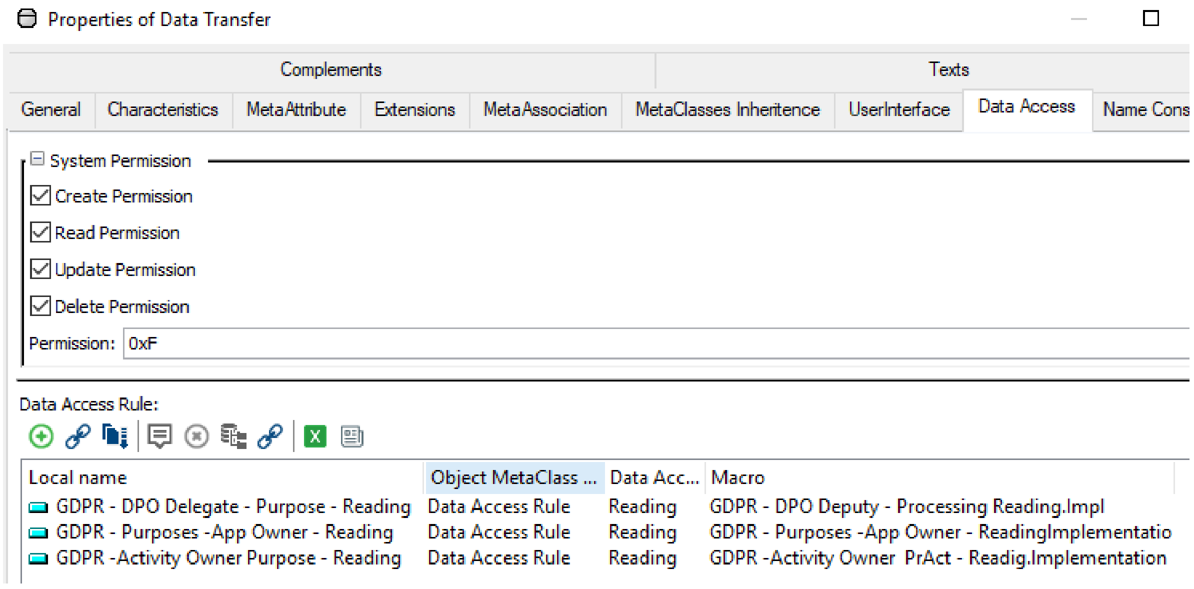
Data Access Type: Reading

User Profile:

Local name	Command Line	Object MetaClass Nar
Data Protection Officer (DPO)	/RW'GDPR' /RO'HBPA;HITA;	Profile
DPO Correspondent	/RW'GDPR' /RO'HBPA;HITA;	Profile
Privacy Team	/RW'GDPR' /RO'HBPA;HITA;	Profile

3. Connecting the data access rules with the **Data Transfer** and **Processing Activity** MetaClasses.

   E.g.: in the **Data Transfer** MetaClass properties, **Data Access** tab, all of the three rules are connected to the MetaClass.



Properties of Data Transfer

Complements | Texts

General | Characteristics | MetaAttribute | Extensions | MetaAssociation | MetaClasses Inheritence | UserInterface | Data Access | Name Cons

System Permission
- ☑ Create Permission
- ☑ Read Permission
- ☑ Update Permission
- ☑ Delete Permission

Permission: 0xF

Data Access Rule:

Local name	Object MetaClass ...	Data Acc...	Macro
GDPR - DPO Delegate - Purpose - Reading	Data Access Rule	Reading	GDPR - DPO Deputy - Processing Reading.Impl
GDPR - Purposes -App Owner - Reading	Data Access Rule	Reading	GDPR - Purposes -App Owner - ReadingImplementatio
GDPR -Activity Owner Purpose - Reading	Data Access Rule	Reading	GDPR -Activity Owner PrAct - Readig.Implementation

MANAGING SHAPES

Shapes provided by Hopex

Hopex provides several sets of shapes used to represent the objects in diagrams:

- pictures (icons)
- pictures.7220
- pictures.7600 (corresponding to Hopex V1R3 pictures)
- pictures.9000 (corresponding to Hopex V4 pictures)
- pictures.15000 (corresponding to Hopex V5 pictures)
- pictures.17000 (corresponding to Hopex Aquila pictures)

Each set of shapes is sorted by categories:

- Art
- Background
- Method

The shapes provided by **Hopex** are stored in:

- <Hopex Application Server>\ <name of the HAS instance>\.shadowFiles\hopex.core\<module version>\Mega_Std

```
E.g.: C:\ProgramData\MEGA\Hopex Application
Server\5000\.shadowFiles\hopex.core\17.0.0+6442\Mega_Std
```

Customizing the shapes

To customize the shapes provided by **Hopex** you must first install **Hopex Application Server Customization** module.

☛ *To install **Hopex Application Server Customization** module, see **Modules > Installing HAS Customization module** documentation.*

Your customized shapes are stored in:

- <Hopex Application Server>\ <name of the HAS instance>\.shadowFiles\has.custom\<module version>\hopex.core\Mega_Std

```
E.g.: C:\ProgramData\MEGA\Hopex Application
Server\5000\.shadowFiles\has.custom\15.2.0+13\hopex.core\Me
ga_Std
```

Your shapes overload the shapes provided by **Hopex**.

DATA WRITING ACCESS

Hopex Administration is provided with tools required for writing access management.

This chapter explains how to create a *writing access diagram* and how to customize its characteristics.

The following points are covered here:

Introduction to Writing Access Management

☛ *Managing writing access areas is available with the **Hopex Power Supervisor** technical module only.*

The administrator declares the users and defines the writing access.

The structure of *writing access* is defined in the *writing access diagram*. There is a hierarchical link between writing access.

☺ *Implementation of this function does not replace a structured management of projects. To operate, the writing access diagram must model itself on organization of projects.*
Clear functional breakdown of your projects simplifies management of operational follow-up of writing accesses.

Users

user

A user is a person with a login and at least one assigned profile.

At creation of an environment, two users are declared by default. These two users have administration rights to manage repositories and users:

- the "Administrator" person (Login "System", password: Hopex)

  ☛ *The "Administrator" user cannot be deleted. It has no profile (it has all rights).*

- The "Mega" person (Login "mega", password: Hopex)

You must declare other users who will access repositories.

If several environments are defined for a site, users must be defined in each of these environments. To do this, export the user diagram from the reference environment, and import it into each of the other environments.

Do not manually create a user with the same name in other environments. If you do, the user will have a different absolute identifier in each environment, and you would actually have created different users with the same name.

☺ *Bizzdesign recommends that you create the repositories before defining the users, so you can declare the user access rights when they are created.*

To access a repository, a user must identify himself/herself. Then, depending on user writing access area, he/she is able to modify the repository.

User group

A person can belong to one or more groups. A user group is a group of persons with a login.

Persons belonging to a group:

- depend on the same environment.
- share the same connection characteristics defined by the **profile** of the group and its assignment.
- connect to the application with their **login**, but with access rights defined on the **login** of the group.
- share the assignments defined for the group.
- share the personal characteristics defined for the group.

Writing Access Areas

Each user or group of users is connected to a writing access area. It is the person or person group that carries the writing access area.

Each object is connected to a writing access area.

☛ *At creation, the object inherits the writing access area of the person who created it.*

The "Administrator" writing access area is provided by default at installation. This writing access area:

- cannot be deleted.
- is the highest writing access area level; it does not depend on any other writing access area. In principle it should be reserved for repository administration.
- is the writing access area to which "Administrator" and "Mega" are connected.
  When the writing access diagram has been installed, **Bizzdesign** recommends that you change the writing access area level of "Mega". It is not advisable that a default user has such extensive rights.

All other writing access areas depend on at least one writing access area.

Writing access areas are interconnected by hierarchical links. This is a strict hierarchy, with no circular dependencies: a writing access area cannot be declared at a higher level than the writing access area on which it depends, either directly or via a succession of dependencies.

A user can modify an object connected to his/her writing access area or to a hierarchically lower writing access area.

The writing access area of an object can be modified by the administrator:

- by specifically changing the object writing access area
- when modifying the writing access area of another object (project, process, diagram, etc.) if the propagation option is enabled.

Writing Access Diagram

There is one writing access diagram per environment.

☛ *If several environments use the same protection configuration, the same user diagram must be used in all these environments.*

Rules

Installation of a writing access diagram diagram must respect the following rules so as to minimize user management costs:

- Any object must be modifiable by users that may need to modify it, without administrator intervention.
- The administrator should intervene only in exceptional circumstances.

Use

The writing access diagram is used similarly to a diagram. The persons, person groups and writing accesses are handled just like standard objects:

- Creation and modification of the name, etc., are done in the same way as for standard objects.
- Be careful however when you delete a writing access.

☛ *See Deleting Writing Access Areas.*

Avoid using the Cut command for a person or person group, as this can result in errors in the writing access diagram if the person or person group is not deleted from the repository or not linked to a writing access.

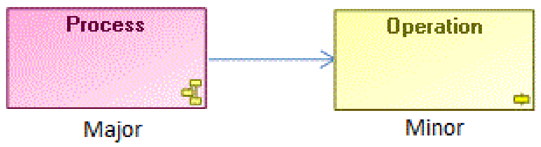
Link Orientation: Major and Minor Objects

When two objects are linked, one object is *major* and the other *minor*.

You cannot delete a minor object if you do not have writing access on the major object.

The major object in the link is the one whose nature changes with the presence or absence of the link. For example a process, defined as a succession of operations, is modified if you remove an operation. The process is then major for the link. If the objects are protected, you must have the correct authorization for modifying the major object in order to create or delete the link.

The minor object in a link is the one whose nature is not modified or only slightly modified by presence or absence of this link. For example, removing an operation from a process does not change characteristics of this operation. Therefore the process is minor in the link.



In the above example, you must have writing access on the org-unit (the major object) to disconnect or delete the message (minor object).

OPENING THE WRITING ACCESS DIAGRAM

☛ *To access the writing access diagram, you must have a license for the **Hopex Power Supervisor** technical module.*

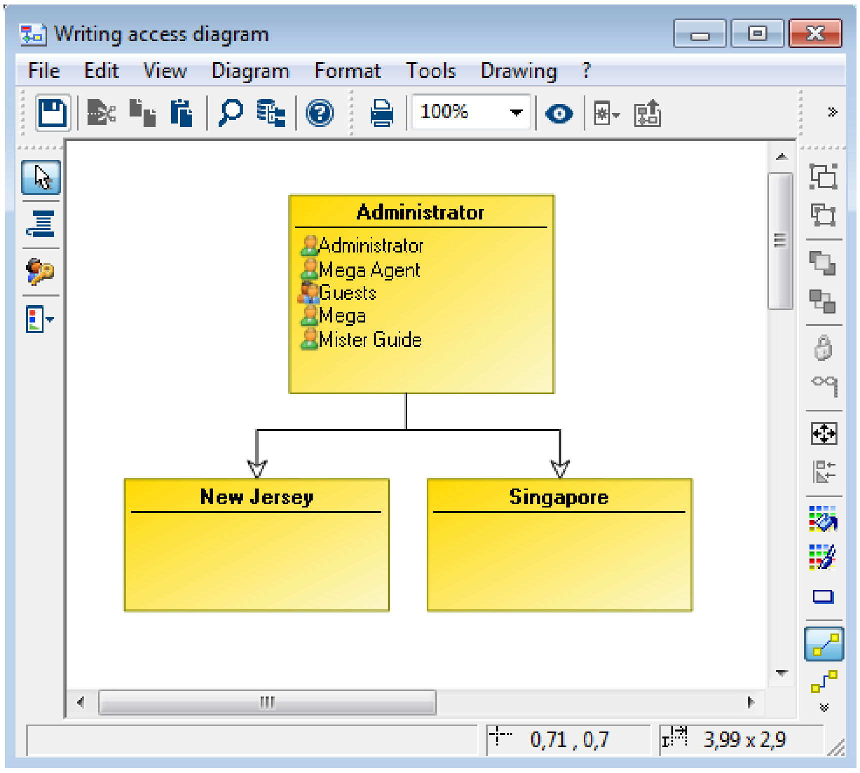
Opening the Writing Access Diagram (Windows Front-End)

To open the writing access diagram:

1. From **Hopex Administration**, connect to the environment concerned.

   ☛ *See Connecting to an Environment.*

2. In the **User Management** folder, right-click **Data Writing Access** and select **Open Diagram**.
   The diagram opens in a new window.



COMPILING THE WRITING ACCESS DIAGRAM

Running writing access diagram compilation assures consistency of behavior of **Hopex** with declarations of the diagram.

If the diagram is not compiled, there is a risk that certain users will be able to update objects that are normally protected.

When modifying the writing access diagram, so as to warn of rejects due, for example, to writing access restrictions or deletions before compilation it is recommended that:

- all changes made on the user workstations should be uploaded to the administrator workstation, or
- all private workspaces dispatched.

To compile the writing access diagram from the **Administration** application:

1. From **Hopex Administration**, connect to the environment.

   ☛ *See Connecting to an Environment.*

2. Expand the **User management** folder of the environment.
3. Right-click the **Data Writing Access** folder and select **Compile**.
   When compilation is complete, a message indicates whether the operation was successful or whether the diagram contains errors.

   ☛ *The most frequent errors are:*

   A writing access area (other than '"Administrator") is not attached to any other.

   A person or person group is not attached to any authorization.

Defining Writing Access Areas

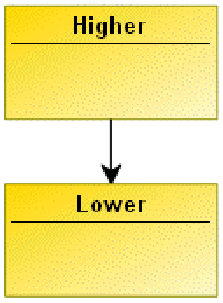
The writing access diagram is available if you have the Hopex Power Supervisor technical module. With this diagram, you can create users and manage their writing access rights for repositories and product functions. By default only one writing access area is defined, named "Administrator". Attached to it are the "Administrator" and "Mega" persons. This is the highest writing access level and it should normally be reserved for repository management. You cannot delete this writing access area.

The *writing access diagram* enables creation of writing access areas.

Creating a Writing Access Area

To create a writing access area:

1. In the diagram insert toolbar, click **Writing Access Area** then click in the diagram.
2. Enter the name of the writing access area.
   Dependency of writing access areas is determined by creation of a "lower" link which starts from the higher writing access area to the lower writing access area.



Defining Writing Access Area Persons or Person Groups

To define persons or person groups of a writing access area:

1. From **Hopex Administration**, open the writing access diagram.
   ☛ *See Opening the Writing Access Diagram.*
2. Right-click the writing access area and select **Properties**.
3. Select the **Users** tab and click **Connect**.

4. In the query tool, click the arrow in the first field and select the target (*Access Area Member*, Person or Person Group).

   Access area member groups all persons and person groups belonging to an access area. This area defines objects that can be accessed by the person or person group.

5. (optional) In the second field, enter the character string to be queried.
6. Click **Find** .
7. In the results list, select the required access area member and click **Connect**.

   ☛ *Press the [CTRL] key to select several members simultaneously.*

   The person or person group you have connected appears in the list of access area members of the selected writing access area.

Defining a Writing Access Area at Creation

To assign to all objects created by a user a writing access area different from the writing access area of this user, you must associate a writing access area at creation with the user concerned.

To define a writing access area at creation of a user:

1. Open the properties dialog box of the person.
2. Select the **Characteristics** tab.
3. In the **Writing Access Area at Creation** field, select the required writing access.

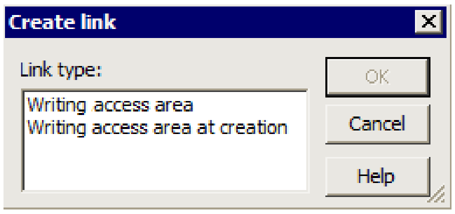
   ☛ *See Writing access area and writing access area at creation.*

To define a writing access area at creation of a user:

1. From **Hopex Administration**, open the writing access diagram.

   ☛ *See Opening the Writing Access Diagram.*

2. If this is not already done, place the person and the writing access area concerned in the diagram.
3. Draw a link between the person and the required writing access area. The **Create Link** dialog box opens.

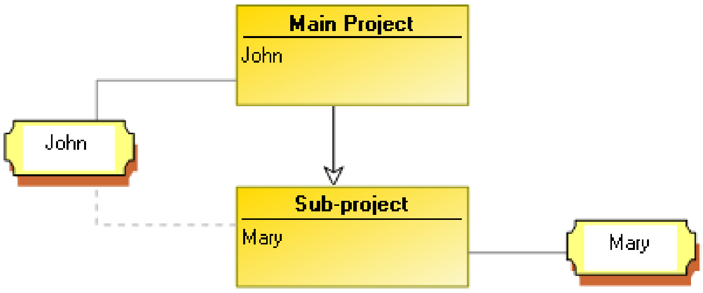


Writing access area
Writing access area at creation

4. Select **Writing Access Area at Creation** and click **OK**.

   ☛ *Value "None" of* ***Writing Access Area at Creation*** *signifies that the user creates objects in the same writing access area as that to which he/she belongs.*

When the writing access area at creation has been created, it is represented by a dotted line link between the person and the writing access area in the writing access diagram.



In the above example, user John has:

- a writing access area of level "main project"
- a writing access area at creation of level "sub-project"

Objects created by John can therefore be modified by Mary.

Modifying Writing Access Areas of Objects

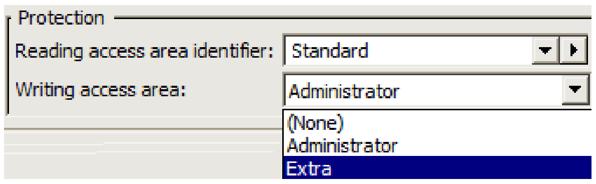
If you have a writing access area of level higher than or equal to that of an object, you can modify the writing access area of this object in the object properties dialog box.

To modify writing access area of an object:

1. Connect to **Hopex**.

   ☛ *See Connecting to Hopex Customizing Desktop.*

2. In your **Hopex** desktop, open the object properties window, select the **General** tab, then the **Administration** sub-tab.
3. In the **Protection** pane, in the **Writing Access Area** field, select a writing access area via the drop-down menu.



Protection
Reading access area identifier: Standard
Writing access area: Administrator

4. Click **OK**.

Modifying Writing Access Areas of an Object Group

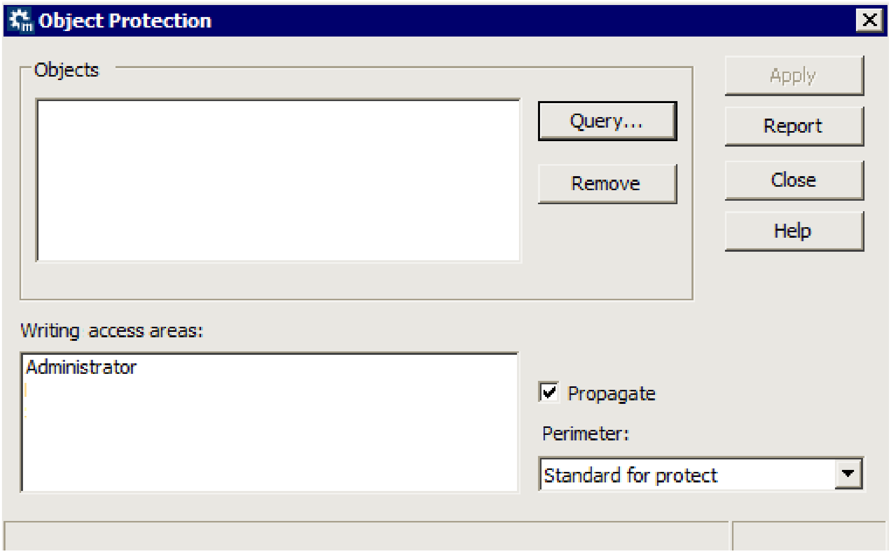
If you have a writing access area of level higher than or equal to that of an object group, you can modify the writing access area of this object group.

To modify the writing access area of an object group:

1. Connect to **Hopex**.

   ☛ *See Connecting to Hopex Customizing Desktop.*

2. In your **Hopex** workspace, select **Tools > Manage > Protect Objects**. The **Protect Objects** dialog box opens.
3. In the **Objects** pane, click **Query** and select the object group.
4. In the **Writing Access Areas** pane, select the writing access area you want to assign to the object group.
5. (Optional) Select **Propagate** if you want the writing access area to be propagated to all dependent objects of the object group, as a function of the perimeter selected.



Writing access areas: Administrator

[✓] Propagate

Perimeter: Standard for protect

6. Click **Apply**.
   Object protection is applied.
7. Click **Report** to check if a conflict has been encountered at protection propagation in the repository.
   Conflicts may arise, especially when **Propagate** is selected and that some children objects already have a defined writing access area (through a direct link).
   To solve this conflict, on each object in conflict, you must manually delete the link between the object and the writing acces area, to define the writing access area you want to keep it.

   ☛ *See Associating Objects with Writing Access Areas.*

Deleting Writing Access Areas

To delete a writing access area:

1. From **Hopex Administration**, open the writing access diagram.

   ☛ *See Opening the Writing Access Diagram.*

2. Open the properties dialog box of the writing access area concerned.
3. Select the **Users** tab and disconnect all **Access Area Members** of the writing access area, and connect these **Access Area Members** to another writing access area.
4. Delete the writing access area.

The writing access areas dependent on the deleted writing access area are, after updating, no longer attached to the writing access areas tree. It is therefore preferable to first delete their links with the obsolete writing access area and attach them to a writing access area that will be retained.

Objects which had this writing access area can be protected with another writing access area. Otherwise, they are considered as being protected at the highest level, with "Administrator" writing access area level.

☛ *For more details on protection of objects, see Protecting Objects.*

Propagating Object Writing Access Areas to Child Objects

The **Administration** navigation window of the **Hopex** workspace allows:

- access to writing access areas
- simple automation of writing access area propagation to connected and child objects.

  ☛ *See Associating Objects with Writing Access Areas.*
- connecting an object to a writing access area

You can propagate a writing access area from all objects connected to dependent objects, for a repository of the environment.

☛ *This action can take some considerable time, depending on repository size.*

To propagate a writing access area:

1. Connect to **Hopex**.

   ☛ *See Connecting to Hopex Customizing Desktop.*
2. In your **Hopex** workspace menu bar, select **View > Navigation Windows > Administration**.
3. In the **Navigation** tab, expand the **Writing Access Area** folder.
4. Right-click the writing access area to be propagated and select **Propagation of writing access area to associated occurrences**.
5. Click **Yes** to confirm.

Associating Objects with Writing Access Areas

The **Administration** navigation window of the **Hopex** workspace allows:

- access to writing access areas
- simple automation of writing access area propagation to connected and child objects.
- connecting an object to a writing access area

  ☛ *See Propagating Object Writing Access Areas to Child Objects.*

To connect an object to a writing access area:

1. Connect to **Hopex**.

   ☛ *See Connecting to Hopex Customizing Desktop.*

2. In your **Hopex** workspace menu bar, select **View > Navigation Windows > Administration**.
3. In the **Navigation** tab, expand the **Writing Access Area** folder.
4. Right-click the required writing access area and select **Connect > Object**.
5. In the query dialog box, find the required object and click **OK**.

To display the list of objects associated with a writing access area:

- Right-click the writing access area and select **Objects associated with writing access area**.
  A dialog box displays a list of these objects.

Tips on Using Writing Access Areas

Common data

Bizzdesign recommends that you manage data common to several projects in a specific project. This simplifies control of their evolutions.

Tips

Bizzdesign recommends:

- definition of certain users:
  - with writing access area level higher than projects so as to manage conflicts between projects, for example "Project Management".
  - with writing access area at creation at the level of the project, to avoid creation of objects that cannot be modified by the project.

    ☛ *See Defining a Writing Access Area at Creation.*
- that only the "Administrator" user should be connected to the "Administrator" writing access area.
- that if a person produces on several projects, the person should have one **Hopex** user per project.
  Objects are therefore directly created in the correct owner project, greatly simplifying management.

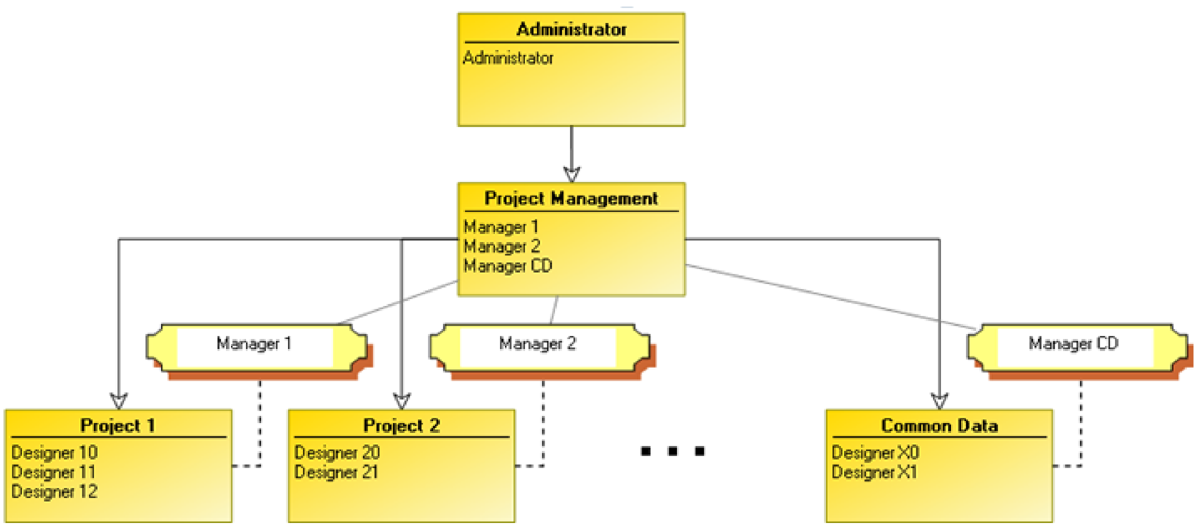
Typical example

The following example presents a typical case of writing access area use:

- Only the Administrator user has "Administrator" writing access area level.
- All managers can modify objects of all projects.
- Objects created by a manager are attached to a dedicated project.

  ☛ *Manager 1 can modify objects of all projects, by default the objects he/she creates are in project 1.*

- Data common to different projects ("Common Data") is managed in a dedicated project with a specific writing access area.



CUSTOMIZING WRITING ACCESS AREA MANAGEMENT

This section describes how to use and customize management of writing access areas:

Calculated Writing Access Area

As standard, the writing access area of an object is stored in the "_Authorization" MetaAttribute and takes the value of the writing access area absolute identifier. It is assigned at creation and you can modify it.

You can install up calculated writing access area.

For example, you can deduce the writing access area of an operation from that of the process on which it depends. You need only change the writing access area of a process, and those of the dependent operations will automatically adapt.

In this case, watch performance.

To customize the writing access area of an object:

- Replace the "_Authorization" MetaAttribute (which carries the object writing access area) by a calculated MetaAttribute.

Calculated MetaAttribute

A calculated MetaAttribute is a software device enabling deduction of the MetaAttribute value of an object as a function of data around the object or dependent on other sources (system, current user, etc.).

Hopex uses a set of "conventional" MetaAttributes (including the writing access area) that do not require metamodel definition.

A substitution device is available in **Hopex**; it enables replacement of an implicit MetaAttribute by another for a MetaClass.

This device is required when you need to alter behavior of an existing MetaAttribute by implementing a calculated MetaAttribute.

To customize writing access area of an object, you must:

1. Create a MetaAttribute with characteristics close to those of the "_Authorization" MetaAttribute.

2. Substitute the "_HexaIdAbs" value of the new MetaAttribute by the "_HexaIdAbs" value of the "_Authorization" MetaAttribute.
3. Calculate the writing access area.

Installing a Writing Access Diagram

To install a writing access area diagram in an environment already in production:

1. From **Hopex Administration**, open the writing access diagram.

   ☛ *See Opening the Writing Access Diagram.*

2. Ask users who have a private workspace in progress to close this.
3. Create an "Old Data" writing access area

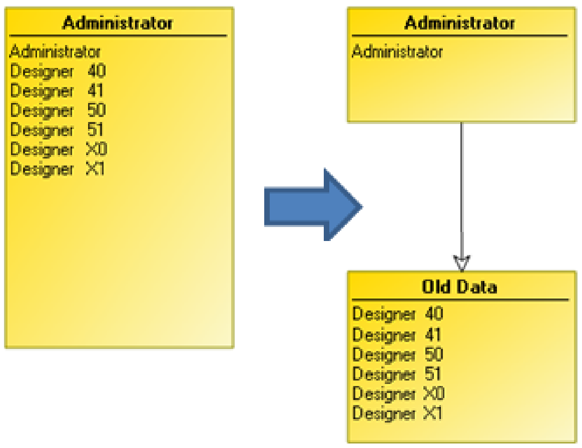
   ☛ *See Creating a Writing Access Area.*

4. Attach all users (except the "Administrator" user) and all objects of all repositories to the "Old Data" writing access area.

   ☛ *See Defining Writing Access Area Persons or Person Groups.*

   ☛ *See Associating Objects with Writing Access Areas.*

   Users may resume working.

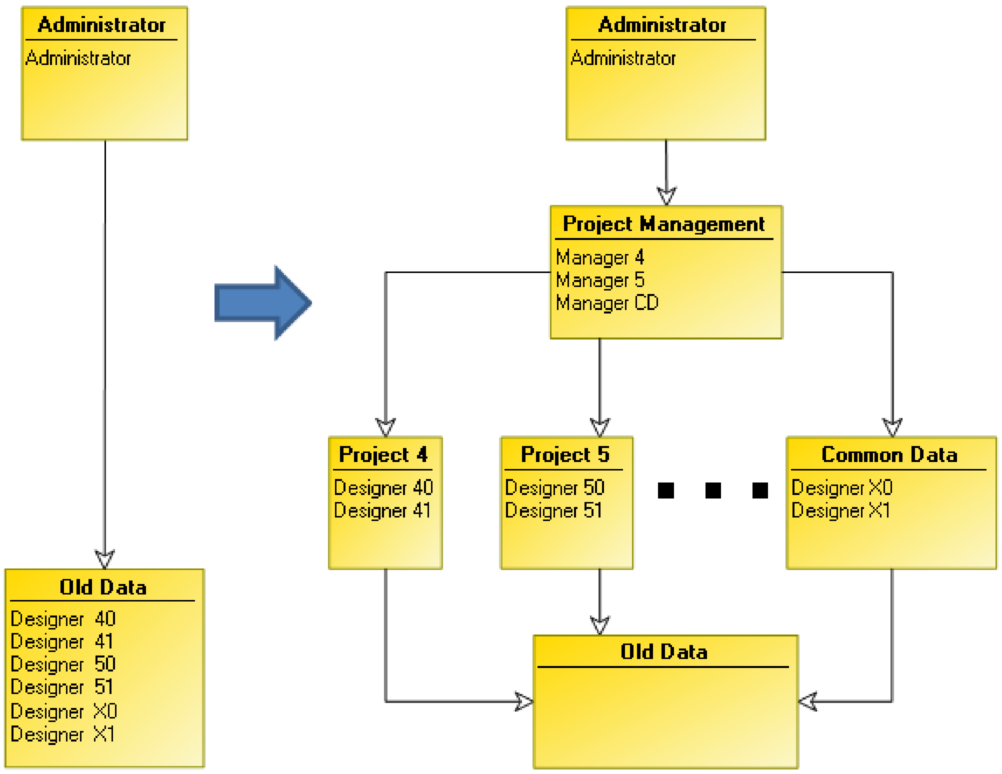


5. Create new writing access areas according to projects.
6. Distribute users between these writing access areas.
7. Distribute objects between these projects/writing access areas for all repositories of the environment.

   ☛ *See Associating Objects with Writing Access Areas.*

   Until this distribution is completed, projects can interfere with each other, since they have rights to modify objects created before distribution.

8. (Optional) When all objects of all repositories have been distributed, you can delete the "Old Data" writing access area.



☛ *If the environment is new and there is no data to be distributed between the new writing access areas, you do not need to draw the diagram with the "Old Data" writing access area.*

Locking Validated Objects

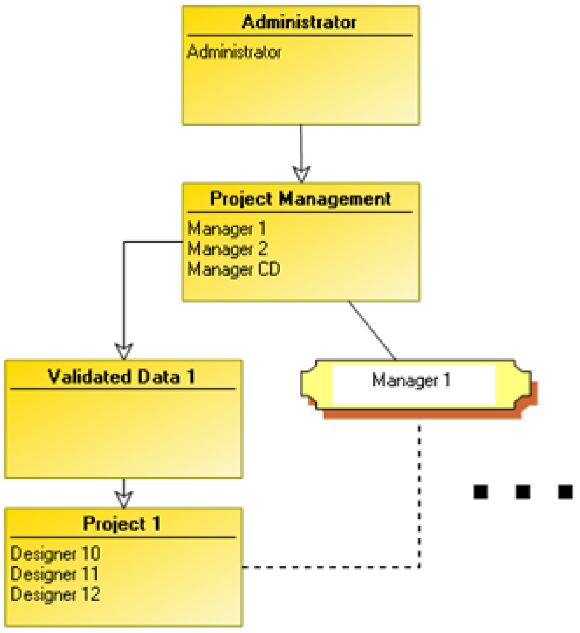
When objects have been validated, you can configure the writing access diagram so that these objects cannot be modified.

To lock objects:

1. From **Hopex Administration**, open the writing access diagram.

   ☛ *See Opening the Writing Access Diagram.*

2. Between the writing access area "Project Management" and that of the project, insert a writing access area dedicated to validated data of the project.



3. When data is validated, modify its writing access area from "Project" level to the higher level "Validated Data" writing access area.

   ☛ *See Modifying Writing Access Areas of Objects.*

4. (Optional) If validated data must be modified:
   - it is modified by a user of "Project Management" writing access area level.
   - it is lowered to the "Project" writing access area level.

So the project perimeter is distributed between two writing access areas, but remains perfectly determined.

Merging Two Projects

To merge two projects:

1. From **Hopex Administration**, open the writing access diagram.

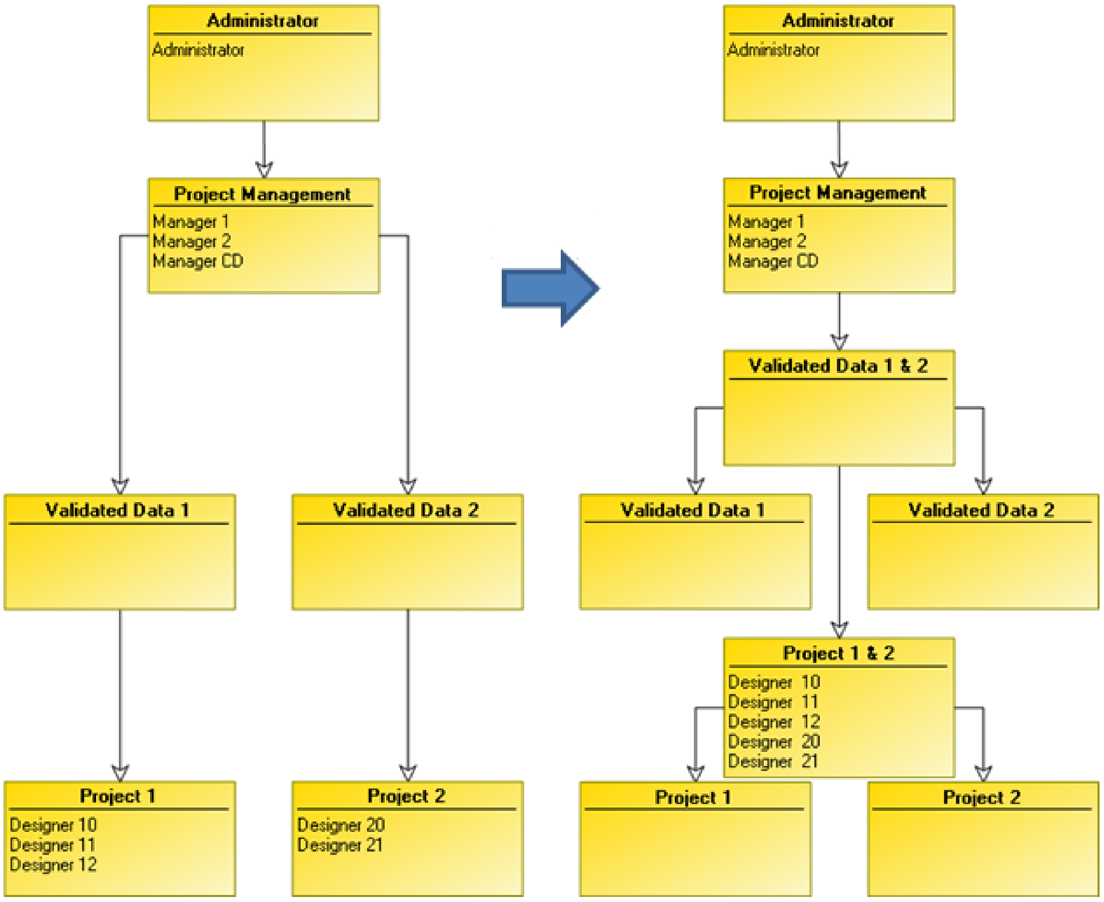
   ☛ *See Opening the Writing Access Diagram.*

2. Create a new writing access area for the new project.

   ☛ *This new writing access area must be of higher level than the writing access area it will replace.*

3. Connect users of merged projects to this new writing access area.

   ☛ *See Defining Writing Access Area Persons or Person Groups.*



Splitting a Project

To split a project:

1. From **Hopex Administration**, open the writing access diagram.

   ☛ *See Opening the Writing Access Diagram.*

2. Create writing access areas connected to new projects.
   These writing access areas must be of must be of higher level than the writing access areas they will replace.

   ☛ *See Creating a Writing Access Area.*

3. Ask users who have a private workspace in progress to close this.
4. Distribute users between these writing access areas.

   ☛ *See Defining Writing Access Area Persons or Person Groups.*

   ☛ *Users may resume working.*

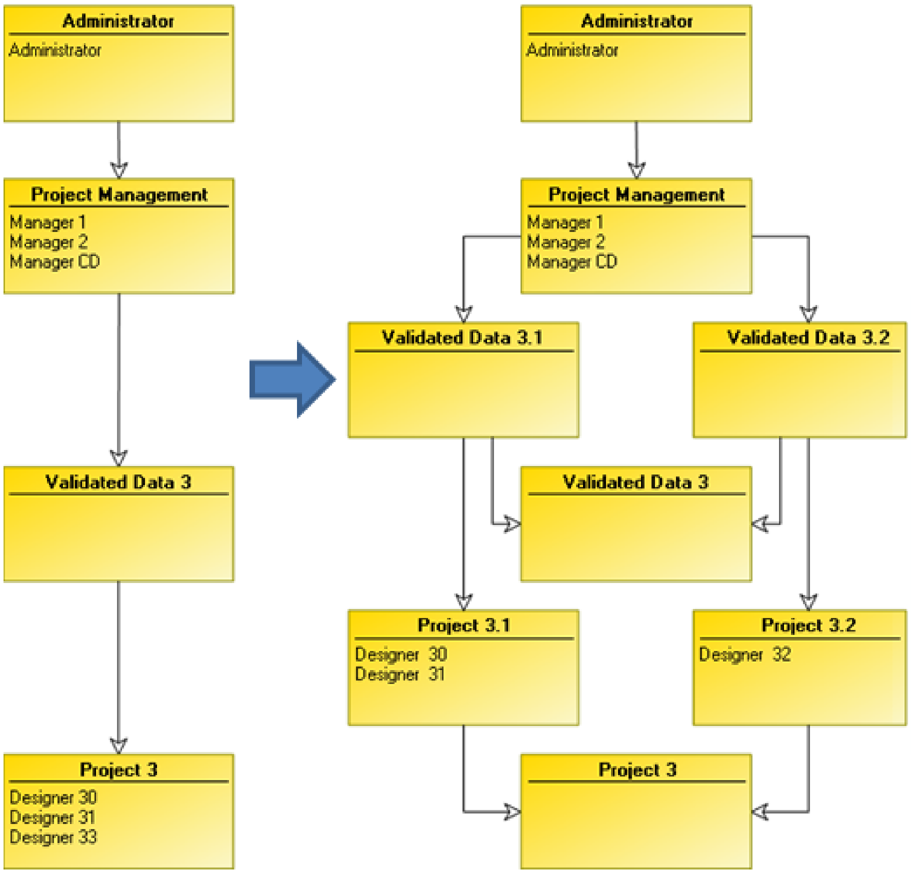
5. Distribute the objects between the new projects.

☛ *See Associating Objects with Writing Access Areas.*

Until this distribution is completed, projects can interfere with each other, since they do not yet have rights to modify objects created before distribution.

6. (Optional) Delete the old writing access areas.

☛ *(Optional) When all objects of all repositories have been distributed, you can delete old writing access areas.*



MANAGING USERS FROM THE WRITING ACCESS DIAGRAM

This section presents how to:

Creating Persons with Writing Access Areas

At creation, the user is not connected to any writing access area. To implement protection, the person should be connected to a writing access area by creation of a link between person and writing access area.

To create a person with a writing access area, from the writing access diagram:

1. From **Hopex Administration**, open the writing access diagram.

   ☛ *See Opening the Writing Access Diagram.*

2. In the writing access diagram, right-click a writing access area and select **Properties**.
3. In the **Users** tab, click the **New** button ⊕ .
   A selection dialog box appears.
4. In the **MetaClass** field, select **Person**.
   The **Creation of Person** dialog box opens.
5. Follow the procedure Defining a Person.
   The person is created and connected to the selected writing access area.

   ☛ *A person depends on a single writing access area.*

Creating Person Group with Writing Access Areas

To create a person group with a writing access area, from the writing access diagram:

1. From **Hopex Administration**, open the writing access diagram.

   ☛ *See Opening the Writing Access Diagram.*

2. In the writing access diagram, right-click a writing access area and select **Properties**.
3. In the **Users** tab, click the **New** button ⊕ .
   A selection dialog box appears.
4. In the **MetaClass** field, select **Person Group**.
   A new person group appears in the list of access area members. The new group is connected to the selected writing access area.

5. (optional) Modify the **Short Name** of the new person group.

Managing Users from the Writing Access Diagram

User access rights to repositories and functions can be restricted by the administrator. You can carry out this modification user by user, or on all users simultaneously.

☛ *To manage user access rights, the **Hopex Power Supervisor** technical module is required.*

To manage users from the writing access diagram:

1. From **Hopex Administration**, open the writing access diagram.

   ☛ *See Opening the Writing Access Diagram.*

2. From the writing access diagram, select **Diagram > Described Object > Manage Users**.
   The **Users** administration dialog box opens.

Compiling the Writing Access Diagram

Changes to the writing access diagram only take effect after it has been compiled.

To compile the diagram:

- Select **Diagram > Described Object > Compile Writing Access Diagram**.

If the diagram has been modified, it is automatically compiled at closing. This allows you to check the validity of the user diagram.

When modifying the writing access diagram, so as to warn of rejects due, for example, to authorization restrictions or deletions before compilation it is recommended that:

- all changes made on the user workstations should be uploaded to the administrator workstation, or
- all private workspaces dispatched.

When compilation is complete, a message indicates whether the operation was successful or whether the diagram contains errors. The most frequent errors are:

- A writing access area (other than '"Administrator") is not attached to any other.
- A person or person group is not attached to any authorization.

Transferring the Writing Access Diagram

On workstations where the network is not available, when the user diagram has been updated and compiled it must be exported to the administrator workstation and imported on user workstations. You must also do this if the same diagram is to be used in several different environments.

To run export:

- Select **Diagram > Described Object > Export Writing Access Diagram**.

Import on user workstations is carried out in the normal way.

Customizing Writing Access Diagram Display

You can customize writing access diagram display:

- diagram structure representation
- display of persons connected to a writing access area

Customizing Diagram Structure Representation

You can customize representation of the structure of the writing access diagram using the drawing reorganization function.

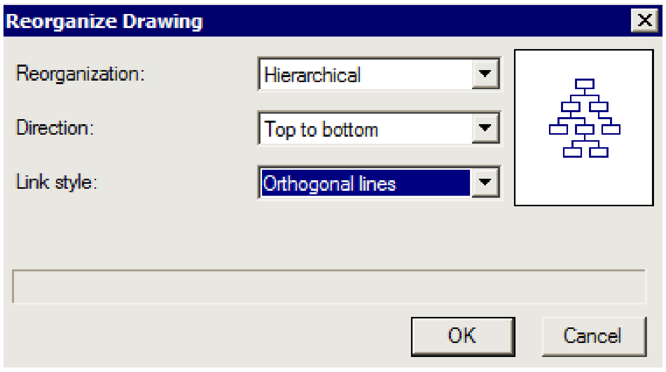
☺ *The automatic drawing reorganization functionality is automatically activated on loading a diagram that does not yet include a drawing.*

To modify organization of an existing drawing:

1. From **Hopex Administration**, open the writing access diagram.

   ☛ *See Opening the Writing Access Diagram.*

2. Select **Drawing > Reorganize Drawing**.
3. Select the desired reorganization mode, the direction and the style of links in the diagram.


Reorganize Drawing
Reorganization: Hierarchical
Direction: Top to bottom
Link style: Orthogonal lines

☺ *The miniature image alongside the reorganization options gives you a view of each type of reorganization.*

4. Click **OK** to apply the modifications.

Customizing Writing Access Area Display

You can show or hide persons, person groups, objects connected to a writing access area.

To define writing access area display:

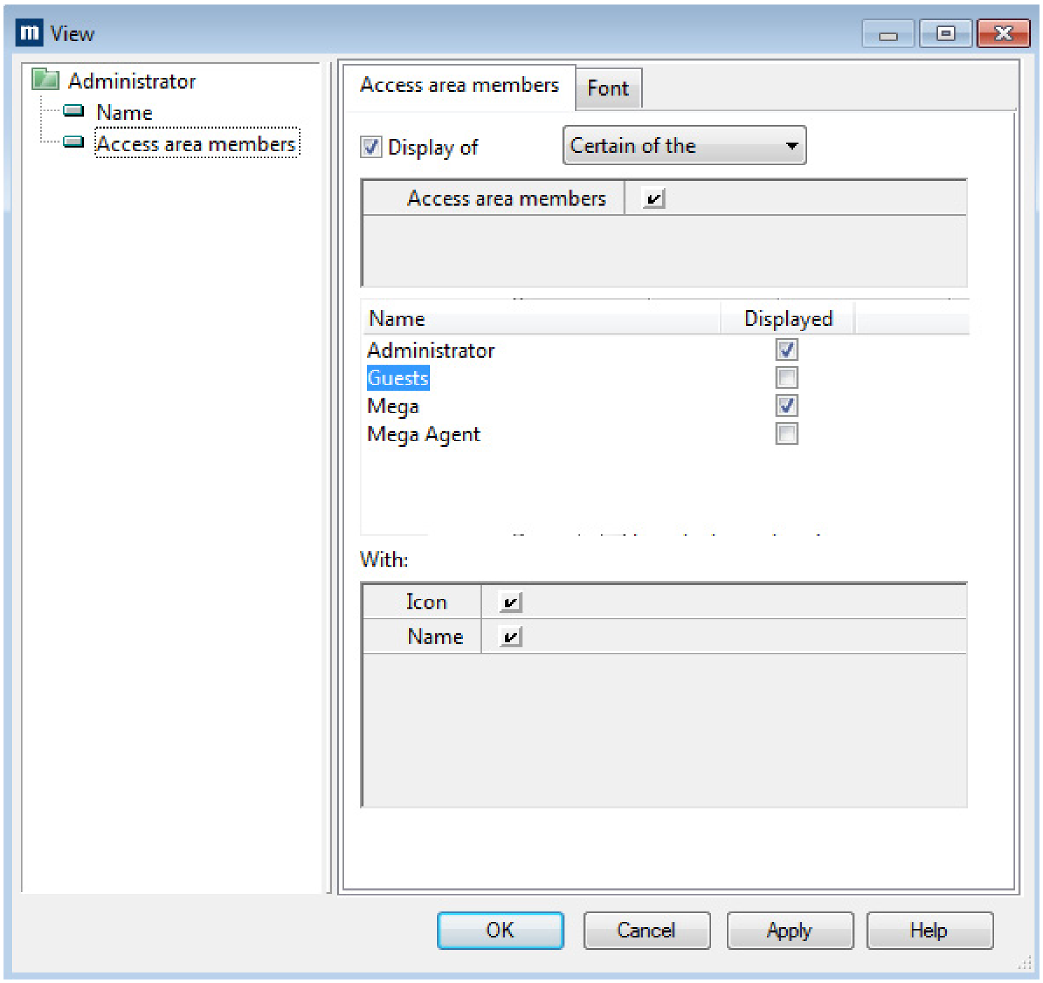
1. From **Hopex Administration**, open the writing access diagram.

   ☛ *See Opening the Writing Access Diagram.*

2. Right-click the writing access area concerned and select **Shapes and Details**.
   The **View** dialog box opens.
3. In the tree on the left, select **Access Area Members**.
4. In the pane on the right, select **Access Area Members**.
5. Select the **Display of** option, then click the arrow and select **Certain of the** access area members.

   ☛ *To display all or none of the access area members, select **All the** or **None of the**.*

6. Select the persons you want to see displayed.



Display of [✓] Certain of the

Access area members	✓

Name	Displayed
Administrator	✓
Guests	
Mega	✓
Mega Agent	

With:

Icon	✓
Name	✓

7. Click **OK**.

DATA READING ACCESS

The following points are covered here:

INTRODUCTION

☛ *Managing reading access areas is only available with the **Hopex Power Supervisor** technical module.*

The following points are detailed here:

The Need to Manage Sensitive Data

Certain modeling projects may be confidential or contain confidential or sensitive data: costs, risks, controls.

The **Hopex** administrator may therefore need to mask objects corresponding to confidential or sensitive data.

These objects must be visible only to authorized users.

To meet these requirements concerning data confidentiality, **Hopex** offers functionalities for implementing consistent and effective confidentiality policies.

This features applies to data in the data repositories only (i.e.: confidentiality management does not apply to SystemDb repository data).

General Concepts

To implement a data confidentiality policy, objects must be organized in distinct sets. Each set of objects is a ***reading access areas***.

A user is a person with a login.

A person can belong to a group. A user group is a group of persons with a login.

Each user or group of users is associated with a reading access area. It is the person or person group that carries the reading access area.

The reading access area to which the person or person group belongs determines the objects that the user or group of users can see. A user or user group can only see objects located in his/her own or lower confidentiality areas.

With definition of reading access areas, hidden objects are inaccessible. This concept differs from that of the filter, which hides occurrences of MetaClasses so as not to disturb the final view of the user, see Permissions to Access UIs.

Activating Data Reading Access Management

When you activate reading access management, confidential data is visible only to authorized users. Before activating reading access management, **Bizzdesign** recommends that you familiarize yourself with reading access management using **Bizzdesign**.

☛ *For more details on confidential data, see Confidential or Sensitive Object Behavior.*

To manage confidential or sensitive data , you must first activate data reading access area management.

To activate data reading access management:

1. Connect to **Hopex Administration** and select the environment.

   ☛ *See Connecting to an Environment.*

2. In the **User Management** folder, right-click **Data Reading Access** and select **Activate Management**.

   A message warns you that activation of reading access management is irreversible.

3. Carefully read this warning message, then click **Yes** if you wish to activate data reading access management.

Consulting Environment Reading Access Information

When working in **Hopex** you can check:

- if reading access management is activated in your environment or not.
- the reading access area to which the connected user belongs.

To consult reading access information of your current environment:

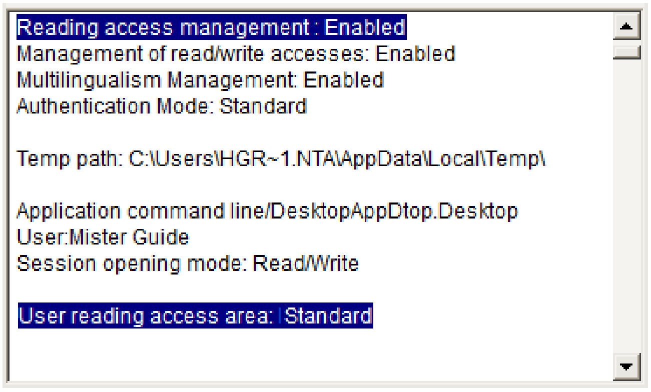
1. Connect to **Hopex**.

   ☛ *See Connecting to Hopex Customizing Desktop.*

2. In the **Hopex** menu bar, select **Help > About Hopex**.
   The **About Hopex** dialog box appears.
3. Click **System Information**.
4. In the menu bar of the **System Properties** dialog box, select **System Info > Edit**.
   The Megasys.txt text file opens.

At the beginning of the file you can consult the properties of:

- ***Reading access management***
- ***Reading access area of the user***



Reading access management : Enabled
Management of read/write accesses: Enabled
Multilingualism Management: Enabled
Authentication Mode: Standard

Temp path: C:\Users\HGR~1.NTA\AppData\Local\Temp\

Application command line/DesktopAppDtop.Desktop
User:Mister Guide
Session opening mode: Read/Write

User reading access area: Standard

Managing Reading Access in Hopex

You can set up and manage data reading access in **Hopex** in two ways:

Reading access area matrix method:

1. Create the different user reading access areas you require.
2. Distribute persons or person groups in user reading access areas.
3. Distribute objects in object reading access areas.
4. Associate user reading access areas with object reading access areas.

☛ *For more details, see Reading Access Area Matrix.*

Reading access diagram method:

1. Define organization and hierarchy of the different reading access areas you require.
2. Create and organize these users in a reading access diagram.
3. Associate persons or person groups with different reading access areas. Objects created by users are then distributed in the user reading access area.

☛ *For more details, see Reading Access Diagram.*

Compiling the Reading Access Diagram

Running the reading access diagram compilation ensures consistent **Hopex** behavior with diagram declarations.

If the diagram is not compiled, there is a risk that certain users will be able to see objects that are normally hidden.

To compile the reading access diagram:

1. From **Hopex Administration**, connect to the environment.

☛ *See Connecting to an Environment.*

2. Expand the **User management** folder of the environment.
3. Right-click the user **Data Reading Access** folder and select **Compile**.
   On completion of compilation, a message indicates the result of the operation.

Reading Access Area Matrix

The reading access area matrix enables organization of user groups with object groups. Only a user with administrator profile connected to the maximum reading access area can configure reading access areas.

In the reading access area matrix, you can create two types of reading access areas:

- an **object reading access area** , grouping only **Hopex** objects.
- a **user reading access area** , grouping only persons or person groups

  The user reading access area corresponds to the view the person or person group has of the repository: it defines objects that can be accessed by the person or person group.

  To ensure coherence of the reading access diagram, if you begin managing reading access of your data from the reading access area matrix, you must continue to manage reading access from this matrix.

You can create links between these two types of reading access area.

Accessing the Reading Access Area Matrix

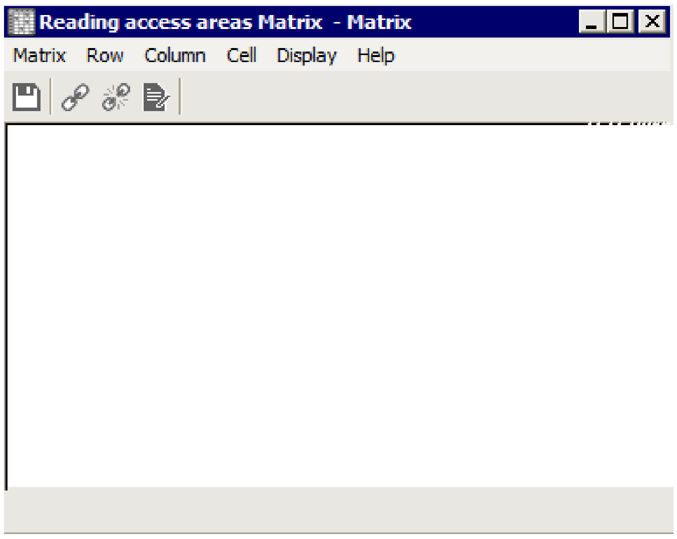
To access the reading access area matrix:

1. From **Hopex Administration**, connect to the environment.

   ☛ *See Connecting to an Environment.*

2. Expand the **User management** folder of the environment.

3. Right-click the user **Data Reading Access** folder and select **Reading Access Matrix**.
   An empty reading access area matrix appears.



Adding an Object Reading Access Area

To add an object reading access area in the matrix:

1. Open the reading access area matrix.
   ☛ *See Accessing the Reading Access Area Matrix.*
2. Select **Row > Create**.
3. In the creation window, enter the name of the object reading access area and click **Finish**.

Adding a User Reading Access Area

To add an user reading access area in the matrix:

1. Open the reading access area matrix.
   ☛ *See Accessing the Reading Access Area Matrix.*
2. Select **Column > Create**.
3. In the creation window, enter the name of the user reading access area and click **Finish**.

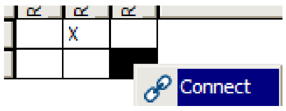
Associating User Reading Access Areas with Object Reading Access Areas

To associate a user reading access area with an object reading access area

1. Open the reading access area matrix.

   ☛ *See Accessing the Reading Access Area Matrix.*

2. In the reading access area matrix, right-click the cell at the intersection of the user reading access area and the object reading access area and select **Connect**.



A cross represents the association between the two selected reading access areas. The corresponding links are automatically drawn in the reading access diagram.

If you begin management of reading access of your data from the reading access area matrix, you must continue management of reading access from this matrix. Do not manually modify links created automatically in the reading access diagram; you may invalidate the diagram.

☛ *For more details on the reading access diagram, see Reading Access Diagram.*

Associating Users with User Reading Access Areas

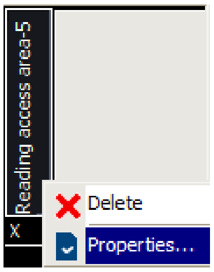
In the case of the reading access area matrix, to associate a user (or user group) with a reading access area:

1. From **Hopex Administration**, connect to the environment.

   ☛ *See Connecting to an Environment.*

2. Expand the **User management** folder of the environment.
3. Right-click the user **Data Reading Access** folder and select **Reading Access Matrix**.
   The reading access area matrix appears.

4. Right-click the user reading access area concerned and select **Properties**.



The user reading access area properties window appears.

5. In the **Persons** tab, click **Connect**.

   ☛ *If you want to create a new Person (or new person group) and associate it with the reading access area, click **New**.*

6. In the query tool, click the arrow in the first field and select **Person** (or **Person Group**).

   ☛ *If you want to select persons and person groups, select Access Area Member.*

7. (optional) In the second field, enter the character string to be queried.
8. Click **Find** .
9. In the query result list, select the person (or person group) required and click **Connect**.

   ☛ *Press the [CTRL] key to select several persons and/or person groups simultaneously.*

   The user (or user group) you have connected appears in the list of users in the selected reading access area.

Reading Access Diagram

The *reading access diagram* enables organization of the repository by sets of objects, unlike the writing access diagram which enables organization by work groups.

The reading access diagram enables definition of reading access areas and their hierarchical organization. This diagram also enables creation of users and their association with reading access areas.

The reading access diagram can be opened only by an Administrator profile user connected to the maximum reading access area.

☛ *The reading access diagram feature is accessible only if you have the **Hopex Power Supervisor** technical module.*

If you begin management of reading access of your data from the reading access area matrix, you must continue to manage reading access from this matrix. Do not manually modify links created automatically in the reading access diagram, you might invalidate the diagram.

Warning: on exiting the reading access diagram, if a message indicates that the diagram is incorrect, the diagram is not compiled and reading access management does not operate. Bizzdesign recommends that you correct the error that prevents diagram compilation.

This section covers the following points:

Reading Access Diagram Operating

The reading access diagram implements reading access areas of **General** type. These areas can group both objects and persons and/or person groups.

Reading access areas are organized hierarchically. **Bizzdesign** provides two extreme reading access areas:

- **Maximum Reading Access**, the highest reading access level.
- **Standard**, the lowest reading access level.

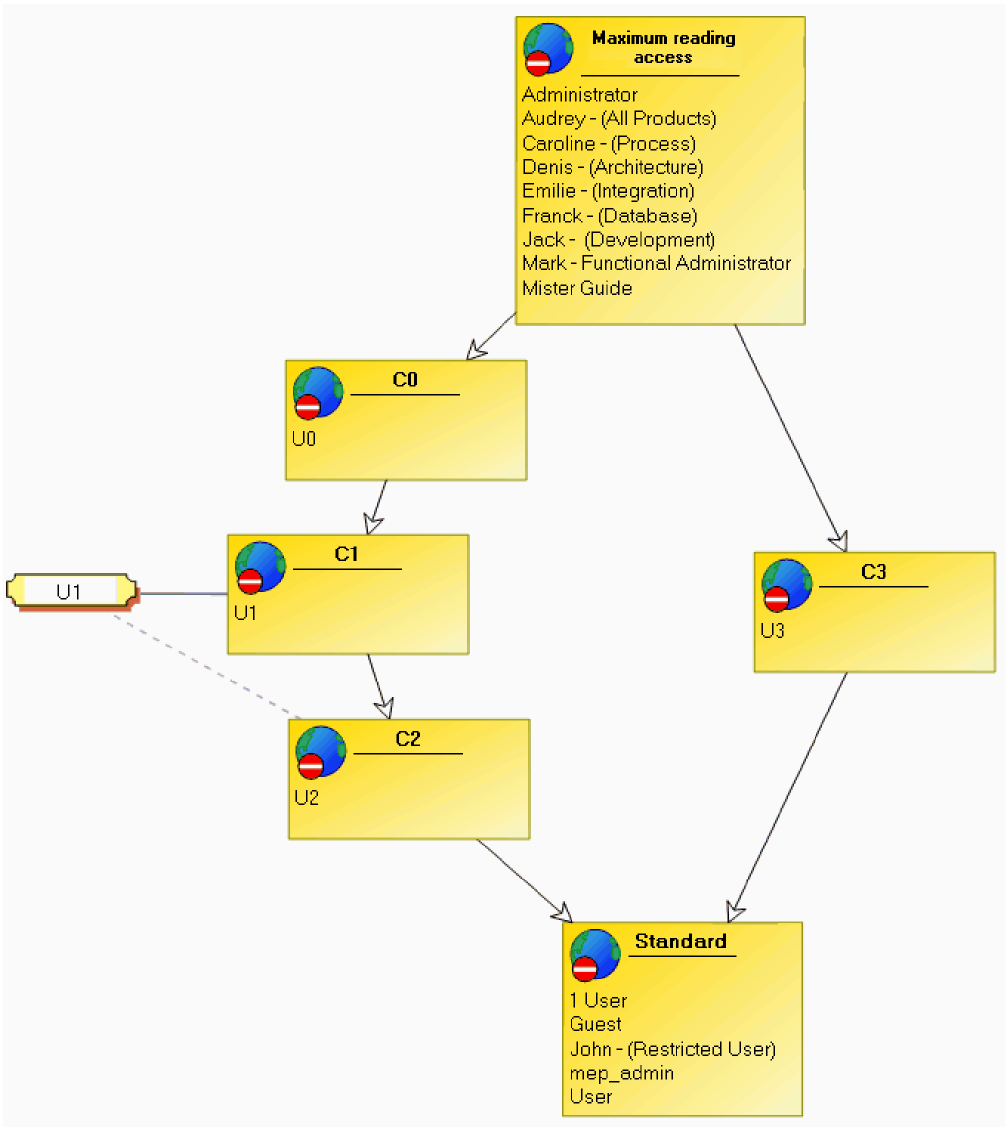
Each object belongs to a reading access area (**Standard** by default).

Each person or person group is connected to one of the reading access areas. The persons or person groups that are connected to:

- **Maximum reading access** sees all repository data.
- A created reading access area sees all data of this area, as well as that of lower level reading access areas.
- **Standard** sees only non-confidential data of the repository.

For example, a user U1 connected to a reading access area C1 sees all objects that belong to:

- his/her reading access area (C1)
- lower level reading access areas (C2 and Standard).



In the preceding reading access diagram, user U1:

- is connected to reading access area C1
- to a reading access area at creation C2.

When user U1 creates an occurrence of a MetaClass:

- if sensitive (high sensitivity), this belongs to reading access area C1.
- if non-sensitive (standard sensitivity), this belongs to reading access area C2.

☛ *For more details on MetaClass sensitivity, see Managing MetaClass Sensitivity and Reading Access Areas.*

If a user does not have a reading access at creation, any occurrences of a non-sensitive MetaClass he/she creates belong to the standard reading access area.

However, Web sites, reports, reports (MS Word) Report DataSets are always created at the reading access level of the user. This ensures confidentiality of the information they may contain.

Users connected to reading access area C3 cannot see objects belonging to reading access areas C0, C1, and C2, since area C3 does not belong to the same hierarchical branch as areas C0, C1, and C2.

Activating the reading access diagram

To be able to access the reading access diagram, you must first activate the reading access diagram option.

☛ *By default, only the reading access area matrix is accessible.*

To activate the reading access diagram option:

1. From **Hopex Administration**, connect to the environment.

   ☛ *See Connecting to an Environment.*
2. Right-click the environment and select **Options > Modify**.
3. In the **Options** tree, select **Compatibility > Windows Front-End > Administration**.
4. In the right pane, select the **Activate reading access diagram** option.
5. Click **OK**.
   The reading access diagram option is activated. The reading access diagram is accessible.

Prohibiting Reading Access Diagram Modification

By default, reading access diagram modification is authorized.

To prohibit reading access diagram modification:

1. From **Hopex Administration**, connect to the environment.

   ☛ *See Connecting to an Environment.*
2. Right-click the environment and select **Options > Modify**.
3. In the **Options** tree, select **Compatibility > Windows Front-End > Administration**.

4. In the right pane, for the **Authorize modification of writing access and reading access diagrams** option select "Prohibit".
5. Click **OK**.
   Reading diagram modification is prohibited.

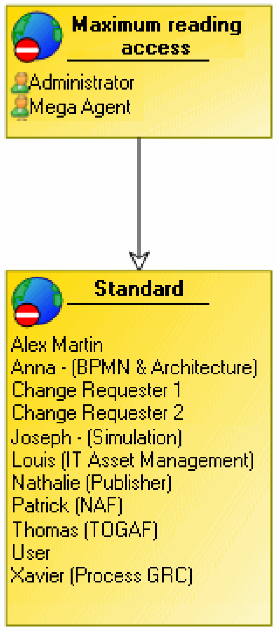
Opening the reading access diagram (Windows Front-End)

To access the reading access diagram:

1. From **Hopex Administration**, connect to the environment.

   ☛ *See Connecting to an Environment.*

2. Expand the **User management** folder of the environment.
3. Right-click the **Data Reading Access** folder and select **Open Diagram**.
   The reading access diagram appears.



By default, the reading access diagram contains two reading access areas:

- **Maximum Reading Access** is the highest reading access area level. Users connected to this area can see all objects in the repository.
- **Standard** is the lowest reading access area level.

  ☛ *There can only be one maximum level and one minimum level (standard) reading access area in the diagram.*

Organizing Reading Access Areas

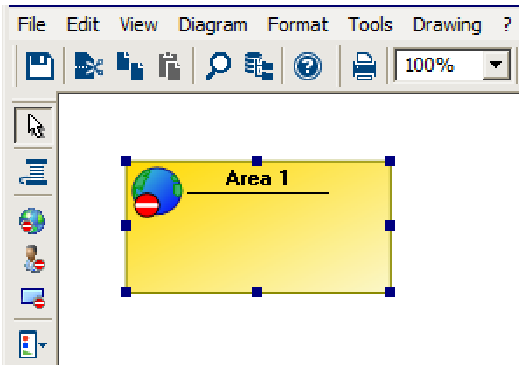
Creating reading access areas

To create a *reading access area* in the diagram:

1. Open the reading access diagram.

   ☛ *See Opening the reading access diagram (Windows Front-End).*

2. In the diagram insert toolbar, click **General Reading Access Area** , then click in the diagram.
3. In the creation wizard dialog box that appears, enter the name of the reading access area and click **Create**.
   The creation wizard allows you to modify the reading access area type if necessary.
4. Click **Finish**.
   The general reading access area appears in the diagram.



Connecting two reading access areas

Reading access areas must be hierarchically interlinked. Except for **Maximum Reading Access** and **Standard** reading access areas, each reading access area must be connected to a lower level area and a higher level area.

To connect two reading access areas:

1. Open the reading access diagram.

   ☛ *See Opening the reading access diagram (Windows Front-End).*

2. In the diagram insert toolbar, click **Link** and draw a link between the two reading access areas (from the higher level reading access area to the lower level reading access area).

Displaying reading access areas associated with a reading access area

To display object (or user) reading access areas associated with a reading access area:

1. Open the reading access diagram.

   ☛ *See Opening the reading access diagram (Windows Front-End).*

2. Open the properties dialog box of the reading access area in question.
3. Select the **Matching Object reading access areas** or **Matching User reading access areas**.
   The associated object reading access areas or user reading access areas are listed.

Adding a User in the Reading Access Diagram

You can add a person or person group in the reading access diagram.

Adding a person in the reading access diagram

To add a person in the reading access diagram:

1. Open the reading access diagram.

   ☛ *See Opening the reading access diagram (Windows Front-End).*

2. In the diagram insert toolbar, click **Person**, then click in the diagram.

   ☛ *If the **Person** icon is not present in the insert toolbar, add it from the **View > Views and Details** menu.*

   The **Add Person** dialog box appears.
3. In the **Name** field, click the arrow to find the person then click **Connect**.

   ☛ *To add a new person, in the **Name** field, enter the name of the person then click **Create**. Also create the login of the person.*

Adding a person group in the reading access diagram

To add a person group in the reading access diagram:

1. Open the reading access diagram.

   ☛ *See Opening the reading access diagram (Windows Front-End).*

2. In the diagram insert toolbar, click **Person Group**, then click in the diagram.

   ☛ *If the **Person Group** icon is not present in the insert toolbar, add it from the **View > Views and Details** menu.*

   The **Add Person Group** dialog box appears.
3. In the **Name** field, click the arrow to find the person group then click **Connect**.

   ☛ *To add a new person group, in the **Name** field, enter the name of the person group then click **Create**. Also create the login of the person group.*

Connecting Users to Reading Access Areas

A user can:

- be connected to a ***reading access area***
  This area defines the view the user has of the repository and the objects the user can access.
- have a ***reading access area at creation***
  Occurrences created by the user belong to this reading access area at creation.

> ☛ *If a user does not have a reading access area at creation, the occurrences he/she creates will belong to the standard reading access area.*

Reading access area of the user

To connect a user to a reading access area:

1. Open the reading access diagram.

   > ☛ *See Opening the reading access diagram (Windows Front-End).*

2. Right-click the reading access area concerned and select **Properties**.
3. Select the **Persons** tab and click **Connect**.

   > ☛ *If you want to create a new user and associate him/her to a reading access area, select **New > Person**.*

   The access area member search dialog box appears.
4. In the first query field, select the type of access area member you wish to connect: **Person** (or **Person Group**).

   > ☛ *If you want to connect persons and person groups, select Access Area Member.*

5. (optional) In the second query field, enter the character string to be queried.
6. Click **Find**.
7. In the result list, select the person (press the [CTRL] key to select several) and click **Connect**.
   The user appears in the reading access area.

Reading access area at creation

You can assign a reading access area at creation to an existing user from:

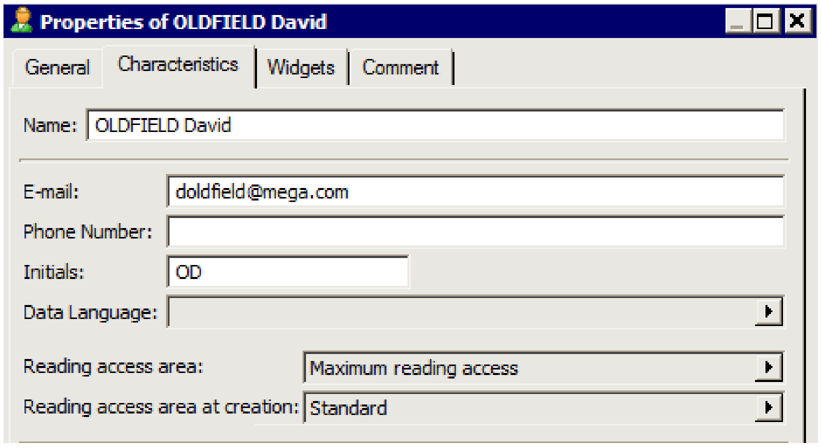
- the reading access diagram
- the **Properties** dialog box of the person

To assign a reading access area at creation to a user from the reading access diagram:

1. Place the user in the reading access diagram.
2. Connect the desired reading access area at creation to the person. This area must be at a level lower than or the same as the reading access area of the user.
   A dialog box asks you to select the type of link to be created: **Access area member** or **Access area member at creation**.
3. Select the link type **Access area at creation member**.
   This area is the reading access area at creation of the user.

To assign a reading access area at creation to a user from the user properties window:

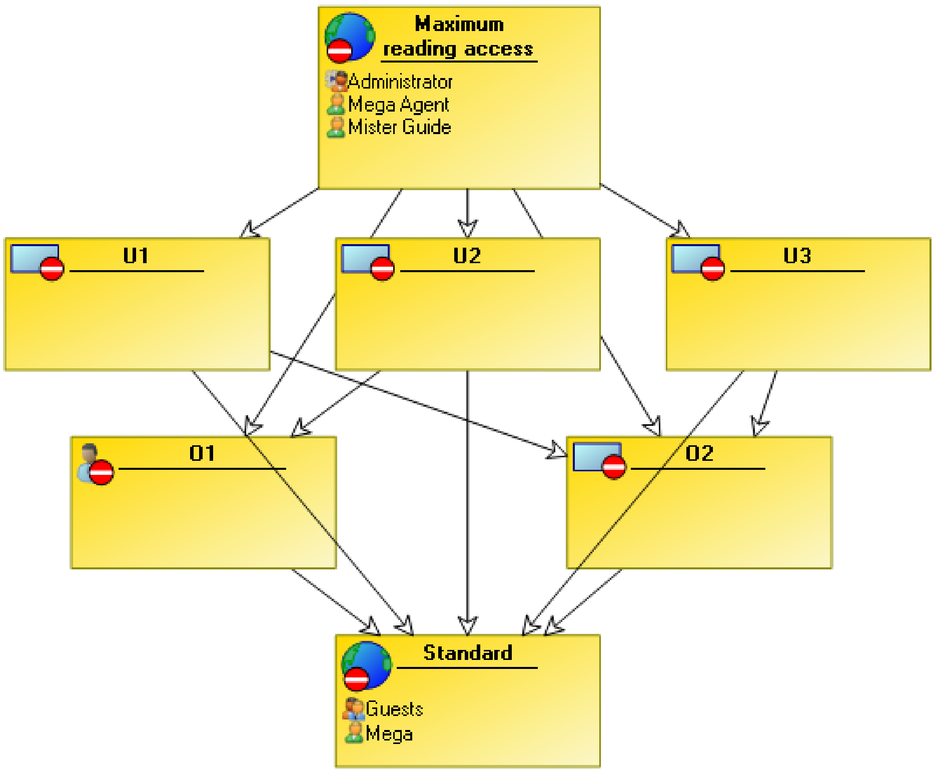
1. Open the **Properties** dialog box of the person and select the **Characteristics** tab, see *Modifying the Properties of a User*.
2. In the **Reading access area at creation** field, select the required value.



Name: OLDFIELD David
E-mail: doldfield@mega.com
Phone Number:
Initials: OD
Data Language:
Reading access area: Maximum reading access
Reading access area at creation: Standard

Consulting Reading Access Diagram Information:

At reading access diagram compilation, user and object are connected to the ***Maximum reading access*** reading access area and the ***Standard*** reading access area.



Open the user reading access area properties dialog box to consult:

- the list of persons connected to the area and to connect new users (**Persons** tab)
- the list of reading access areas for associated objects (**Matching Object Reading Access Areas** tab)

Open the object reading access area dialog box to consult:

- the list of user reading access areas associated with an object reading access area (**Matching User Reading Access Areas** tab)

Customizing Reading Access Area Display

By default, when you create a reading access area, all users belonging to this area are displayed in the diagram reading access area. You can decide to hide certain users in this reading access area.

To define users displayed in their reading access area:

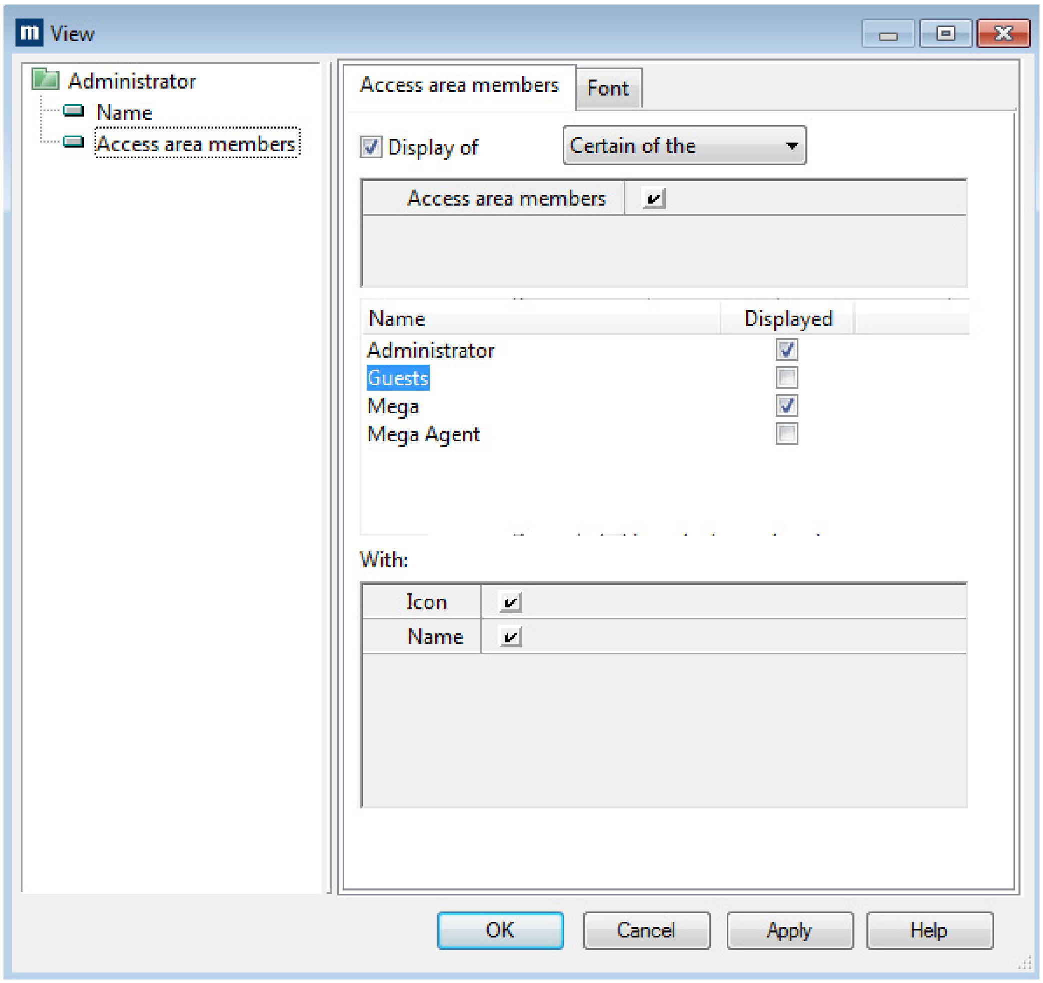
1. From **Hopex Administration**, open the reading access diagram.

   ☛ *See Opening the reading access diagram (Windows Front-End).*

2. Right-click the reading access area concerned and select **Shapes and Details**.
   The **View** dialog box opens.
3. In the tree on the left, select **Access Area Members**.
4. In the pane on the right, select **Access Area Members**.
5. Select the **Display of** option, then in its drop-down menu select **Certain of the** *Access area members*.

   ☛ *To display all or none of the access area members, select* ***All the*** *or* ***None of the****.*

6. Select the persons you want to see displayed.



[✓] Display of: Certain of the

Access area members	✓

Name	Displayed
Administrator	✓
Guests	☐
Mega	✓
Mega Agent	☐

With:

Icon	✓
Name	✓

7. Click **OK**.

CONFIGURING DATA READING ACCESS

In the **Hopex** desktop, the navigation window allows access to certain data reading access functions.

This section presents how to:

- Associating Objects with Reading Access Areas
- Associating user reading access areas with objects
- Propagating Reading Access Areas
- Managing MetaClass Sensitivity and Reading Access Areas
- Confidential or Sensitive Object Behavior
- Modifying Reading Access Areas

Associating Objects with Reading Access Areas

The **Administration** navigation window of the **Hopex** desktop allows access to general and object reading access areas.

The reading access areas tree is used to:

- connect objects to a given reading access area
- quickly propagate the reading access area to child objects

This tree simplifies management of propagation of reading access areas (see Propagating a reading access area from Hopex). Its advantage compared with the classic propagation tool is that the propagation trace is kept thanks to the link.

Connecting objects to object reading access areas

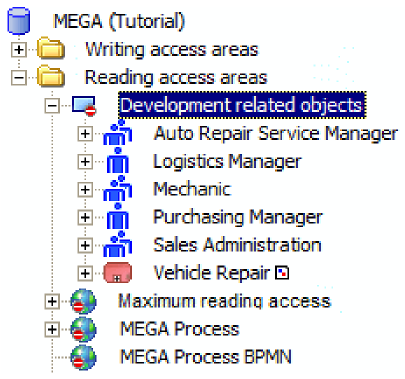
To connect an object to an object reading access area:

1. Connect to **Hopex**.

   ☛ *See Connecting to Hopex Customizing Desktop.*

2. Open the **Administration** navigation window.
3. Expand the **Data Reading Access** folder.
4. Right-click the object reading access area concerned and select **Connect Object**.

5. In the query dialog box, find and select the desired objects and click **OK**.
   The objects are connected to the object reading access area and appear in the tree.



MEGA (Tutorial)
Writing access areas
Reading access areas
Development related objects
Auto Repair Service Manager
Logistics Manager
Mechanic
Purchasing Manager
Sales Administration
Vehicle Repair
Maximum reading access
MEGA Process
MEGA Process BPMN

Disconnecting objects from reading access areas

To disconnect an object connected to a reading access area:

1. Connect to **Hopex**.
   ☛ *See Connecting to Hopex Customizing Desktop.*
2. Open the **Administration** navigation window.
3. Expand the folder of the reading access area concerned.
4. Right-click the object you want to disconnect and select **Disconnect**.
   The object disappears from the tree.

Displaying the list of objects associated with a reading access area

To display the list of objects associated with a reading access area:

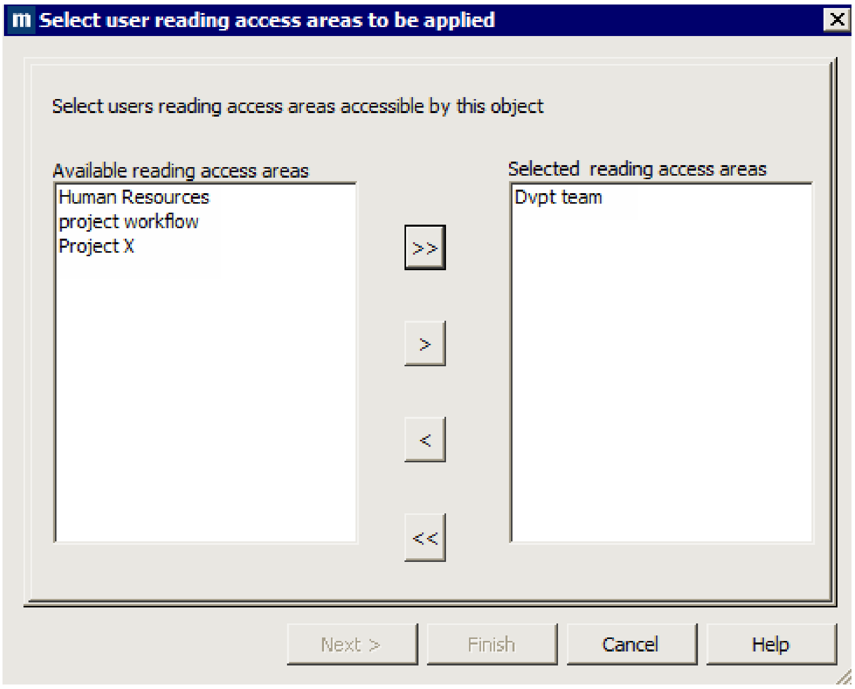
1. Connect to **Hopex**.
   ☛ *See Connecting to Hopex Customizing Desktop.*
2. Open the **Administration** navigation window.
3. Expand the **Data Reading Access** folder.
4. Right-click the reading access area concerned and select **Objects associated with reading access area**.
   A dialog box displays a list of these objects.

Associating user reading access areas with objects

To associate an object with one or several groups of users, proceed as follows:

1. Connect to **Hopex**.
   ☛ *See Connecting to Hopex Customizing Desktop.*
2. Select an object.
3. In the object properties dialog box, select the **General** tab, then the **Administration** subtab.

4. In the **Reading Access Area** drop-down list, click the arrow and select **Associate User Reading Access Areas**.
5. Using the arrows in the selection dialog box, move the user user reading access areas from available to selected, then click **Next**.



Select users reading access areas accessible by this object

Available reading access areas: Human Resources, project workflow, Project X

Selected reading access areas: Dvpt team

6. In the next dialog box, if an object reading access area corresponds to the user reading access areas, you are invited to validate this area, otherwise you must enter the name of a new user reading access area, which will be created corresponding to the previously selected user reading access areas.
7. Click **Finish**.

If you had to create a new object reading access area, this is automatically added in the reading access diagram and is connected to the corresponding user reading access area or areas, as well as to the ***Standard*** reading access area.

Propagating Reading Access Areas

A reading access area can be propagated to objects connected to a given object. You can propagate reading access areas from:

- **Hopex Administration**
- **Hopex**.

☛ *The propagation trace is kept when you propagate from **Hopex**.*

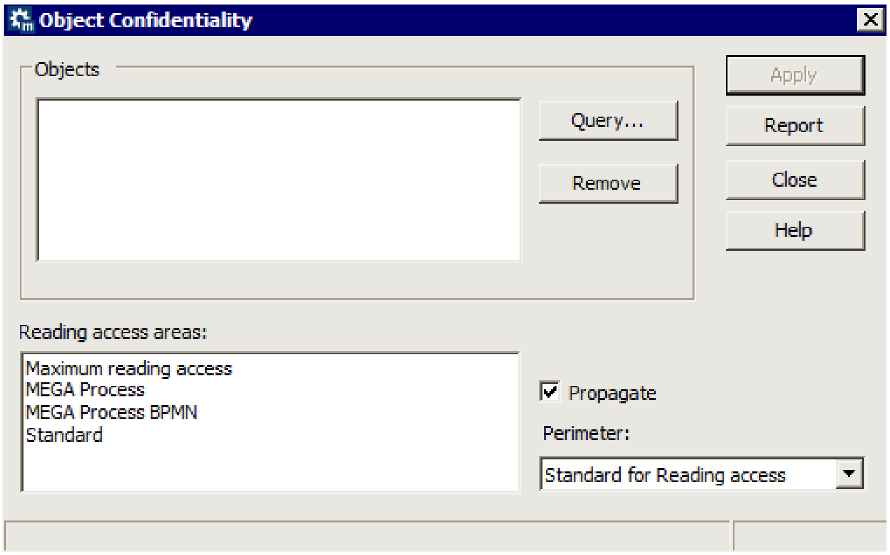
Propagating a reading access area from Hopex Administration

To propagate a reading access area:

1. Connect to **Hopex Administration** and select the repository concerned.

   ☛ *See Accessing Repositories.*

2. Right-click the repository concerned and select **Object Management > Object Confidentiality Setting**.



3. In the **Object Confidentiality** dialog box that opens, click **Query**.
4. Select the desired objects using the query tool, then click **OK**.
5. In the **Object Confidentiality** dialog box, in the **Reading Access Areas** frame , select the reading access area you want to apply to the selected objects.
6. (Optional) Select **Propagate** to propagate the reading access area to sub-objects.
7. Click **Apply**.

   ☛ *The operator used to propagate reading access areas is the "Standard for reading access" operator.*

Propagating a reading access area from Hopex

To propagate a reading access area:

1. Connect to **Hopex**.

   ☛ *See Connecting to Hopex Customizing Desktop.*

2. Open the **Administration** navigation window.
3. In the **Administration** navigation window, right-click the reading access area you wish to propagate and select **Propagation of reading access area to associated occurrences**.
   A dialog box may warn you of the presence of propagation conflicts.

   ☛ *Conflicts can arise if child objects are already connected to a different reading access area. You are informed that propagation stops* *at this child object (the physical link that connects the object to a reading access area is stronger than the reading access attribute value)*

☛ *The operator used at reading access area propagation is the "Standard for reading access" operator.*

☛ *Alternatively, you can use the confidentiality area propagation tool via the menu **Tools > Manage > Object Confidentiality Setting**.*

Managing MetaClass Sensitivity and Reading Access Areas

☛ *The attribute enabling configuration of MetaClass sensitivity is accessible only if you have the **Hopex Power Supervisor** technical module.*

In the reading access management frame:

- a user is connected to a reading access area that defines all the objects he/she can see.

  ☛ *See Connecting Users to Reading Access Areas.*

- an object type (MetaClass) is characterized by its sensitivity.

A MetaClass can be of sensitivity:

- **standard** (default value)
  Occurrences of the MetaClass created by a user belong to the user reading access at creation area or the Standard reading access area if the user does not have a reading access at creation area.
- **High**
  Occurrences of the MetaClass created by a user belong to the reading access area of the user that creates them.

  ☛ *To modify the sensitivity of a MetaClass, you must have rights to modify **Hopex** data. The option "Authorize Hopex Data Modification" must be activated at environment level, see Options.*

You can modify sensitivity value of the MetaClass.

☛ *See Modifying MetaClass sensitivity.*

☛ *By default, a MetaClass is **Standard** sensitivity.*

Opening the Hopex MetaClasses reading access configuration dialog box

To open the **Hopex** MetaClasses reading access configuration dialog box:

1. From **Hopex Administration**, connect to the environment.

   ☛ *See Connecting to an Environment.*

2. Expand the **User management** folder of the environment.
3. Right-click the **Data Reading Access** folder and select **Configure Hopex MetaClasses for reading access**.
   The **Hopex MetaClasses Reading Access Configuration** dialog box appears, listing the available **MetaClasses**.

   The icon alongside the name of each MetaClass indicates that default values:

   - have been modified
   - are retained.

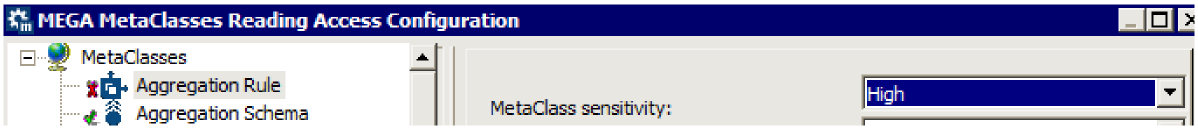
Modifying MetaClass sensitivity

To modify MetaClass sensitivity:

1. Open the **Hopex** MetaClasses reading access configuration dialog box.

   ☛ *See Opening the Hopex MetaClasses reading access configuration dialog box.*

2. In the list of **MetaClasses**, select the desired MetaClass.
3. In the right pane, select the MetaClass sensitivity value:



MEGA MetaClasses Reading Access Configuration

MetaClasses
Aggregation Rule
Aggregation Schema

MetaClass sensitivity: High

☛ *A red cross alongside the name of the MetaClass indicates that at least one attribute of the MetaClass has no longer its default value.*

4. Click **OK**.

   Modifications carried out may be canceled when your environment is updated. Remember to back up your extensions (metamodel and technical data).

Hiding confidential or sensitive objects in a diagram

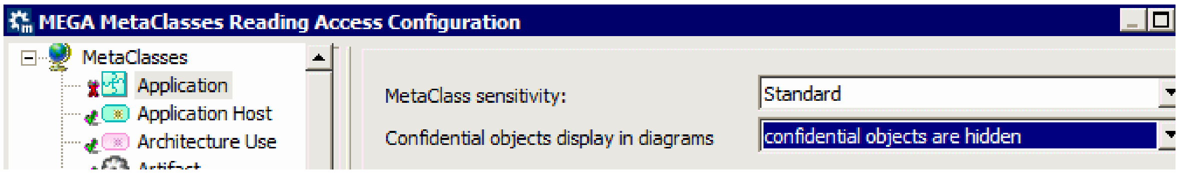
So as not to distort a diagram, confidential or sensitive objects are visible by default.

To hide confidential or sensitive objects in a diagram:

1. Open the **Hopex** MetaClasses reading access configuration dialog box.

   ☛ *See Opening the Hopex MetaClasses reading access configuration dialog box.*

2. In the list of **MetaClasses**, select the desired object.
3. In the right pane, in the **Confidential objects display in diagrams** box, select "Confidential objects are hidden".



MEGA MetaClasses Reading Access Configuration

MetaClasses
Application
Application Host
Architecture Use

MetaClass sensitivity: Standard

Confidential objects display in diagrams: confidential objects are hidden

☛ *A red cross alongside the name of the MetaClass indicates that the MetaClass default value has been modified.*

4. Click **OK**.

   ☛ *These modifications are not taken into account in the metamodel until the metamodel is compiled. Compile the metamodel before closing the **Hopex Administration** module, see Compiling an Environment.*

   Objects corresponding to this MetaClass will be hidden in the diagram.

   ☛ *When you select "Confidential objects are visible", objects corresponding to this MetaClass appear grayed in the diagram and you cannot access information relating to these objects.*

   Modifications carried out may be canceled when your environment is updated. Remember to back up your extensions (metamodel and technical data).

Confidential or Sensitive Object Behavior

A confidential object is inaccessible to a user that does not have access to the corresponding reading access area.

It is as if the object did not exist, the object:

- does not appear in lists.
- is ignored in query results.
- does not appear in reports, reports (MS Word), Report DataSets or Web sites.
- is not exported, duplicated, deleted or backed up, and its possible "children" (operations of a process, for example) are considered as orphans.
- only appears when another object is created with the same name or when a higher level object with a lower reading access level is deleted.
- cannot be modified.

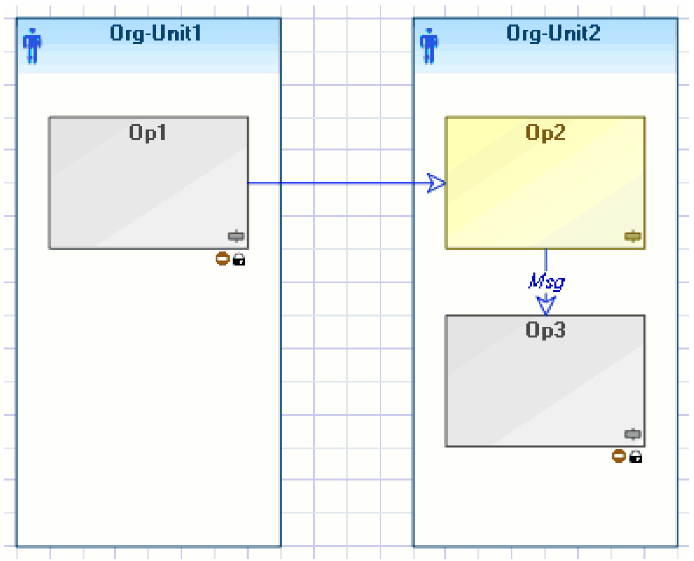
For more details on reading access areas, see Managing Reading Access in Hopex.

Displaying a confidential or sensitive object in a diagram

By default, confidential or sensitive objects are visible in diagrams.

☛ *To hide confidential or sensitive objects, see Hiding confidential or sensitive objects in a diagram.*

The name of a confidential or sensitive object is visible in the diagram, but its properties are not accessible. It appears grayed and an icon at bottom right indicates that the object is confidential or sensitive.



A confidential or sensitive object can only be resized and moved.

To hide MetaClass occurrences in diagrams:

1. Connect to **Hopex**.

   ☛ *See Connecting to Hopex Customizing Desktop.*

2. In the **Hopex** menu bar, select **View > Navigation Windows > MetaStudio**.
3. Expand the **MetaClass** folder and its sub-folders.
4. Right-click the desired MetaClass and select **Properties**.
5. In the **Characteristics** tab (**Standard** subtab) for the **Confidential objects display in diagrams** property, select **Confidential objects are hidden**.
6. Click **OK**.
   The occurrences of the corresponding MetaClass are now hidden in diagrams.

   ☛ *You can also configure display of objects in diagrams via the reading access diagram, see Hiding confidential or sensitive objects in a diagram.*

Export and Duplication

In the case of object export and duplication, if these operations have an impact on confidential objects, a warning message will ask you if you want to continue. If you do execute export or duplication, the confidential objects concerned will be neither copied nor exported.

Invisible objects (confidential) are not duplicated during object duplication. Only a message in the status bar indicates that the duplication result is incomplete.

Generation of reports and Report DataSets

When created, reports and Report DataSets are created with the reading access level of their creator (and not with the reading access level at creation).

Generation of reports (MS Word) and Web sites

When created, reports (MS Word) and Web sites are created with the reading access level of their creator (and not with the reading access level at creation).

They are then always generated at their reading access level.

A **Reading access area path** attribute exists on all reading access areas. If this attribute is specified, Web sites and reports (MS Word) are generated in this folder. Reports (MS Word) and Web sites are generated in this folder to facilitate reading access management of generated files by defining access rights to this generation folder. This task should be handled by the system administrator.

Confidential report (MS Word) and Web site generation paths can be defined in the properties dialog box of a reading access area.

To define this path:

1. In the reading access area properties dialog box, select the **Characteristics** tab.
2. In the **Reading access area path** field, specify the required path.
3. Click **OK**.

Macros

The principle of reading access management in macros is to carry out all calculations in **Hopex** and hide confidential or sensitive objects from users that do not have sufficient reading access area access rights to view them.

By default a macro is executed at user reading access level.

A macro can also be executed at its own reading access level.

If you execute a macro with a reading access level higher than the level of the current user, the methods.

- GetProp(xxx, "display") and GetFormatted return empty,
- GetProp("xxx") returns the value.

ExecuteGlobal and ***CreateObject*** ("Mega.Application") methods are prohibited in macros.

Other properties are accessible.

So that a macro can be executed with its reading access level:

1. In the macro properties dialog box, select the **Characteristics** tab.
2. In **_ExecutionOptions**, select the **Execution at Reading Access level** option.

Confidential or sensitive objects and namespaces

If an object with a namespace is not confidential or sensitive, but its parent is confidential or sensitive, the name of the latter will be masked in **Hopex**. It appears in **Hopex** as:

```
" ***::Operation 1 "
```

Modifying Reading Access Areas

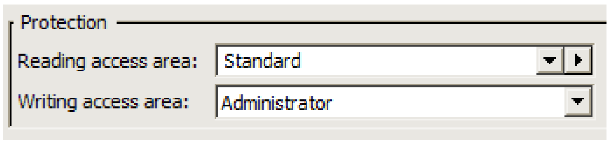
This section explains how to modify the reading access area area of an object or a user:

Modifying object reading access areas

From an object properties window, you can consult the reading access area to which it belongs.

To determine the reading access area of an object:

1. Right-click the object and select **Properties**.
2. In the object **Properties** dialog box, select the **General** tab, then the **Administration** subtab.
   In the **Protection** frame, you can consult and modify the **Reading access area**.


Protection
Reading access area: Standard
Writing access area: Administrator

Modifying user reading access areas

You can modify a user reading access area from:

- the user management window (**Reading access area** column of the person)

  ☛ *See Modifying the Properties of a User.*

  ☛ *The reading access diagram is compiled to take account of modifications.*

- the person properties window

  ☛ *See Defining a Person.*

  Warning: if you modify the reading access area in the user properties dialog box, you must recompile the reading access diagram so that the modification will be taken into account.

- the reading access diagram

  ☛ *See Connecting Users to Reading Access Areas or Reading access area of the user.*

You can modify a user reading access area at creation from:

- the person properties window

  ☛ *See Defining a Person.*

  Warning: if you modify the reading access area in the user properties dialog box, you must recompile the reading access diagram so that the modification will be taken into account.

- the reading access diagram

  ☛ *See Reading access area at creation.*

METACLASS CONFIDENTIALITY EXCEPTIONS

The following MetaClasses cannot be made confidential:

_Add-ins_data
_Add-ins_meta
_Brick
_Class
_ClassCommand
_Code Template
_Command
_Dispatch
_Executable
_MappingTypeItem
_MappingTypeItemProperty
_Object
_Operator
_Property
_Proposed_Table
_Resource
_StdFile
_Style
_TagAttributeDef
_TagAttributeDefValue
_TagDef
_Template
_Text
_Transaction
_TransactionData
_TransferredObject
_Type
_UML Reserved Word
Method author

MEGA Repository
ChangeItemData
ChangeItemDataTechnical
ChangeItemSystem
Generation kinematics
Component Template
DiagramTypeLink
DiagramTypeLinkStyle
DiagramTypeObject
DiagramTypeCollection
DiagramTypeField
DiagramTypeFormat
DiagramTypeParam
DiagramTypePath
DiagramTypePathPart
DiagramTypePopulating
DiagramTypeProperty
DiagramType
DiagramTypeView
Stem Codes Folder
Web Site Templates Folder
Analysis Templates Folder
Diagram Types Folder
HTML Formatter
Generality
Generator
Programming Language
Language
Animation Mask
Matrix Template

MetaAssociation	Associative Object
MetaAssociationEnd	Default Associative Object
MetaAssociationType	Generic Object
MetaAttribute	System Generic Object
MetaAttributeGroup	Analysis Parameter
MetaAttributeValue	Profile
Metaclass	Query Parameter
MetaClassDiagramType	Generation rule
MetaCommand	Modeling Rule
MetaField	Modeling Regulation
MetaList	Query
MetaListType	Web Site Template
MetaPattern	SQL Clause Type
MetaPicture	TaggedValue
MetaPropertyPage	Analysis Type
MetaTest	user
MetaTree	Descriptor Setting
MetaTreeBranch	DBMS Version
MetaTreeNode	Writing access area
Method	Reading access area

Command File Syntax

The following points are covered here:

- ✓ Command file extensions
- ✓ Object Naming Rules
- ✓ Commands
- ✓ Basic Syntax

COMMAND FILE EXTENSIONS

A command file is a file containing repository update commands. It can be generated by backup or object export (.MGR extension) or by logfile export (.MGL extension).

Command files can be obtained in two ways:

- By logical backup or by object export (.MGR): the *absolute identifiers* (IdAbs) of the imported objects are used and the authorization levels are kept.

  An absolute identifier is a string of characters associated with each object in the repository. This string is computed using the date and time the session was opened, the number of objects created since the session started, and the object creation date (in milliseconds). An absolute identifier provides a unique way to identify an object in the repository, so that even if the object is renamed, all the links to it are retained.

- By logfile export (.MGL): the commands contain, in addition to the *absolute identifiers* (IdAbs) of objects, that of the user executing each command to check at import that the user had the necessary rights to execute this update.

Result of the import of these files is therefore different:

- ".MGR" corresponds to an image, complete or partial, of the repository at a given moment. It is therefore recommended that it be imported into an empty repository to rebuild this image.
- ".MGL" corresponds to commands to be applied to the repository to pass from initial state to final state.

At import, checks are performed automatically as a function of the file extension:

- For command files with the MGR extension, the absolute identifiers of the imported objects are used and the writing access levels are kept.
- For command files with the MGL extension (exported logfile), the absolute identifiers of the imported objects are used. Writing access levels are checked. The authorization levels are kept if the updates are consistent with the writing access diagram for the environment, otherwise they are rejected.

OBJECT NAMING RULES

Object naming will depend on the uniqueness rule applied to its name. This rule is important, since the name appears in command files.

An object has a unique name

An object must have a unique name throughout the repository.

`For example, a report template (MS Word) has a name that appears in command files.`

A name is unique in a given context

Several objects can have the same name, but the name must be unique in a particular context: therefore it has a namespace.

`For example, an operation of an organizational process: its name must be unambiguous within the process, but several different process operations can carry the same name.`

For these objects, two names are presented to the user in the user interface:

Con-cept	Example	Comment
Name	Hire::Call candidate	Complete object name. Unique in the repository. Calculated from the local name and the name-space name (which can itself have a name-space). Here the operation "Call candidate" belongs to the "Hire" process.
Local Name	Call candidate	Name of the object in its namespace. Unique in the namespace.

Two names are used in **Hopex** command language:

Concept	MetaAttribute	Example of value
Internal name of the object. It contains HexaIdabs of the object.	Name	14B8162B3F3A0347
Local name of the object and Hex-aIdabs of its namespace.	Generic Local Name	Call candidate [85ED06B63EC95B6F]

These build rules ensure respect of naming rules imposed by the repository:

- The name must be unique: the IdAbs is built to be so.
- The local name must be unique in its context: specify a uniqueness constraint on the GenericLocalName.

☛ *If the object is detached from its namespace, in the local name the indicated HexaIdAbs is then a string of 16 "0".*

Objects without name constraint

There is no name uniqueness constraint on certain objects such as messages: the same operation can send or receive several messages with the same name.

In this case, the object constitutes its own namespace.

Two names are used in **Hopex** command language:

Concept	MetaAttribute	Example of value
Internal "Name" of the object. It contains HexaIdabs of the object.	Name	14B8162B3F3A0347
Local name of the object and HexaIdabs of its namespace (itself).	Message Local Name	Convocation [14B8162B3F3A0347]

COMMANDS

Commands on objects

- .Create (creation of an object)
- .Update (modification of an object MetaAttribute)
- .Delete (deletion of an object)

Commands on links

- .Connect (creation of a link between two objects)
- .Disconnect (deletion of a link between two objects)
- .Change (modification of a link MetaAttribute)

Other Commands

- .Validate (triggers intermediate save on import)
- .Description (produces display in import dialog box)

Rules to be respected

Command files must comply with the following rules:

- A command line cannot contain more than 5000 characters.
- Object names are limited to:
  - 63 characters for object types without namespace.
  - 255 characters for object types with namespace (name or local name).
- Commands begin with a verb infinitive prefixed by ".".
- The "." of the command must be in the first column.
- Use a hyphen (-) at the end of a line to indicate that it continues on the next line.
- Comment lines are indicated by a hyphen (-) at the beginning of the line.
- Use double-quotes (") around values that contain spaces or characters other than letters or digits.

Remarks

- Certain objects are functionally invalid if one of their MetaAttributes is not entered or a link is not defined. For example, a diagram type object must be connected to another object by a descriptor type link. We say that the diagram describes this object.
- To exchange data between two **Hopex** environments, they must have identical metamodels and coherent user diagrams.

Commands as function of file type

Each command must consist of:

- a verb indicating the action to be carried out
- a list of parameters required to carry out this action (object types and names)
- a keyword ".CHK" followed by a list of the IdAbs of objects impacted by this command.

☛ *The fact of repeating the object name and IdAbs in the command enables its correct execution, even if the object has been renamed.*

The difference between the command of an ".MGR" file and the same command of an ".MGL" file is in the ".CHK":

- they have the same verb
- they have the same list of parameters
- the ".CHK" of MGL contains in addition in last position, the IdAbs of the user that issued the order.

☛ *A third file format (obsolete in this version) is ".MGE". In these files commands do not have a .CHK. The IdAbs are assigned as required. It is therefore not possible to process "namespaced" objects for which the namespace IdAbs cannot be assigned, since it forms part of their name.*

References to the metamodel

Each metamodel instance (MetaClass, MetaAttribute, ...) can be prefixed by its IdAbs. This assures permanence of files despite renamings which may be carried out in the metamodel.

```
Example:

"~OsUiS9B5iiQ0[Operation]" is equivalent to "Operation".
```

BASIC SYNTAX

See:

Creating an Object

Syntax	***.Create ."Object type""Object name" - .CHK "..."***
Example 1	.Create ."~IdAe93gyh020[Report template (MS Word)]" "Application documentation" - .CHK "w0e4VVXC)440e0SDsNpple00"
Example 2	.Create ."~OsUiS9B5iiQ0[Operation]" "14B8162B3F3A0347" - .CHK "GZB5hOXE)Sq0C30000mCpCpC"

In this command, the ".CHK" is the concatenation of the following IdAbs:

- IdAbs of the object
- IdAbs of object writing access
- In the case of the MGL, the IdAbs of the user that made this command.

Certain MetaAttributes, such as "Creation date" or "Creator" can only be updated at object creation. They must therefore be incorporated in this command.

Example:

```
Create ."~OsUiS9B5iiQ0[Operation]" "14B8162B3F3A0347" -
    .CHK "GZB5hOXE)Sq0C30000mCpCpC" -
    ."~510000000L00[Creation Date]"           "2003/08/13 10:42:51" -
    ."~(10000000v30[Creator]"               "OmNRasMwq400" -
    ."~520000000L40[Create Version]"          "25088"
```

The "Creator" and "Modifier" MetaAttributes contain the IdAbs of users that have created and modified the object. If they are not specified in the command, they automatically take the IdAbs of the user importing the file.

Similarly, "Link creation date" and Link modification date" are specified from the import date if they are absent.

Deleting Objects

Syntax	***.Delete ."Object type" "Object name" -*** ***.CHK "..."***
Example 1	.Delete ."~IdAe93gyh020[Report template (MS Word)]" "Application documentation" - .CHK "w0e4VVXC)440"
Example 2	.Delete ."~OsUiS9B5iiQ0[Operation]" "14B8162B3F3A0347" - .CHK "GZB5hOXE)Sq0"

In this command, the ".CHK" is the concatenation of the following IdAbs:

- IdAbs of the object
- In the case of the MGL, the IdAbs of the user that made this command.

Deletion of an object systematically results in:

- Loss of its attribute and text values.
- Deletion of all of the links around the object.

Modifying an Object

Syntax	***.Modify ."Object type" "Object name" -*** ***.CHK "..." -*** ***."metaattribute1" "Value1" -*** ***."metaattribute2" "Value2"***
Example 1	.Update ."~MrUiM9B5iyM0[Application]" "874B9C483D7828C6" - .CHK "PjqX8n9UzOCA" - ."~610000000P00[Modification Date]" "2010/09/07 10:26:30" - ."~b10000000L20[Modifier]" "xDqT)UdFwC10" - ."~2yUL4SsRp4B0[Application Code]" "GESTCAT11" - ."~ByUL4SsRpeB0[Operating Application Date]" "1995/10/04 23:00:00"
Example 2	.Update ."~gsUiU9B5iiR0[Organizational Process]" "0A496AAE407D1621" - .CHK "Vba2kgMV05Y5" - ."~pjRX1OOKne20[Process Frequency]" "Q""

In this command, the ".CHK" is the concatenation of the following IdAbs:

- IdAbs of the object
- In the case of the MGL, the IdAbs of the user that made this command.

The "Modification date" and "Modifier" MetaAttributes can be modified like standard MetaAttributes. If they are not specified in the command, they automatically take the file import date and the IdAbs of the user importing the file.

In the case of "Example 2" with a tabulated attribute, the value to be entered is the internal value (for process frequency this is "D" and not "Daily").

Modifying Texts

Syntax	***.Modify ."Object type" "Object name" -*** ***.CHK "..." -*** ***."Text name" "Text format"*** ***Text value*** ***.***
Example 1	.Update ."~MrUiM9B5iyM0[Application]" "DFE4E02F4D4D2BB3" - .CHK "AH(tl0UJDDxA" - ."~f10000000b20[Comment]" "g3TCfAJnyq00" 00680SbnxCMPqRc5SN6bpSsvXS6DfCZ5dN38rPcLaN31cPcLaRc5iC35dUpOpRsPST7HkUsnY 00680C6PSRcPSN6nfQ6DcSt9XC7HbKqqWQ5CWR6nbR4GWVJjd2WrzQNPSQtTbP6vfTLmqN 35Z 00602Sc5mPbnaPbmmC39pRqCWPMrj87Hk86PlS71XOsbiQNHX86vlS5mn3N9X3NqA000A .

Text format	Value
ASCII text	0 000000000000
Binary text	1 000000000001
RTF text	"MRDYO5Oe(smC"
HTML text	"LQDYO58M6tmC"
ANSI text	"G3000000W10S"

In this command, the ".CHK" is the concatenation of the following IdAbs:

- IdAbs of the object
- In the case of the MGL, the IdAbs of the user that made this command.

In a command file, each line of text is limited to 74 characters.

End of the text marked by a line containing only one point (".") in the first column.

A semi-colon in the first column inserts a blank line (the rest of the line must be left blank).

☛ *When text is extracted, lines are divided after character 73 and a semicolon is inserted in position 74, indicating that the next line is to be concatenated with the previous one.*

Complementary indicators:

- Each text modification applies to its totality. It is not possible to add just one complement.
- The semicolon character (;) in first position inserts an blank line. To reinitialize a text, it is sufficient that the text value contains just a semicolon.
- The characters period and semicolon (.;) in first and second position create a line containing a period only.
- Two semicolon characters (;;) in first and second position create a line containing a semicolon only.
- Apostrophe (’) and quotation marks (") are authorized as text values.
- The semicolon character (;) enables cutting of text lines exceeding 74 characters. The semicolon is therefore the last significant character in the line.

To reinitialize a text, it is sufficient that the text value contains just a semicolon.

```
Example:
Update ."~MrUiM9B5iyM0[Application]" "DFE4E02F4D4D2BB3" -
        .CHK "AH(tl0UJDDxA" -
        ."~f10000000b20[Comment]" "000000000000"
;
.
```

To delete a text, the text value should be left empty.

```
Example
Update ."~MrUiM9B5iyM0[Application]" "DFE4E02F4D4D2BB3" -
        .CHK "AH(tl0UJDDxA" -
        ."~f10000000b20[Comment]" "000000000000"
.
```

A reinitialized text contains nothing but it exists, while a deleted text no longer exists. For example the query *"Select Application where Comment null"* returns applications that have no comment, but not those that have a reinitialized comment.

Modifying a Name

Syntax	***.Modify ."Object type" "Object name" -*** ***.CHK "..." -*** ***."Name or Local name" "Value "***
Example 1	.Update ."~ldAe93gyh020[Report template (MS Word)]" "Report template (MS Word)-1" - .CHK "RJ(tBUUJD5(AV(WEIeZIDT4B" - ."~210000000900[Name]" "Report template (MS Word)-New"
Example 2	.Update ."~MrUiM9B5iyM0[Application]" "DFE4E02F4D4D2BB3" - .CHK "AH(tl0UJDDxA" - ."~g20000000f60[Generic Local name]" "Application- 1[0000000000000000]"

Creating and Modifying an Object with a Single Command

At object creation, creation of a "modification" command by MetaAttributes to be specified is of no interest: MetaAttributes (non-textual) can be directly assigned by the create command.

Syntax	***.Create ."Object type" "Object name" -*** ***.CHK "..." -*** ***."metaattribute1" "Value1" -*** ***."metaattribute2" "Value2"***
Example 1	.Create ."~MrUiM9B5iyM0[Application]" "DFE4E02F4D4D2BB3" - .CHK "AH(tl0UJDDxAC30000mCpCpC" - ."~510000000L00[Creation Date]" "2011/02/05 11:41:35 PM" - ."~610000000P00[Modification Date]" "2011/02/06 12:31:38 AM" - ."~(10000000v30[Creator]" "V(WEIeZIDT4B" - ."~b10000000L20[Modifier]" "V(WEIeZIDT4B" - ."~520000000L40[Create Version]" "29248" - ."~620000000P40[Update Version]" "29248" - ."~)20000000z70[Confidentiality area identifier]" "sTlVwxdH3100" - ."~2yUL4SsRp4B0[Application Code]" "AA" - ."~KyUL4SsRpCC0[Version Number]" "12" - ."~ByUL4SsRpeB0[Operating Application Date]" "2011/02/11 11:00:00" - ."~)Sgoy)ygu020[Application Type]" "M" - ."~nOU8g8IMCb30[Converted Name Version]" "0" - ."~PYq45X2wBP92[Date of C&A Completion]" "2011/02/06 11:00:00" - ."~Ol8pdE(l8r(0[Deployment Date]" "2011/02/27 11:00:00" - ."~g20000000f60[Generic Local name]" "Application- 1[0000000000000000]" - ."~PZq41c2wBXP2[Security Planning]" "Operational" - ."~a20000000H60[LanguageUpdateDate]" "2011/02/05 23:41:57"

Creating a Link Between Two Objects

Syntax	***.Connect ."Object type" "Object 1 name" ."MetaAssociationEnd" "Object 2 name" - .CHK "..."***
Example 1	.Connect ."~MrUiM9B5iyM0[Application]" "DFE4E02F4D4D2BB3" ."~mi54NLn-HzCj0[Application within Internal Architecture]" "DFE4F2274D4D2C43" - .CHK "AH(tl0UJDDxAbJ(td8VJDD4B" - ."~710000000T00[Link creation date]" "2011/02/06 00:58:15" - ."~810000000X00[Link modification date]" "2011/02/06 00:58:15" - ."~720000000T40[Link Creator]" "V(WEIeZIDT4B" - ."~920000000b40[Link Modifier]" "V(WEIeZIDT4B" - ."~410000000H00[Order]" "9999"

In this command, the ".CHK" is the concatenation of the following IdAbs:

- IdAbs of object 1
- IdAbs of object 2
- In the case of the MGL, the IdAbs of the user that made this command.

The "Order" MetaAttribute is optional. If present, the order is numeric on maximum four positions, otherwise the order is set by default to 9999.

The "Link creator" and "Link modifier" MetaAttributes contain the IdAbs of users that have created and modified the link. If they are not specified in the command, they automatically take the IdAbs of the user importing the file.

Similarly, "Link creation date" and Link modification date" are specified from the import date if they are absent.

☛ *If this link is used to build a namespace, it must be completed by modification of the local name of the namespaced object to maintain repository consistency.*

Similar to object creation, it is possible to specify link MetaAttributes (except texts) directly in this command, without passing via a modification command.

Modifying a Link

With the exception of its header, this command has the same syntax as the object modification command.

☛ *See Modifying an Object.*

Syntax	***.Change ."Object type" "Object 1 name" ."MetaAssociationEnd" "Object 2 name" -*** ***.CHK "..." -*** ***."metaattribute 1" "Value 1" -*** ***."metaattribute 2" "Value 2" -*** ***."Text name" "Text format" -*** ***Text value*** ***.***
Example	.Change ."~MrUiM9B5iyM0[Application]" "DFE4E02F4D4D2BB3" ."~mi54NLnHzCj0[Appli-cation within Internal Architecture]" "DFE4F2274D4D2C43" - .CHK "AH(tl0UJDDxAbJ(td8VJDD4B" - ."~810000000X00[Link modification date]" "2011/02/06 1:22:26 AM" - ."~920000000b40[Link Modifier]" "V(WEIeZIDT4B" - ."~b20000000L60[LinkLanguageUpdateDate]" "2011/02/06 01:22:26" - ."~C3cm9FyluS20[Link Comment]" "g3TCfAJnyq00" 00680SbnxCMPqRc5SN6bpSsvXS6DfCZ5dN38rPcLaN31cPcLaRc5iC35dUpOpRsPST7HkUs nY 00680C6PSRcPSN6nfQ6DcSt9XC7HbKqqWQ5CWR6nbR4GWVJjd2WrzQNPSQtTbP6vfTLmq N35Z 00362Sc5mPbnaPbmmC39pR68WR69XS5nX3N9X3NqA000A .

In this command, the ".CHK" is the concatenation of the following IdAbs:

- IdAbs of object 1
- IdAbs of object 2
- In the case of the MGL, the IdAbs of the user that made this command.

The MetaAttributes "Link modification date", and "Link modifier" can be modified just like standard MetaAttributes. If they are not specified in the command, they automatically take the file import date and the IdAbs of the user importing the file.

Deleting a Link

Syntax	***.Disconnect ."Object type" "Object name 1" ."MetaAssociationEnd 2" "Object name 2" -*** ***.CHK "..."***
Example	.Disconnect ."~MrUiM9B5iyM0[Application]" "DFE4E02F4D4D2BB3" ."~mi54NLn-HzCj0[Application within Internal Architecture]" "DFE4F2274D4D2C43" - .CHK "AH(tl0UJDDxAbJ(td8VJDD4B"

In this command, the ".CHK" is the concatenation of the following IdAbs:

- IdAbs of object 1
- IdAbs of object 2
- In the case of the MGL, the IdAbs of the user that made this command.

Deletion of a link results in loss of link MetaAttributes values.

☛ *If this link is used to build a namespace, it must be completed by modification of the local name of the "namespaced" object to maintain repository consistency. Its namespace has become "[0000000000000000]".*

Managing Translations

For each language supported by **Hopex**, two MetaAttributes indicate the last modification date of translations in a language:

- "[LanguageUpdateDate (Language)]" for an object
- "[LinkLanguageUpdateDate (Language)]" for a link

These MetaAttributes are managed following the same rules as the"Modification date" and "Link modification date" MetaAttributes:

- They can be modified in the same way as standard MetaAttributes.
- If they are not specified in the command modifying a translation, they automatically take the file import date.

Validating Import

Syntax	***.Validate***

This command does not contain a .CHK and produces an intermediate save operation at import.

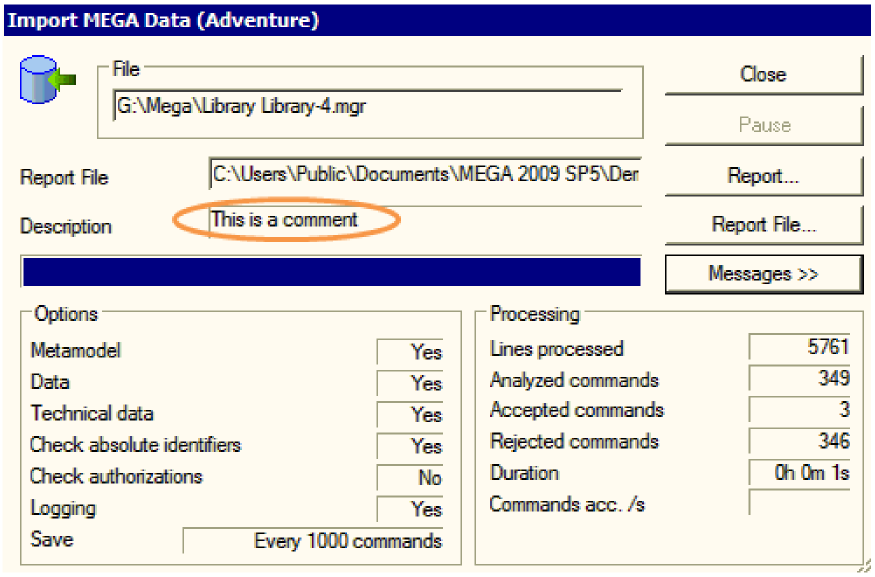
☛ *This command invalidates save operation selection made by the user interface. For example, to validate consistency of a command file, the user generally imports with "Save Never". Commands are then saved until the last ".Validate" of the file.*

Displaying a Comment in the Import Dialog Box

Syntax	***.Description 'Text'***
Example	.Description 'MetaClass: Acceptance Criteria'

This command does not contain a .CHK.

The text appears on the user interface at import.



File: G:\Mega\Library Library-4.mgr

Report File: C:\Users\Public\Documents\MEGA 2009 SP5\Der

Description: This is a comment

Options

Metamodel	Yes
Data	Yes
Technical data	Yes
Check absolute identifiers	Yes
Check authorizations	No
Logging	Yes
Save	Every 1000 commands

Processing

Lines processed	5761
Analyzed commands	349
Accepted commands	3
Rejected commands	346
Duration	0h 0m 1s
Commands acc. /s	

Transforming an MGL File to MGR

☛ *See Command file extensions.*

You do not need to transform an .mgl file to .mgr.

To obtain the same result, when importing an .mgl file:

- in the data import dialog box, in the **Checks** frame, clear the **Check Writing Accesses** check box.

Transforming an MGR File to MGL

☛ *See Command file extensions.*

You do not need to transform an .mgr file to .mgl.

To obtain the same result, when importing an .mgr file:

- in the data import dialog box, in the **Filter** frame, select the **Reassign User** check box.
  Each command is then processed as if its CHK contained the IdAbs of the importing user. Writing access checks are carried out related to its rights.

  ☛ *At import in the CHK of an MGL command, the "Reassign User" check box also allows substitution of the IdAbs of the user by that of the person importing.*

OPTIONS

This chapter presents the various tools and options used to configure and customize **Hopex**.

The following points are covered here:

INTRODUCTION TO OPTIONS

Option Overview

Hopex options concern:

- site technical configuration.
- values proposed by default for each function of **Hopex**. These values can be modified by users on each workstation.
  This configuration is described in the guides covering each function.

Hopex options are accessible at several levels. **Hopex** functions can be configured at the following levels:

- site

  The site is the location where ***Hopex*** *is installed; it is the root of the application.*

- environment
- profile (which groups a configuration common to several users)
- user

By default option levels follow an inheritance mechanism:

- the environment inherits options define at site level.
- the profile inherits options defined at environment level.
- the user inherits options defined at connection profile level.

Customizations made at user level are of highest priority, followed in order of priority by those made at profile, environment and site levels.

Having modified option values, it is recommended that you dispatch or save your work, close Hopex and then reopen it. Refresh issues can occur if these precautions are not taken.

For detailed information on these options, see the context-sensitive help in the lower part of the window.

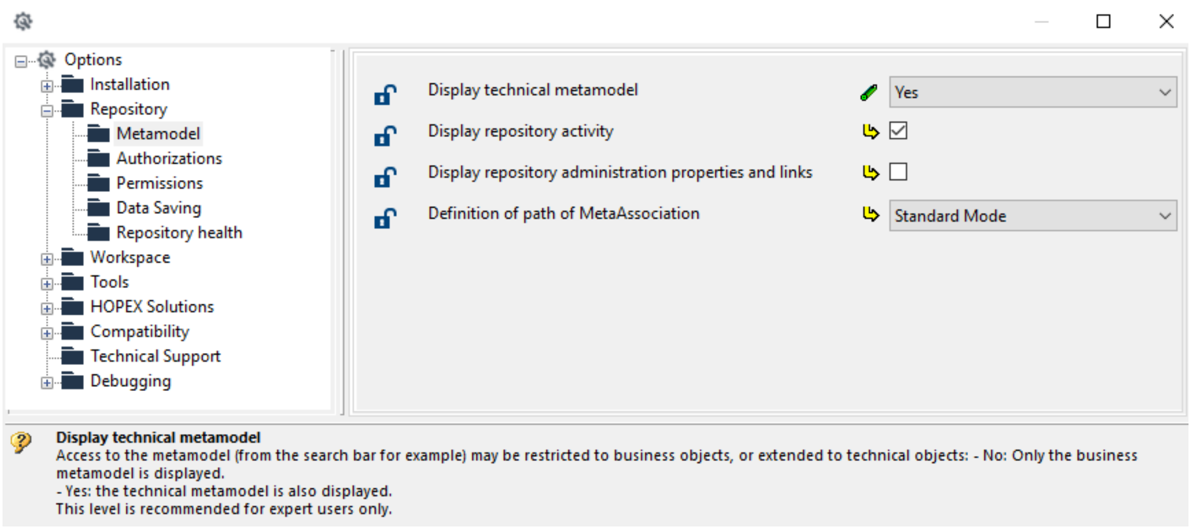
Option Window Presentation

The left pane of the window contains the various option groups available for the site, environment, profile, and user.

☛ *See Managing User-Related Options.*

The right pane enables configuration of the various options corresponding to the group selected in the left pane.

☛ *Options vary depending on your available products.*



Display technical metamodel: Yes
Display repository activity: ☑
Display repository administration properties and links: ☐
Definition of path of MetaAssociation: Standard Mode

Display technical metamodel
Access to the metamodel (from the search bar for example) may be restricted to business objects, or extended to technical objects: - No: Only the business metamodel is displayed.
- Yes: the technical metamodel is also displayed.
This level is recommended for expert users only.

For more details on an option:

- Click the option name to display the context-sensitive help in the lower part of the window.

☛ *When the user has a private workspace in progress, you cannot modify its options from **Hopex Administration**.*

Accessing Options

Options Level

In **Hopex Administration** you can modify options at the following levels:

- site
- environment

☛ *To modify options at profile or user level, see Modifying options for a profile or Modifying options at user level.*

Modifying options at site level

Storage: option values at site level are stored in the MegaSite.ini file. This file is accessible in the HAS console: **Modules > Module Settings > MegasiteSetings**.

To modify options at site level:

1. Start **Hopex Administration**.

   ☛ *See Accessing Hopex Administration.*

2. In the navigation tree, right-click the site name and select **Options > Modify**.
   The site options window opens.

Modifying options at environment level

Storage: option values at environment level are stored in the MegaEnv.ini file. This file is accessible in the environment folder: **<HAS instance repository>> Repos > <environment name>**.

To modify options at the environment level from **Hopex Administration** :

1. From **Hopex Administration**, connect to the environment.

   ☛ *See Connecting to an Environment.*

2. Right-click the environment name and select **Options > Modify**.
   The environment options window opens.

Option Inheritance

An option inherits a value defined at a higher level:

- A user inherits options defined at the connection profile level.
- A profile inherits options defined at the environment level.
- An environment inherits options defined at the site level.

The icon located opposite the option indicates the inheritance, or not, from the higher level:

- **Default value** indicates the inheritance from the higher level.
- **Modified value** indicates that the inherited option value has been modified. The value is no longer inherited from the higher level.

To specify that an option does not inherit the value defined at higher level:

1. Open the options page.

   ☛ *See Options Level.*

2. Click **Default value** .

   The icon changes in **Modified value** .

Controlling Modification of Options

You can prohibit modification of any option at a level lower than your current level.

```
Example: if you open options of the environment, you can
prohibit modification of all options at user level.
```

Prohibiting modification of a lower level option

To prohibit modification of a lower level option:

1. Access the options.

   ☛ *See Options Level.*

2. Click icon located opposite the option concerned.

   The padlock closes : option modification by a user is now prohibited from **Hopex**.

Unlocking the modification of a lower level option

To unlock modification of a lower level option:

1. Access the options.

   ☛ *See Options Level.*

2. Click the closed padlock icon .

   The padlock opens : modification of the option is again possible.

Reinitializing Option Values

You can reinitialize the values for:

- an option
- an option group

Reinitializing the values of an option

To reinitialize the value of an option:

1. Access the options.

   ☛ *See Options Level.*

2. Click **Modified Value**, the icon changes to **Default Value**.
   The value of the option is reinitialized.

Reinitializing the values of an option group (Windows Front-End)

To reinitialize values of an option group from **Hopex Administration**:

1. Access the options.

   ☛ *See Options Level.*

2. In the options tree, right-click the option group and select **Reinitialization**.
   All the options in the group selected are reset to their default values.

GENERATING THE LIST OF OPTIONS

You can generate a report that lists all options classified by group, with their comments.

To generate the list of options:

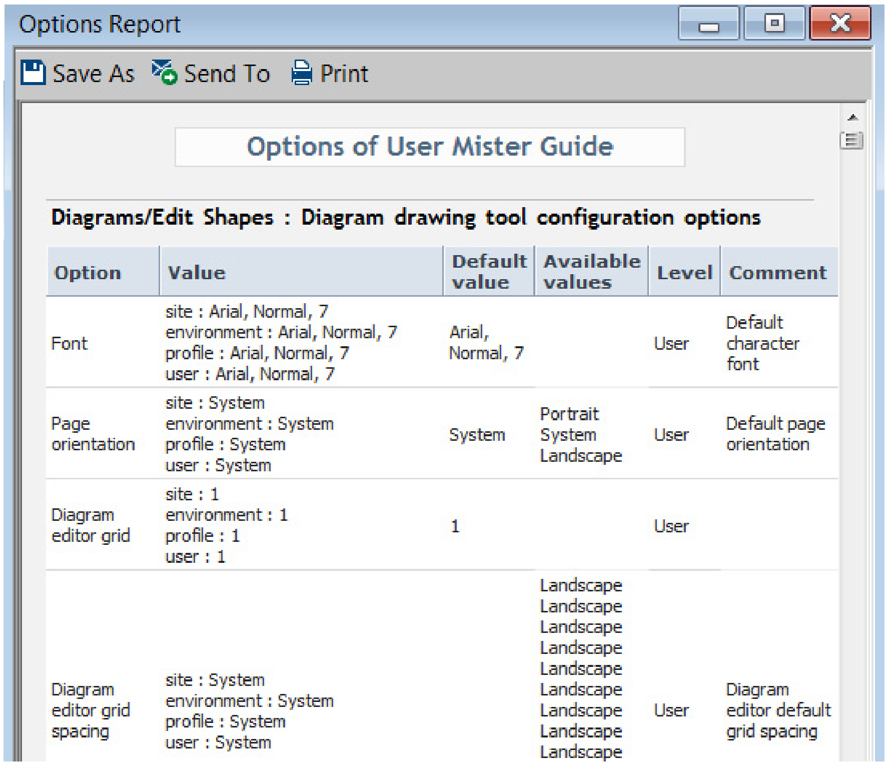
1. Access the Options at the required level (site, environment, or user).

   ☛ *See Accessing Options.*

2. Right-click **Options** and select **Report**.
   Report generation may take some time.

The Options Report includes:

- the names of available options
- the values available for each option
- the default value
- a comment explaining the option use context
- the option level:
  - user
  - environment
  - site



Options of User Mister Guide

Diagrams/Edit Shapes : Diagram drawing tool configuration options

Option	Value	Default value	Available values	Level	Comment
Font	site : Arial, Normal, 7 environment : Arial, Normal, 7 profile : Arial, Normal, 7 user : Arial, Normal, 7	Arial, Normal, 7		User	Default character font
Page orientation	site : System environment : System profile : System user : System	System	Portrait System Landscape	User	Default page orientation
Diagram editor grid	site : 1 environment : 1 profile : 1 user : 1	1		User	
Diagram editor grid spacing	site : System environment : System profile : System user : System		Landscape Landscape Landscape Landscape Landscape Landscape Landscape Landscape Landscape	User	Diagram editor default grid spacing

To save this report in .html format:

❱ Click **Save As** and select *.htm format.

OPTION GROUPS

☛ *At user configuration level, certain options are grayed. They can be defined only for an environment or site and not for a user.*

Note that repository and modeling options contain important information for the functional administrator.

Installation

Options linked to installation:

- company information
- activated data languages
- user management
- Web user desktop (Web application)

Options available at environment level only:

- licenses
- documentation (URL)
- customization
- machine Translation
- cache management (advanced)
- currency
- electronic mail
- security

Repository

Options linked to the repository:

- display of some advanced metamodel part
- authorizations
- data saving (dispatch)

Options available at environment level only:

- permissions
- logging

Workspace

Options linked to the user workspace:

- desktop
- properties
- trees
- modeling and method regulations
- dashboards

These options enable to display certain functionalities or not.

Tools

Options linked to **Data Exchange**:

- import
- export
- exchanges with third party tools

Options linked to the **Documentation** generated by Hopex:

- reports
- Web sites

Options linked to the **Diagrams**:

- display
- intellibar
- status indicators

Options linked to **Assessments**

Options linked to **Collaboration**:

- history management
- review note management
- notification and object follow-up management
- Social
- Workflows
- Environment level only:
  - change management
  - workspace management

Options linked to the **Mapping Editor**

Options linked to **Explorer**

Options linked to **Query**

Options linked to **Simulation** (Environment level only)

Hopex Solutions

Options linked to Solutions:

- Common Features
- IT Architecture
- IT Portfolio Management
- Privacy Management
- Business Process Analysis
- Data Management
- Loss Data Collection (Environment level only)

Compatibility

Compatibility options with deprecated or Windows Front-End specific features.

Technical Support

Options regarding Technical Support access.

Debugging

Options regarding debugging.

Available for Hopex administrator and functional administrator profiles.

Managing Hopex Data Customization

To ensure a correct use of **Hopex**, by default it is forbidden to modify **Hopex** data. Modifying a **Hopex** object can generate errors at **Hopex** upgrades, import of correctives, etc.

The **Authorizing Hopex Data Modification** option allows modifying the **Hopex** metamodel or any other **Hopex** technical object.

This option should only be selected in certain highly specific cases, at debugging operations or at Bizzdesign request, and for a temporary period.

This option is:

- locked by default at environment level, with "Prohibit" value
  Only **Hopex Administrator** profile is allowed to modify this option.

  ☛ *See Controlling Modification of Options.*

- accessible in the **Options > Installation > Customization** folder.

  Specify this access level only for a highly advanced profile.

Languages and Dates

This chapter presents how to manage data languages and dates in **Hopex**.

The following points are covered here:

Managing Languages

See:

Supported Data Languages

Hopex supports more than 30 data languages: Arabic, Bosnian, Bulgarian, Chinese (Simplified), Chinese (Traditional), Croatian, Czech, Danish, Dutch, English, Spanish, Finnish, French, Greek, Hebrew, Hungarian, Icelandic, Indonesian, Italian, Japanese, Korean, Malay, Norwegian, Polish, Portuguese, Romanian, Russian, Serbian, Slovak, Slovenian, Swedish, Thai, Turkish, Vietnamese.

At installation, **Hopex** includes the following data languages: English, French, German, Italian, Spanish, and Portuguese. If needed, you can install additional languages (from the supported language list).

Installing Additional Languages

At installation, **Hopex** includes the following data languages: English, French, German, Italian, Spanish, and Portuguese.

If needed, you can install additional languages (from the supported language list).

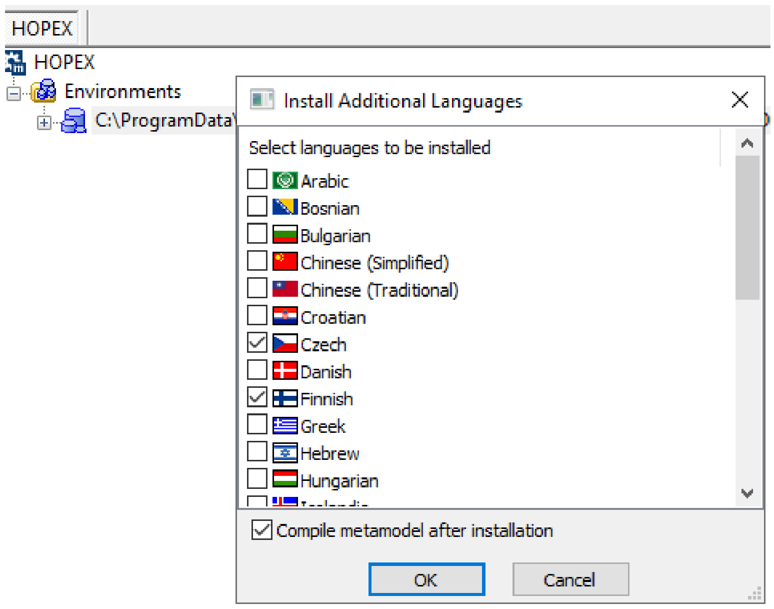
To install additional data languages in **Hopex**:

1. Connect to **Hopex Administration** and select the environment concerned.

   ☛ *See Connecting to an Environment.*

2. Right-click the environment and select **Metamodel > Install Additional Languages**.
   The dialog box **Install Additional Languages** opens.

3. Select the languages you want to be available in **Hopex**.



Selected: ✓ Czech, ✓ Finnish, ✓ Compile metamodel after installation

4. Click **OK**.
   A window indicates progress of import of the corresponding libraries.
   The selected languages are added to the list of available languages (in the options: **Installation > Languages**).
   For these languages to be available in **Hopex**, you must select them in the options, see Defining Available Data Languages.

Defining Available Data Languages

Among the data languages available in **Hopex**, you must select those that can be used to translate or show data:

- at environnement level, for all of the users
- at user level

☛ *For more information on the use of languages, see the* ***Hopex Common Features*** *guide, "****Hopex*** *in a Multilingual Context" section.*

To define the data languages available for all users:

1. Connect to **Hopex Administration**.
   ☛ *See Accessing Hopex Administration.*
2. Access the environment option management window.
   ☛ *See Modifying options at environment level.*
3. In the Options tree, select **Installation > Languages**.
4. In the right pane, select the languages in which you want the users to be able to work the data.
5. Click **OK**.

Defining the Language of e-mails in Workflows

To define the language of e-mails in workflows:

1. Access the options management window.

   ☛ *See Modifying options at environment level.*

2. In the options tree, select **Tools > Collaboration > Workflows**.
3. In the right pane, for the **Language for sending mail** option, select the language to be defined in e-mails.
4. Click **OK**.

Managing Languages in Web Applications

You can modify:

- the interface language in Web applications
- the data language in Web applications.

☛ *To manage languages in Web applications, see Hopex Administration Web guide.*

MANAGING DATE AND TIME FORMATS

In Hopex, the date and time formats depend on the data language format.

These formats are defined for each language in the Windows parameters of Hopex installation server.

If needed, you can change these formats in Hopex.

This customization is lost at Hopex upgrade.

This modification uncompile technical data.

To change the date/time format for a language:

1. Connect to **Hopex**

   *Check that you are allowed to modify Hopex data (**Options > Installation > Customization**), see Managing Hopex Data Customization.*
2. In the **Hopex** search tool, select **Languages**.
3. Click **Find** .
4. In the result list, right-click the language concerned and click **Properties**.
5. In the **Characteristics** tab, select the first **Characteristics** subtab.
6. In the **_LanguageCharacteristics** pane, add the date/time format you want to be customized:

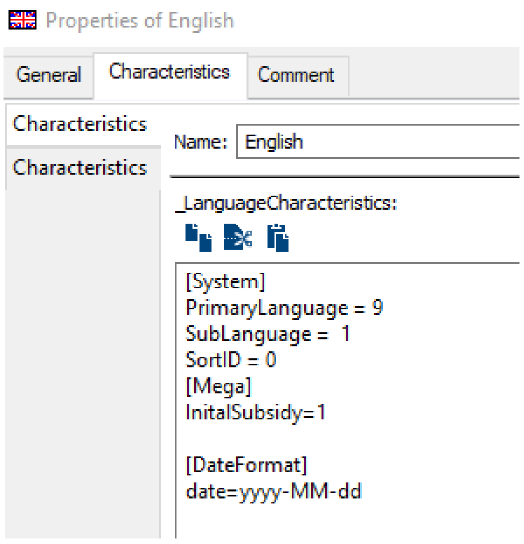
```
[DateFormat]
Date=<date format>
time=<time format>
```

   For dates, you can use separating characters like for example: "/", ",", "-", or " ".

```
Examples:
date=yyyy/MM/dd displays 2018/04/24
date=d-MM-yy displays 4-03-18
date=dd MMM yy displays 04 jul 18
```

For times, you can use separating characters like for example: "/", ".", or ":".

```
Examples:
time=HH:mm:ss displays 04:30:20
time=H:m displays 4:30
```



Name: English

_LanguageCharacteristics:

```
[System]
PrimaryLanguage = 9
SubLanguage =  1
SortID = 0
[Mega]
InitalSubsidy=1

[DateFormat]
date=yyyy-MM-dd
```

7. Click **OK**.
   The modified formats (date and/or time) are automatically taken into account.

   This modification uncompile technical data.

8. Compile technical data.

☛ *See Compiling an Environment.*

Date Format:	**Description**
d	The day of the month with one or two digits 1…9, 10, 11,..31
dd	The day of the month with two digits 01…09, 10, 11,..31.
M	The numeric format month with one or two digits 1…9, 10, 11, 12
MM	The numeric format month with two digits 01…09, 10, 11, 12
MMM	The abbreviated name of the month
Y	The year with one or two digits 9, 22
yy	The year with two digits 09, 22
yyyy	The year with four digits 2022

Time Format	**Description**
HH	Time on two digits 00…23
H	Time on one or two digits 0…23
mm	Minutes on two digits 00…59
m	Minutes on one or two digits 0…59
ss	Seconds on two digits 00…59
s	Seconds on one or two digits 0…59

Frequently Asked Questions

The following points are covered here:

Common Operations

How do I copy a repository from one environment to another?

Standard procedure: make a logical backup of the repository, then carry out a logical restore of the backup in an empty repository of the target environment.

For GBMS environments you can copy repository files (EMA, EMB, EMS, EMV) in a folder carrying the repository name, then create a reference for the repository in the second environment, but only if the metamodel is exactly the same in both environments.

Can I create a reference for an environment in another site?

No, the functional rule is that a reference for a Hopex environment should only be referenced in an installation (site).

Can I delete a user?

Yes, you can delete a user: see Deleting a User.

When you delete a user from the repository, all actions linked with this user are lost.

To delete a user but retain its actions, modify user repository access mode to **Inactive user** (see Preventing a User Connection). The user no longer appears in the connection dialog box, but its actions are kept.

☛ *You cannot delete the "Administrator" user.*

Can I delete a writing access area?

Yes, you can delete a writing access area.

☛ *You cannot delete the "Administrator" writing access area.*

When you delete a writing access area, the objects that were attached to it pass implicitly to "Administrator" writing access level.

Bizzdesign recommends that before deletion, you modify the writing access of objects attached to the writing access area.

Recurrent Messages

Abnormal operation when refreshing a private workspace

Symptom: Message stating "Could not refresh your private workspace".

Reason: Rejects occurred when importing private workspace updates into the reference repository.

Solution:

1. Examine the reject file Rmmjj.MGL (eg: MGLR07150000.MGL) in the user work repository (<repository>\USER\<user code>).
2. Identify and process causes of rejects (see Rejects When Dispatching).
3. Delete reject files that are no longer needed.

☛ *For as long as reject files are not deleted, a warning persists when connecting to Hopex.*

Environment version

Symptom: Message "Your environment and site are not of the same version" when opening an environment from the Hopex administration console" (or "Your environment and site are not of the same version. Your environment requires updating. Refer to documentation for how to carry out this action").

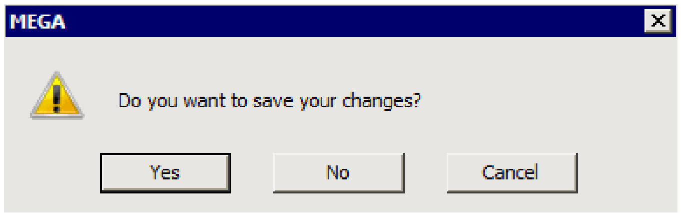
- Click **OK**.
  Another window appears displaying a second message: "Your environment requires an update for compatibility with your version of Hopex". Do you wish to run this procedure now? ".

Reason: It is possible that the environment has not been created, or is not at the same version level as the site referencing it: an update is therefore proposed:

- if you have a physical backup of environment data, accept modification by clicking **Yes**.
- If this is not the case, refuse the modification by clicking **No** to exit the Hopex data administration console. Then execute physical backup of data.

"Later" option not proposed at disconnect

Symptom: When you exit **Hopex**, the dialog box that appears does not propose the "Later" option to save your modifications.


MEGA

Do you want to save your changes?

Reason:

- You are not connected to **Hopex (Windows Front-End)** and the license used does not have technical module **MEGA Lan**.

  You prevent other users from dispatching their work performed in their private workspace.

- You are connected to **Hopex (Web Front-End)**, in a public workspace, your modifications are automatically saved.

PRODUCT CODES

Access your list of available products

To view the products to which you have access:

1. Start **Hopex Administration**.

   ☛ *See Accessing Repositories.*

2. In the menu bar, select **Help > About Hopex**.
3. Click **System Information**.
4. With the drop-down menu, select **Available components**.
   All the products for which you have a license are listed as well as their associated code.

   ☛ *For information on product availability (Windows Front-End, Web Front-End) and storage, see online documentation (**Concepts > Products**).*

GLOSSARY

absolute identifier	An absolute identifier is a string of characters associated with each object in the repository. This string is computed using the date and time the session was opened, the number of objects created since the session started, and the object creation date (in milliseconds). An absolute identifier provides a unique way to identify an object in the repository, so that even if the object is renamed, all the links to it are retained.
access area member	Access area member groups all persons and person groups belonging to an access area. This area defines objects that can be accessed by the person or person group.
access path	An access path indicates which folder you can use when creating a reference for an environment database or a site environment. When all repositories are created in the same location as the environment, the path created at installation (the DB folder under the environment root) is sufficient and is given as the default. When you want to save a repository in a folder other than that of the environment, you must declare a new access path.
access rights	User access rights are the rules that manage access to software functions and databases. You can restrict the access rights of a user to the different repositories defined in his/her work environment. You can also restrict what software features a given user can run, such as the descriptor editor, modifying queries, designing report templates (MS Word), deleting objects, dispatching private workspaces, importing command files, managing environments, repositories, users, writing access, etc.

administration	Administration consists of managing the work environment of repository users. This function is usually the responsibility of the administrator. Administration tasks include making backups of repositories, managing conflicts in data shared by several users and providing users with queries, descriptors, report templates (MS Word), etc. common to several projects.
Administration desktop	The **Hopex Administration** desktop (Web Front-End) is the Web version of the **Administration** (Windows Front-End) application accessible via an internet browser. It enables to manage Hopex users and permissions.
administrator	The administrator is a person who has administration rights to manage sites, environments, repositories and users. In addition to Administrator (who cannot be deleted) and MEGA users, created at installation, you can grant administration rights to other users.
attribute	See *Characteristic*.
business role	A business role defines a function of a person in a business sense. A person can have several business roles. A business role is specific to a repository.
characteristic	A characteristic is an attribute that describes an object or a link. Example: the Flow-Type characteristic of a message allows you to specify if this message is information, or a material or financial flow. A characteristic can also be called an Attribute.
command file	A command file is a file containing repository update commands. It can be generated by backup or object export (.MGR extension) or by logfile export (.MGL extension).
comparison	You can compare objects in two repositories, creating a file that will modify objects in one repository to make these equivalent to objects in the other repository. This comparison also allows you to list the differences between the contents of the two different objects.
compilation	Compilation is carried out after migration or customization. Compilation checks configuration of the environment concerned. When completed, processing for all users of this environment is speeded up. Metamodel compilation includes in parallel translation in the current language. You can also translate the metamodel into another language.

consolidation	Consolidation groups the updates from stand-alone workstations or remote sites (with Lan) and merges them in a reference site. After dispatch of the private workspaces of each of the users, the repository log is exported and reinitialized. The logfiles are imported into the reference repository, then this is recopied on each of the user sites.
database	See *repository*.
description	Descriptors allow you to create reports (MS Word) containing part of the contents of the repository. Descriptor for an object includes the object characteristics, to which can be added the characteristics of objects directly or indirectly linked to it. The readable format for each of the objects encountered is entered as text in Word for Windows. You can insert descriptors into reports (MS Word) or report templates (MS Word) or use them to produce reports. Descriptors can be created or modified using the **Hopex Power Studio** technical module.
desktop	The desktop specific to each user contains projects, diagrams, reports (MS Word), etc. handled by this user. The user has a different workspace in each of the repositories he/she accesses.
discard	Discarding the work performed in a private workspace cancels all modifications made since the last dispatch. All the work you have done since the beginning of the private workspace is lost. A warning message reminds the user of this. The user can request the discard of his/her private workspace from the **File > Discard** menu or at disconnection.
environment	An environment groups a set of *users*, the *repositories* on which they can work, and the *system repository*. It is where user private workspaces, users, system data, etc. are managed.
external reference	An external reference enables association of an object with a document from a source outside **Hopex**. This can relate to regulations concerning safety or the environment, legal text, etc. Location of this document can be indicated as a file path or Web page address via its URL (Universal Resource Locator).
functionality	A functionality is a means proposed by the software to execute certain actions (for example: the shapes editor and the descriptor editor are functionalities proposed as standard).
general UI access	General UI access defines if tools are available or not. By default, general UI accesses have value *A (A: Available, *: default value)

importing	Importing a command file, a backup file, or a logfile consists of applying the commands in the file to this new repository.
link	A link is an association between two types of object. There can be several possible links between two object types, for example: Source and Target between Org-Unit and Message.
link orientation	The two objects connected by a link do not normally have symmetrical roles. For example, to connect an operation to an organizational process with the **Hopex Power Supervisor** technical module, you must be authorized to modify the organizational process, since this action will modify its behavior. This action will not however modify the operation. You do not need authorization to modify the operation to create this link. The organizational process is said to be major for this link, the operation is minor. This characteristic is used by administration tools for object export, protection, object comparison and querying isolated objects.
lock	A lock is a logical tag assigned to an object to indicate that it is currently being modified by a user. Simultaneous access to an object by several users can also be checked. Locks apply to all types of object. When a user accesses an object to modify it, a lock is placed on the object. When a lock is placed on an object, another user is only able to view the object. This second user will not be able to modify the object until the first user dispatches his/her work, and the second user refreshes. This is done to avoid conflicts between the state of the object in the repository, and the obsolete view of the second user.
logfile	Logfiles contain all the actions performed by one or more users over a given period. The private workspace log contains all the changes made by a user in his/her private workspace. This logfile is used to update the repository when the user dispatches his work.
logfile export	Export of a logfile creates a command file from the logfile of user actions in a repository. You can keep this file and import it later into a repository. You can selectively export modifications made in the work repository, or those made to technical data (descriptors, queries, etc.) in the system repository.
login	A Login uniquely defines a user or user group. It can be assigned to only one Person or Person Group.

major	The major object in the link is the one whose nature changes with the presence or absence of the link. For example a process, defined as a succession of operations, is modified if you remove an operation. The process is then major for the link. If the objects are protected, you must have the correct authorization for modifying the major object in order to create or delete the link.
matrix	A matrix is a table comprising rows and columns containing objects from the repository. Matrices show the relationships between two sets of objects and allow you to create or delete links without having to open the diagrams themselves. For example, you can build a matrix showing the messages sent by the different org-units in a project.
MetaAssociation	see "link".
Metaclass	see object type
Metamodel	The metamodel defines the language used for describing a model. It defines the structure used to store the data managed in a repository The metamodel contains all the MetaClasses used to model a system, as well as their MetaAttributes and the MetaAssociations available between these MetaClasses. The metamodel is saved in the system repository of the environment. You can extend the metamodel to manage new MetaClasses. Repositories that exchange data (export, import, etc.) must have the same metamodel, otherwise certain data will be rejected or inaccessible.
minor	The minor object in a link is the one whose nature is not modified or only slightly modified by presence or absence of this link. For example, removing an operation from a process does not change characteristics of this operation. Therefore the process is minor in the link.
model	A model is a formal structure which represents the organization of a company, or its information system. In another sense, a model can be a template for reproducing objects with similar characteristics. This is the case for report templates (MS Word) and matrix models.
object	An object is an entity with an identity and clearly defined boundaries, of which status and behavior are encapsulated. In a **Hopex** repository, an object is often examined along with the elements composing it. For example, a Diagram contains Org-Units or Messages. A Project contains Diagrams, themselves containing Org-Units, Messages etc. Repository administration frequently requires consideration of consistent sets of objects. This is the case for object export, protection and object comparison.

object export	The export of one or several objects enables transfer of a consistent data set from a study to another repository. For example, export of objects from a project includes the project diagrams, together with the objects these diagrams contain, such as messages and org-units, as well as dependent objects. All the links between objects in this set are also exported.
Object type	An object type (or MetaClass) is that part of the database containing objects of a given type. The objects created are stored in the repository according to their type. Segments are used when searching for objects in the repository and when the metamodel is extended to include a new type of object. Example: message, org-unit, etc.
object UI access	Object UI access defines user rights on creation, reading, update, and deletion on these objects and their tools. By default, object UI accesses have value *CRUD (C: create, R: read, U: update, D: delete, *: default value).
person	A person is defined by his/her name and electronic mail address. A person can access **Hopex** once the administrator assigns him/her a login and a profile.
Person group	(Web Front-End specific) A person group groups persons in a group. These persons share the same connection characteristics.
private workspace	A private workspace is a temporary view of the repository in which the user is working before dispatching his/her work. This view of the repository is only impacted by the changes of the user, and does not include concurrent modifications made by other users. This private workspace exists until it is refreshed, dispatched or discarded. Note that a private workspace is kept when the user disconnects from the repository, unless the user indicates otherwise. A user can see modifications dispatched by other users of this repository without dispatching his/her own modifications. To do this, the user refreshes his/her private workspace. The system then creates a new private workspace, and imports the logfile of his/her previous modifications into it.
private workspace log	The private workspace log contains all modifications made by a user in his/her private workspace. It is applied to the repository at dispatch, then automatically reinitialized. This log is stored in the EMB private workspace file.

profile	A profile defines what a person can see or not see and do or not do in tools, and how he/she sees and can do it. The profile defines options, access rights to repositories and products, read/write and read-only rights on objects. All users with the same profile share these same options and rights. A user can have several profiles. A profile is available for all repositories in a single environment.
profile assignment	The profile assignment defines the following for each person: the repository concerned by the assignment, access rights to the repository, the validity period of the assignment, (optional, with access to the repository in read only) connection repository snapshot
protection	When several users work on the same project, it is important to provide them with the means to work on a new part of the project without inadvertently interfering with previous work. To do this, you can protect the objects concerned by assigning to them a writing access area (**Hopex Power Supervisor** technical module). It is then possible to connect these objects to others, if the link does not modify the nature of the object. The link orientation determines whether or not the nature of the object is affected.
publish	Dispatching your work allows the other users to see the changes you have made to the repository. They will see these when they open a new workspace, either by dispatching, refreshing or discarding their work in progress
query	A query allows you to select a set of objects of a given type using one or more query criteria. Most of the MEGA software functions can handle these sets. For example, you can use a query to find all enterprise org-units involved in a project.
reading access	see "reading access area".
reading access area	The user reading access area corresponds to the view the person or person group has of the repository: it defines objects that can be accessed by the person or person group.
reading access diagram	The reading access diagram enables definition of reading access areas and their hierarchical organization. This diagram also enables creation of users and their association with reading access areas.
reflexive link	A reflexive link is a link between two objects of the same type, for example: the link between projects that allows you to define sub-projects.

refresh	Refreshing his/her private workspace allows the user to benefit from changes dispatched by other users since creation of his/her workspace. In this case, it keeps repository modifications carried out by the user without making these available to other users. The system creates a new private workspace and the logfile of previous changes is imported into it.
reject file	When updating a repository (importing, restoring, dispatching), a reject file is created in order to store rejected commands. Rejected commands are stored with the reason for which they were rejected. These are found in the "MegaCrd.txt" file located in the environment folder.
reorganization	Reorganizing a repository consists of executing a logical backup of the repository, reinitializing it and reimporting the logical backup (without log).
report (MS Word)	Reports (MS Word) managed by **Hopex** are objects allowing you to transfer written knowledge extracted from the data managed by the software.
report (MS Word) element	A report (MS Word) element is the instancing of a report template (MS Word) element. It is the result, formatted in the word processing software, of execution of the query defined in the report template (MS Word) element.
report file	The environment report file, MegaCrdYYYYmm.txt (where YYYY and mm represent year and month of creation) indicates all administration operations (backup, export, restore, controls, etc.) carried out in the environment. The report file is stored in the user work folder associated with the environment system repository: SYSDB\USER\XXX\XXX.WRI where XXX is the user code.
report template (MS Word)	A report template (MS Word) is a structure with characteristics that may be reproduced when producing reports (MS Word). A report (MS Word) can be created and modified as many times as necessary; however to produce several reports (MS Word) of the same type, use of a report template (MS Word) is recommended. A report template (MS Word) provides the framework for the report (MS Word), which is completed with data from the repository when the report (MS Word) is created. A template contains the formatting, footers, headers and text entered in MS Word. It also contains report template (MS Word) elements that allow you to format data from the repository. It allows you to produce reports (MS Word) associated with main objects of the repository.

report template (MS Word) element	A report template (MS Word) element is the basic element of a report template (MS Word). It comprises a query which enables specification of objects to be described and a descriptor for their formatting. When creating a report (MS Word) from a report template (MS Word), each report template (MS Word) element is instanced by a report (MS Word) element.
Reporting Datamart	A Reporting Datamart is a replicated RDBMS Database from an Hopex repository content. The Reporting Datamart is made up of data selected at creation and synchronized on regular basis, to keep the Reporting Datamart updated according to the Hopex repository content. The Reporting Datamart feature is to be used as a source for any usage that needs Hopex data (for example: reporting).
repository	A repository is a storage location where Hopex manages objects, links, and inter-repository links. The main part is managed by a database system (SQL Server). The remainder is in a directory tree (content of Business Document versions, locks. The different users in the environment can access the repositories connected to it.
repository log	The repository log stores all the updates of users working in a repository. It is reinitialized at repository reorganization , or by selecting **Repository Log > Manage Repository and Object Log** in the repository pop-up menu. This logfile is stored in the .EMB file of the repository.
repository snapshot	A repository snapshot identifies an archived state of the repository. Creating a repository snapshot allows you to label important states in the repository life cycle. The repository archived states for which a snapshot exists are not deleted by repository cleanup mechanisms (Repository history data deletion).
restore	A physical restore consists of copying previously saved repository files.

saving	The work done in a session is saved when you request it, or when you exit. By default, the software executes an automatic save (the time interval between two saves is specified in the options: **Options > Repository > Data Saving > Background automatic save**). Messages appear asking you to confirm saving the changes you made in each open report template (MS Word) or report (MS Word) (except during the automatic save). It is recommended that you regularly save your work to avoid losing your work if your computer locks up or loses power.
session	A session is the period during which a user is connected to a repository. A session begins when the user authenticates and ends when he/she exits **Hopex**. Sessions and private workspaces can overlap. When you dispatch, refresh or discard a private workspace, a new private workspace is created in the same session. Conversely, a user can keep his/her private workspace when exiting a session.
set	A set is a collection of objects with common characteristics. For example, you can build a set of messages sent or received by a certain org-unit in the enterprise. These sets are usually built using queries, and can be handled by most functions of the software.
setting	A setting is a parameter of which value is only determined when the function with which it is associated is executed. You can use variables to condition a query (**Hopex Power Studio** technical module). When you execute this query, a dialog box asks you to enter these settings, with a box for each setting defined in the query.
site	A site groups together everything that is shared by all **Hopex** users on the same local network: the programs, standard configuration files, online help files, standard shapes, workstation installation programs, and version upgrade programs. The site is installed on a local network resource or on each workstation if you are working without a network connection.
snapshot	See *repository snapshot*
style	A style is a particular format applied to a paragraph of text in a word processor. Styles allow you to systematically apply formats such as fonts, margins, indents, etc. Several styles are available for report (MS Word) configuration. These styles have the M- prefix and are based on the M-Normal style that is similar to the Word Normal style. Note that the M-Normal style text is in blue. All these styles are contained in the Megastyl.dot style sheet.

SystDb	SystDb is a particular repository containing the metamodel and technical data (descriptors, Web site templates, queries, etc.). The metamodel and technical data are common to all repositories in the same environment. Definition of users and their rights are stored in this repository, essential for operation of the software.
system repository	See *SystDb*.
text	You can associate text with each object found when browsing object descriptors (**Hopex Power Studio** technical module). This text is formatted for MS Word. It presents what will be displayed for each of the objects in the generated report (MS Word). In the text you can insert the object name, its characteristics, and its comment. You can also insert the characteristics of other objects linked to it.
trace file	The trace file (Megaerr*.txt) is accessible via menu **Help** or from the **Hopex Server Supervisor** tool (available in the C:\ProgramData\MEGA\Hopex Application Server\<name of the HAS instance> folder). It traces all problems and errors encountered on the workstation. Technical support may ask you to check this file.
user	A user is a person with a login and at least one profile assigned. The code associated with the user is used to generate file names as well as a specific work folder for the user. By default at installation, Administrator (Login: System) and Mega (Login: Mega) persons enable administration of repositories and creation of new users.
work folder	Each user has a work directory in each repository that he/she uses. This directory is located in sub-directory User\XXX of the repository (XXX represents the user code).
workstation	A workstation is defined for each computer connected to the environment. A workstation contains programs and a configuration file that allow you to use **Hopex** on that machine.

writing access	see "writing access area".
Writing access area	A writing access area is a tag attached to an object to protect it from unwanted modifications. Each user is assigned a writing access area. There is a hierarchical link between writing access areas. A user can therefore only modify objects with the same or lower authorization level. The structure of writing access areas is defined in the writing access diagram. By default there is only one writing access area, "Administrator", to which all objects and users are connected. Writing access management is available with the **Hopex Power Supervisor** technical module.
writing access diagram	The writing access diagram is available if you have the **Hopex Power Supervisor** technical module. With this diagram, you can create users and manage their writing access rights for repositories and product functions. By default only one writing access area is defined, named "Administrator". Attached to it are the "Administrator" and "Mega" persons. This is the highest writing access level and it should normally be reserved for repository management. You cannot delete this writing access area.

Technical Articles (EN)

Description of MEGA Data Exchange XML Format

This technical article presents detailed explanation on various tags used in MEGA XML data exchange files.

Introduction to MEGA XML Data Exchange Format

MEGA allows you to import and export data in a standard XML exchange format.

Data contained in MEGA data XML documents can be described in the form of commands as for MGR, MGL or MGE documents. It can also be described in Content mode. Content mode now allows processing of sets of objects independently, free of any context (see Content exchange: <Content> tag, Content Mode, Hierarchical link in content mode).

Importing an XML document in a MEGA repository means executing or creating the commands it contains in this repository.

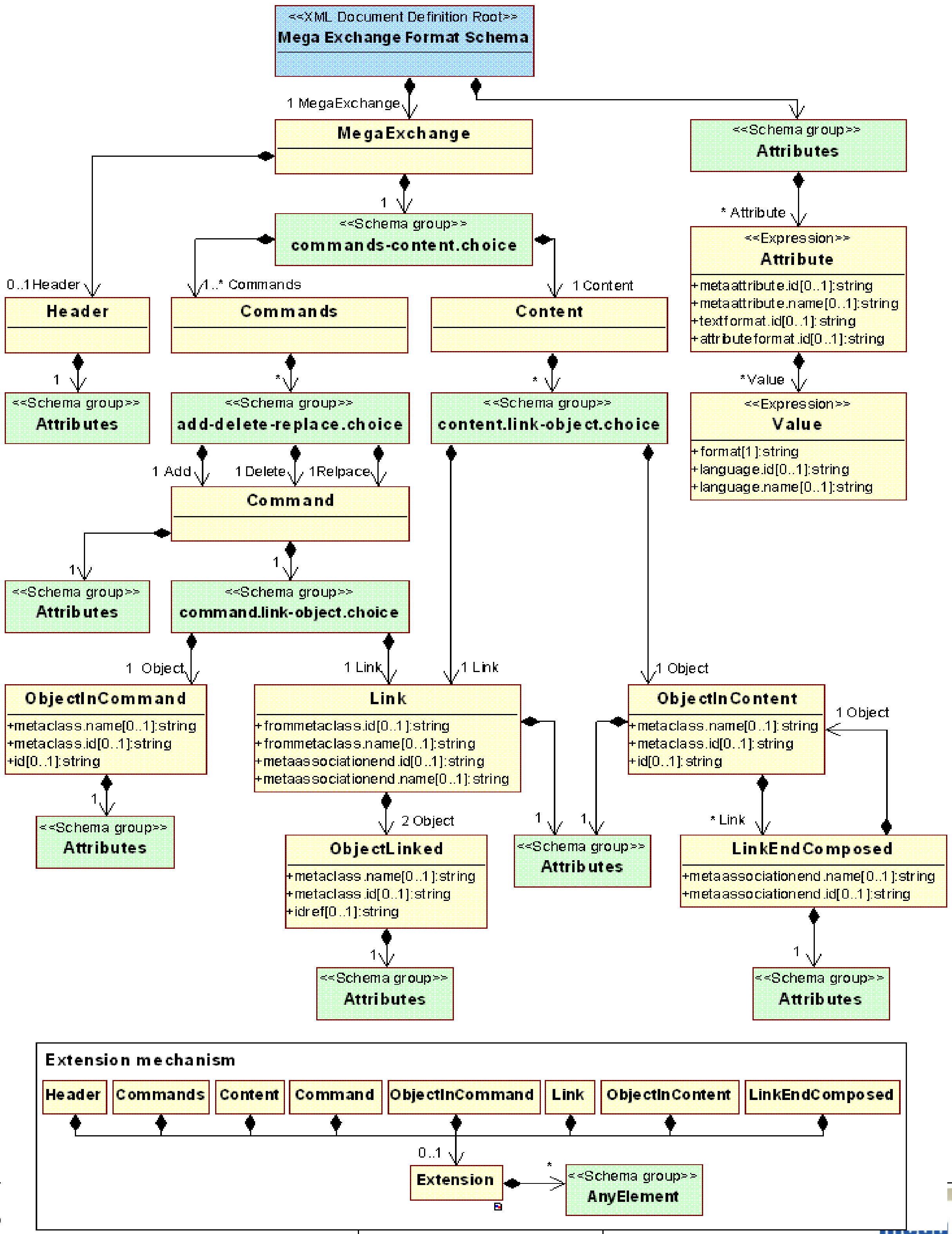
Exporting MEGA (objects, command) consists of dispatching this data in XML standard format. MEGA data exchange XML documents can also be created by software external to MEGA with a view to integrating the data they contain in a MEGA repository. Finally, they can be created manually.

You will find the XSD schema ("xmlmega.xsd") of MEGA data exchange XML files in the MEGA installation "system" directory.

An XSLT style sheet is also provided in the MEGA installation "system" directory, which presents MEGA data exchange XML documents. This style sheet is provided only as an example.

Note: References to documentation concerning the following are given in the glossary: MEGA metamodel, XML language, importing a data file, exporting a data file.

FORMAT XML SCHEMA MODEL



Tags in Brief

<MegaExchange>

This is the root element of the document. It contains all other tags.

(See Logical structure)

<Header>

This tag describes the characteristics of the document itself. We find here for example the exchange format version, the document creation date, the default exchange language, etc.

(See Logical structure)

<Attribute>

An <Attribute> tag enables expression of a command characteristic in an Add>, <Delete> or <Replace> tag, of an object in an <Object> tag, of a link in a <Link> tag, or of the exchange document itself in the <Header> tag.

(See Description of attribute values)

<Value>

This tag contains an attribute value. It appears only in <Attribute> tags. It serves to express a value in a particular format when this value can be expressed in several possible formats in an <Attribute> tag (for example, "internal" or "display" format).

(See Attribute value format)

<Commands>

This tag appears at root element level and enables expression of a set of commands.

(See Command exchange: <Commands> tag)

<Content>

This tag appears at root element level and enables inventory of a set of objects and links.

(See Content exchange: <Content> tag)

<Object>

An <Object> tag contains the description of an object: it is characterized by its type (MetaClass) and its attribute values. It can serve to describe or identify an object, for example an object to be connected at creation of a link.

(See Object Descriptions)

<Link>

A <Link> tag contains the description of a link: its type (MetaClass of the object to be connected and MetaAssociationEnd by which the second object is connected), identifications of the two objects to be connected and the link attribute values.

(See Link description)

<Add>

The <Add> tag is used in the <Commands> tag. It enables representation of an object or link creation command. Content is either an <Object> tag describing the object to be created, or a <Link> tag describing the link to be created.

(See Command exchange: <Commands> tag, Command Mode)

<Delete>

The <Delete> tag is used in the <Commands> tag. It enables representation of an object or link deletion command. Content is either an <Object> tag describing the object to be deleted, or a <Link> tag describing the link to be deleted.

(See Command exchange: <Commands> tag, Command Mode)

<Replace>

The <Replace> tag is used in the <Commands> tag. It enables representation of an object or link update command. Content is either an <Object> tag describing the object to be modified, or a <Link> tag describing the link to be modified.

(See Command exchange: <Commands> tag, Command Mode)

<Extension>

Various tags can contain the <Extension> tag.. When it is present in an element, this tag enables addition of supplementary information to the element (for example, addition of information concerning reasons for command reject). This information can be taken into account in a particular way according to tools used.

FORMAT IN DETAIL

In the remainder of this document, MEGA data exchange XML format will be referred to as "MEGA XML".

MEGA data exchange XML document structure

Physical structure

Like all XML documents, MEGA XML documents must start with XML declaration:

<?xml version="1.0"?>.

Encoding can be specified using the "encoding" attribute.

<?xml version="1.0" encoding="ISO-8859-1"?>

MEGA XML documents should be expressed in a code supported by the applications used to process them. For example, before importing a MEGA XML document it should be confirmed that MEGA can handle documents in the encoding proposed. MEGA import, like any XML analyzer, can basically handle input of Unicode encodings: UTF-8, UTF-16 little endian and big endian and ASCII. Other encodings such as ISO Latin1 FR can be used.

Note: see XML specifications for values to be specified for the "encoding" attribute (ref. Extensible Markup Language (XML) 1.0: http://www.w3.org/TR/2004/REC-xml-20040204/), and MEGA documentation to determine supported encodings.

Note: values that can be specified for the "encoding" attribute to identify the different encodings:

- "ISO-8859-1" corresponds to ISO Latin-1 FR encoding
- "UTF-8" corresponds to Unicode encoding on one or several bytes per character
- "UTF-16" corresponds to Unicode encoding on a multiple of two bytes per character

Logical structure

Like all XML documents, MEGA XML documents can have only one root tag, its name being *<MegaExchange>*.

A MEGA XML document firstly contains information relating to the document itself, such as the document creation date, the format version used or the attribute value expression language. This information is described in a *<Header>* tag.

In addition, information exchanged must be expressed either in a *<Commands>* tag, or in a *<Content>* tag

Example : MEGA data exchange XML document structure

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<MegaExchange>
  <Header>
    <!-- document information -->
    <Attribute metaattribute.name="format_version">Mega Xml Format Version
0.1</Attribute>
  </Header>
  <Commands>
    <!-- exchanged data -->
    <Add>
      <Object metaclass.name="Procedure" id="1">
        <Attribute metaattribute.name="Name">Procedure-1</Attribute>
      </Object>
    </Add>
  </Commands>
</MegaExchange>
```

Exchanged data description modes

Data exchanged via MEGA XML documents can be described in two modes:

- Either as a series of commands to be processed one after the other.
- Or as a repository content or sub-content, in other words as a set of objects.

Command exchange: <Commands> tag

Command series data expression mode is by use of the *<Commands>* tag.

It is this tag that contains command description tags in XML. The order of command description tags within the *<Commands>* tag is significant. It corresponds to the order in which commands must be processed by the MEGA XML document analysis tools. In fact, a command may not be validated unless preceding commands have been processed.

For example, if a command for creation of a link between two objects appears before the command for creation of one (or both) of the objects themselves, the link creation command is not valid from a logical viewpoint.

Example : Command exchange

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<MegaExchange>
  <Header>
    <!-- document information -->
    <Attribute metaattribute.name="xmg_version">XMG version 0.1</Attribute>
  </Header>
  <Commands>
    <!-- exchanged commands -->
    <Add>
      <Object metaclass.name="Procedure" id="1">
        <Attribute metaattribute.name="Name">Procedure-1</Attribute>
      </Object>
    </Add>
  </Commands>
</MegaExchange>
```

Content exchange: <Content> tag

Content description mode uses the *<Content>* tag. This tag contains a collection of objects and links.

These objects and links should be interpreted as independent data free of any context: they are not connected to a particular repository and do not require any other data in order to be significant (except for the metamodel describing them).

Data of an XML exchange document using description mode produces creation commands (creation of described objects and links) when the document is imported.

Order of appearance of tags describing data contained in the *<Content>* tag is significant. It corresponds to the order in which data must be processed by the MEGA XML document analysis tools.

If a link between two objects appears before one (or both) of the objects themselves, the link description is not valid from a logical viewpoint.

Example : Content description mode data exchange

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<MegaExchange>
  <Header>
    <!-- document information -->
    <Attribute metaattribute.name="xmg_version">XMG version 0.1</Attribute>
  </Header>
  <Content>
    <!-- exchanged data -->
    <Object metaclass.name="Procedure" id="1">
      <Attribute metaattribute.name="Name">Procedure-1</Attribute>
    </Object>
  </Content>
</MegaExchange>
```

In content description mode, we can explicitly show the hierarchical view of exchanged data structure. In fact, tags describing objects can themselves contain other tags describing contained objects from a logical viewpoint (for example a procedure containing operations).

This hierarchical description is valid only in content description mode.

Example : <Object> tag containing an <Object> tag

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<MegaExchange>
  <Header>
    <!-- document information -->
    <Attribute metaattribute.name="xmg_version">XMG version 0.1</Attribute>
  </Header>
  <Content>
    <!-- exchanged data -->
    <Object metaclass.name="Procedure" id="1">
      <Attribute metaattribute.name="Name">Procedure-1</Attribute>
      <!-- hierarchical link use  -->
      <Link metaassociationend.name="Contained operation">
        <!-- contained object -->
        <Object metaclass.name="Operation" id="2">
```

```
          <Attribute metaattribute.name="Name">Procedure-1</Attribute>
        </Object>
      </Link>
    </Object>
  </Content>
</MegaExchange>
```

Commands description: <Add>, <Delete>, <Replace> tags

Command tags can be used only in command mode. These tags are contained in the *<Commands>* tag explained in the chapter Command exchange: <Commands> tag.

The three available commands are:

- Creation command represented by an *<Add> tag.*
- Deletion command represented by a *<Delete> tag.*
- Modification command represented by a *<Replace> tag.*

Each of these three commands can be applied to an object or to a link: a tag representing a command can contain one tag only: *<Object>* or *<Link>*.

In command mode, *<Object>* tags cannot directly or indirectly contain other *<Object> tags.* : The hierarchical aspect of exchanged data logical structure cannot be represented by the hierarchical aspect of XML when data is exchanged in command mode.

Example : Object and link creation commands

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<MegaExchange>
  <Header>
    <!-- document information -->
    <Attribute metaattribute.name="xmg_version">XMG version 0.1</Attribute>
  </Header>
  <Commands>
    <!-- exchanged commands -->
    <Add>
      <Object metaclass.name="Class">
        <Attribute metaattribute.name="UML ClassName (Français)">Ma classe</Attribute>
      </Object>
    </Add>
    <Add>
      <Link frommetaclass.name="Package" metaassociationend.name="Contained class ">
        <Object metaclass.name="Package">
          <Attribute metaattribute.name="Name">My package</Attribute>
        </Object>
        <Object metaclass.name="Class">
          <Attribute metaattribute.name="Name">My org-unit</Attribute>
        </Object>
        <Attribute metaattribute.name="Order">9999</Attribute>
      </Link>
    </Add>
  </Commands>
</MegaExchange>
```

Object Descriptions

Objects are described by the *<Object> tag*..

The MetaClass of the object is identified by the name or MEGA absolute identifier (idabs in hexadecimal) of the MetaClass. The name of the MetaClass is specified by the value of the *"metaclass.name"* attribute of the *<Object> tag*, the idabs of the MetaClass is specified by the value of the *"metaclass.id"* attribute of the *<Object> tag*.

In addition, the objects themselves must be identified when we wish to make reference to them in the exchange document (see Object identification mechanisms).

Example : Object metaclass identification by metaclass name

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<MegaExchange>
  <Header>
    <!-- document information -->
    <Attribute metaattribute.name="xmg_version">XMG version 0.1</Attribute>
  </Header>
  <Content>
    <!-- exchanged data -->
    <Object metaclass.name="Procedure">
        <Attribute metaattribute.name="Name">Procedure-1</Attribute>
    </Object>
  </Content>
</MegaExchange>
```

Example : Object metaclass identification by metaclass idabs

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<MegaExchange>
  <Header>
    <!-- document information -->
    <Attribute metaattribute.name="xmg_version">XMG version 0.1</Attribute>
  </Header>
  <Content>
    <!-- exchanged data -->
    <Object metaclass.id="B1EDB25E2C1401BB">
      <Attribute metaattribute.name="Name">Procedure-1</Attribute>
    </Object>
  </Content>
</MegaExchange>
```

Command Mode

In commands description mode, an object is described in an *<Add>*, *<Delete>* or *<Replace>* tag, depending on whether we wish to create, delete or modify the object.

In this case, the *<Object>* tag describing the object can if necessary contain an *<Extension>* tag containing information not allowed for by MEGA XML format. It also contains *<Attribute>* tags specifying attribute values characterizing this object, these tags following the *<Extension>* tag if this is present.

Example : Object creation command by attribute specification

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<MegaExchange>
  <Header>
    <!-- document information -->
    <Attribute metaattribute.name="xmg_version">XMG version 0.1</Attribute>
  </Header>
  <Commands>
```

```
    <!-- exchanged commands -->
    <Add>
      <Object metaclass.name="Procedure" id="1">
        <Attribute metaattribute.name="Name">Procedure-1</Attribute>
        <Attribute metaattribute.name="Type-Procedure">General</Attribute>
      </Object>
    </Add>
  </Commands>
</MegaExchange>
```

An object described in a deletion command must make reference to an existing object recognized by the tool analyzing the MEGA XML document. The object description must therefore specify the value of its MEGA absolute identifier (object "_idabs" attribute) in an *<Attribute> tag*.

<u>Example : Object deletion command</u>

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<MegaExchange>
  <Header>
    <!-- document information -->
    <Attribute metaattribute.name="xmg_version">XMG version 0.1</Attribute>
  </Header>
  <Commands>
    <!-- exchanged commands -->
    <Delete>
      <Object metaclass.name="Procedure">
        <!-- object identifier attribute -->
        <Attribute metaattribute.name="_IdAbs">MnJgyaAJ0100</Attribute>
      </Object>
    </Delete>
  </Commands>
</MegaExchange>
```

An object described in a modification command must make reference to an existing object recognized by the tool analyzing the MEGA XML document. The object description must therefore specify the value of its MEGA absolute identifier (object "_idabs" attribute) in an *<Attribute> tag*. In addition to the *<Attribute>* tag specifying the MEGA absolute identifier value, we find *<Attribute>* tags giving the values of attributes to be changed for the object concerned.

<u>Example : Object modification command</u>

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<MegaExchange>
  <Header>
    <!-- document information -->
    <Attribute metaattribute.name="xmg_version">XMG version 0.1</Attribute>
  </Header>
  <Commands>
    <!-- exchanged commands -->
    <Replace>
      <Object metaclass.name="Procedure">
        <!-- object identifier attribute -->
        <Attribute metaattribute.name="_IdAbs">MnJgyaAJ0100</Attribute>
        <!-- modified attribute -->
        <Attribute metaattribute.name="Code-Procedure">XYZ</Attribute>
      </Object>
    </Delete>
  </Commands>
</MegaExchange>
```

Content Mode

In content description mode, *<Object>* tags describing objects are contained in the *<Content> tag*.

The *<Object>* tag describing the object can if necessary contain an *<Extension>* tag containing information not allowed for by MEGA XML format. It also contains *<Attribute>* tags specifying attribute values characterizing this object. These tags follow the *<Extension>* tag if this is present.

In addition, in content description mode, objects can be described as containing other objects. The *<Object>* tag can therefore indirectly contain other *<Object>* tags describing contained objects (see <u>Hierarchical link in content mode</u>).

The same considerations apply to *<Object>* tags representing contained objects, which can themselves contain *<Object> tags*.

Example: Object description in content mode

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<MegaExchange>
  <Header>
    <!-- document information -->
    <Attribute metaattribute.name="xmg_version">XMG version 0.1</Attribute>
  </Header>
  <Content>
    <!-- exchanged data -->
    <Object metaclass.name="Procedure" id="1">
      <Attribute metaattribute.name="Name">Procedure-1</Attribute>
    </Object>
  </Content>
</MegaExchange>
```

Object identification mechanisms

We need to make reference to objects internal or external to the document within the framework of:

- An object modification command
- An object modification command
- The link between one object and another

We distinguish objects described in the document from those not described in the document but which are the object of a command:

- Objects internal to the document: these are described in the document and can be referenced in the document via use of *"id"* and *"idref"* attributes of the *<Object> tag*.
- Objects external to the document: these are not described in the document but can be the target of a command. These should be recognized by the tool processing the document and are identified by the "_idabs" attribute of the object.

Object identification therefore uses either an identifier internal to the document or a MEGA absolute identifier.

Identification internal to the document is by use of the *"id"* attribute of the *<Object>* tag. The value of the *"id"* attribute must be unique throughout the document with no other constraint on form; it is an ID type attribute (ID type is defined in XML1.0 specifications). Reference to a document object is by using the *"idref"* attribute of the *<Object>* tag, which must therefore have the same value as the *"id"* attribute of the *<Object>* tag representing the referenced object. For example, within the framework of a link, the *<Object>* tags referencing linked objects can each have an *"idref"* attribute.

External identification is by definition of an *<Attribute>* tag defining the value of its MEGA absolute identifier (object "_idabs" attribute). Objects can be created with an "_idabs" attribute by use of the *<Attribute>* tag (see Description of attribute values). Similarly, *<Object>* tags referencing objects, as in the *<Delete>*, *<Replace>* or *<Link>* tags, can identify objects by their MEGA absolute identifier, and in this case they contain an *<Attribute>* tag specifying the value of the object "_idabs" attribute.

Example : Identification by <Object> tag id attribute

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<MegaExchange>
  <Header>
    <!-- document information -->
    <Attribute metaattribute.name="xmg_version">XMG version 0.1</Attribute>
  </Header>
  <Content>
    <!-- exchanged data -->
    <Object metaclass.name="Procedure" id="1">
      <Attribute metaattribute.name="Name">Procedure-1</Attribute>
    </Object>
    <Object metaclass.name="Procedure" id="2">
      <Attribute metaattribute.name="Name">Procedure-2</Attribute>
    </Object>
    <Link frommetaclass.name="Procedure" metassociationend.name="Next procedure">
      <Object metaclass.name="Procedure" idref="1"/>
      <Object metaclass.name="Procedure" idref="2"/>
    </Link>
  </Content>
</MegaExchange>
```

Example : Identification by idabs

```
<?xml version="1.0" encoding="ISO-8859-1"?>
```

```
<MegaExchange>
  <Header>
    <!-- document information -->
    <Attribute metaattribute.name="xmg_version">XMG version 0.1</Attribute>
  </Header>
  <Commands>
    <!-- exchanged commands -->
    <Add>
      <Object metaclass.name="Procedure">
        <Attribute metaattribute.name="Name">Procedure-1</Attribute>
        <!-- object identifier attribute -->
        <Attribute metaattribute.name="_IdAbs">MnJgyaAJ0100</Attribute>
      </Object>
    </Add>
    <Delete>
      <Object metaclass.name="Procedure">
        <!-- object identifier attribute -->
        <Attribute metaattribute.name="_IdAbs">MnJgyaAJ0100</Attribute>
      </Object>
    </Delete>
  </Commands>
</MegaExchange>
```

Link description

Links are described by the *<Link> tag.*.

The MetaAssociation of the link is identified by the name or MEGA absolute identifier (idabs in hexadecimal form) of the MetaAssociationEnd by which the objects are connected.

The name of the MetaAssociationEnd is specified by the value of the *"metaassociationend.name"* attribute of the *<Link>* tag.

The idabs of the MetaAssociationEnd is specified by the value of the *"metaassociationend.id"* attribute of the *<Link>* tag.

One of the attributes *"metaassociationend.name"* or *"metaassociationend.id"* must be present to identify the link. Both can be present simultaneously.

Hierarchical link in content mode

In content mode, a link can be represented hierarchically. The *<Object>* tag representing the first object therefore contains a *<Link>* tag which itself contains the *<Object>* tag describing the connected object.

The *<Link>* tag serves to specify the MetaAssociationEnd by which the second object is connected. To do this, we must therefore specify the *"metaassociationend.name"* attribute or the *"metaassociationend.id"* attribute.

In addition, the *<Link>* tag can contain *<Attribute>* tags describing link attribute values. These *<Attribute>* tags are placed before the *<Object>* tag.

Example: Hierarchical ink description in content mode

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<MegaExchange>
  <Header>
    <!-- document information -->
    <Attribute metaattribute.name="xmg_version">XMG version 0.1</Attribute>
  </Header>
  <Content>
    <!-- exchanged data -->
    <Object metaclass.name="Procedure" id="1">
      <Attribute metaattribute.name="Name">Procedure-1</Attribute>
      <!-- hierarchical link use  -->
      <Link metaassociationend.name="Contained operation">
        <!-- link attributes -->
        <Attribute metaattribute.name="Order">1</Attribute>
        <!-- contained object -->
        <Object metaclass.name="Operation" id="2">
          <Attribute metaattribute.name="Name">Procedure-1</Attribute>
        </Object>
      </Link>
    </Object>
  </Content>
</MegaExchange>
```

Other links

Links that are not hierarchical links can be used in both command mode command tags (<*Add*>, <*Delete*>, <*Replace*>) and in the content mode <*Content*> tag. These links make reference to two connected objects describing but not referencing these objects. The connected objects can be external to the document.

A link between two objects is described by the MetaAssociationEnd by which the second object is connected to the first. To do this, the <*Link*> tag representing the link contains either the MetaAssociationEnd idabs value in hexadecimal form in the *"metaassociationend.id"* attribute, or the MetaAssociationEnd name value in the *"metaassociationend.name"* attribute. In the latter case, the MetaClass of origin of the MetaAssociationEnd must be specified, ie. the MetaClass of the first object. This is done via the *"frommetaclass.id"* or *"frommetaclass.name"* attributes of the <*Link*> tag, specifying either the MetaClass idabs in hexadecimal form or the MetaClass name.

In addition, the <*Link*> tag contains two <*Object*> tags that reference the two connected objects. Each of these has a *"metaclass.id"* attribute specifying either the MetaClass idabs in hexadecimal form, or a *"metaclass.name"* attribute specifying its name. The object is referenced either by specifying the *"idref"* attribute value of the <*Object*> tag in the <*Link*> tag, which should be equal to the *"id"* attribute value of the <*Object*> tag describing the object referenced in the document, or by adding an <*Attribute*> tag specifying the "_idabs" attribute value of the referenced object, which in this case can be an object not described in the document.

Finally, the <*Link*> tag can contain <*Attribute*> tags describing link attribute values. These <*Attribute*> tags are placed after the two <*Object*> tags.

Example : Description of a link between two objects of the document: use of “idref” attribute

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<MegaExchange>
  <Header>
    <!-- document information -->
    <Attribute metaattribute.name="xmg_version">XMG version 0.1</Attribute>
  </Header>
  <Content>
    <!-- exchanged data -->
    <Object metaclass.name="Package" id="1">
      <Attribute metaattribute.name="Name">Package-1</Attribute>
    </Object>
    <Object metaclass.name="Package" id="2">
      <Attribute metaattribute.name="Name">Package-2</Attribute>
    </Object>
    <Link frommetaclass.name="Package" metaassociationend.name="Referenced package">
      <!-- link source object -->
      <Object metaclass.name="Package" idref="1"/>
      <!-- link destination object -->
      <Object metaclass.name="Package" idref="2"/>
      <!-- link attributes -->
      <Attribute metaattribute.name="Order">9999</Attribute>
    </Link>
  </Content>
</MegaExchange>
```

Example : Description of a link between two objects by idabs

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<MegaExchange>
  <Header>
    <!-- document information -->
    <Attribute metaattribute.name="xmg_version">XMG version 0.1</Attribute>
  </Header>
  <Commands>
    <!-- exchanged commands -->
    <Add>
      <Link frommetaclass.name="Package" metaassociationend.name="Referenced package">
        <!-- link source object -->
        <Object metaclass.name="Package">
          <Attribute metaattribute.name="_idabs">ODiTpSNApa10</Attribute>
        </Object>
        <!-- link destination object -->
        <Object metaclass.name="Package">
          <Attribute metaattribute.name="_idabs">dCyP0ZtmxSA8</Attribute>
        </Object>
        <!-- link attributes -->
        <Attribute metaattribute.name="Order">9999</Attribute>
      </Link>
    </Add>
  </Commands>
</MegaExchange>
```

Description of attribute values

Attribute values are specified in *<Attribute>* tags. The following tags can contain attribute values: *<Header>*, *<Link>*, *<Object>*, *<Add>*, *<Delete>*, *<Replace>*.

In an *<Attribute>* tag, the valuated characteristic is identified by the *"metaattribute.name"* attribute, which determines its name, or by the *"metaattribute.id"* attribute, which determines its idabs in hexadecimal form.

The value is directly contained in the *<Attribute>* tag..

Example : Description of an attribute value

```
<MegaExchange>
  <Header>
    <!-- document information -->
    <Attribute metaattribute.name="xmg_version">XMG version 0.1</Attribute>
  </Header>
  <Commands>
    <!-- exchanged commands -->
    <Add>
      <Object metaclass.name="Package">
        <!-- object attributes -->
        <Attribute metaattribute.name="_idabs">ODiTpSNApa10</Attribute>
        <Attribute metaattribute.name="nom">Packagee-1</Attribute>
      </Object>
    </Add>
  </Commands>
</MegaExchange>
```

Attribute value format

Direct values in *<Attribute>* tags are expressed in the format specified by the *<Attribute>* tag representing the *"default_format"* attribute of the *<Header>* tag.

If this attribute is not specified, default format is left to the tool analyzing the document.

Example : Attribute value default format

```
<MegaExchange>
  <Header>
    <!-- document information -->
    <Attribute metaattribute.name="xmg_version">XMG version 0.1</Attribute>
    <Attribute metaattribute.name="default_format">internal</Attribute>
  </Header>
  <Commands>
    <!-- exchanged commands -->
    <Add>
      <Object metaclass.name="Package">
        <!-- object attributes -->
        <Attribute metaattribute.name="_idabs">ODiTpSNApa10</Attribute>
        <Attribute metaattribute.name="nom">Packagee-1</Attribute>
      </Object>
    </Add>
  </Commands>
</MegaExchange>
```

The *<Attribute>* tag enables attribute value specification in several formats. The different forms that the attribute value can take are each contained in a *<Value>* tag, the format being specified by the *<Value>* tag *"format"* attribute.

Example : Attribute value expressed in several formats

```
<MegaExchange>
  <Header>
    <!-- document information -->
    <Attribute metaattribute.name="xmg_version">XMG version 0.1</Attribute>
  </Header>
  <Commands>
    <!-- exchanged commands -->
    <Add>
      <Object metaclass.name="Application">
        <!-- object attributes -->
        <Attribute metaattribute.name="_idabs">ODiTpSNApa10</Attribute>
        <Attribute metaattribute.name="name">Application-1</Attribute>
        <Attribute metaattribute.name="MMI">
          <Value format="internal">W</Value>
          <Value format="display">Windowed</Value>
        </Attribute>
      </Object>
    </Add>
  </Commands>
</MegaExchange>
```

Translatable attributes and attribute value expression language

In the MEGA repository, for a given translatable attribute, there are as many attributes as there are languages into which the attribute is translatable. For example, for the "Name" attribute there are corresponding "Name (English)","Name (French)", "Name (Dutch)", etc. attributes.

In a MEGA XML document, the *"language.id"* and "language.name" attributes of the *<Value>* tag enable specification of the language in which the attribute value is expressed.

Translatable attributes are therefore identified by the name or absolute identifier of the root attribute using the *"metaattribute.name"* and *"metaattribute.id"* attributes of the *<Attribute>* tag, and the language in which the value is expressed using the *"language.name"* and *"language.id"* attributes.

For value expression language to be specified, at least one of the two attributes *"language.name"* or *"language.id"* must be present in the *<Value>* tag.

When neither the *"language.name"* nor the *"language.id"* attribute is present, the value is expressed in the language specified by the *<Attribute>* tag representing the *"model_default_language"* attribute of the *<Header>* tag.

<u>Example : Attribute value expressed in default language</u>

```
  <MegaExchange>
    <Header>
      <!-- document information -->
      <Attribute metaattribute.name="model_default_language"> Français</Attribute>
    </Header>
    <Commands>
      <!-- exchanged commands -->
      <Add>
        <Object metaclass.name="Application">
          <!-- object attributes -->
          <Attribute metaattribute.name="_idabs">ODiTpSNApa10</Attribute>
          <Attribute metaattribute.name="name">Application-1</Attribute>
        </Object>
      </Add>
    </Commands>
  </MegaExchange>
```

Example : Attribute value expressed in several languages

```
<MegaExchange>
  <Header>
    <!-- document information -->
    <Attribute metaattribute.name="model_default_language"> Français</Attribute>
  </Header>
  <Commands>
    <!-- exchanged commands -->
    <Add>
      <Object metaclass.name="Application">
        <!-- object attributes -->
        <Attribute metaattribute.name="_idabs">ODiTpSNApa10</Attribute>
        <Attribute metaattribute.name="name">
          <Value>Invoicing</Value>
          <Value language.name="English">Invoicing</Value>
        </Attribute>
      </Object>
    </Add>
  </Commands>
</MegaExchange>
```

MEGA XML format extensions

MEGA XML format includes an extension mechanism enabling addition of information to different format tags.

A format extension is created by use of the *<Extension>* tag. The extension tag can contain any information expressed in XML.

Information contained in the *<Extension>* tag is not part of the format. It cannot be taken into account by a tool using MEGA XML format.

The following tags can contain an extension tag: *<Header>*, *<Link>*, *<Object>*, *<Add>*, *<Delete>*, *<Replace>*.

Example of use of the *<Extension>* tag by MEGA

The *<Extension>* tag is used by the MEGA import function which creates a report of commands analyzed at import. The report contains commands analyzed in MEGA XML format indicating if they have been accepted, rejected or contain warnings. Therefore each MEGA XML format command in the report can contain an *<Error>* tag or *<Warning>* tags in the *<Extension>* tag of the command concerned. These tags enable appreciation of the import result, but are not to be taken into account in another context.

Example 1 : Use of the *<Extension>* tag by MEGA

```
<MegaExchange>
  <Header>
    <!-- document information -->
  </Header>
  <Commands>
    <!-- exchanged commands -->
    <Add>
      <Object metaclass.name="Application">
        <Extension>
          <Error code="XXXXXXXX">There is no « metaattribute » with « color » value as
« name »<Error>
        </Extension>
        <!-- object attributes -->
        <Attribute metaattribute.name="_idabs">ODiTpSNApa10</Attribute>
        <Attribute metaattribute.name="name">Invoicing</Attribute>
        <Attribute metaattribute.name="color">blue</Attribute>
      </Object>
    </Add>
  </Commands>
</MegaExchange>
```

Attributes of MEGA XML documents used by MEGA

A MEGA XML document exported from a MEGA repository contains a certain number of attributes that describe general elements relating to the document itself. These are described in this section.

Document attributes are located in the *<Header>* tag of the document.

Example: Document attributes

```
<?xml version="1.0" encoding="UTF-8"?>
<MegaExchange>
  <Header>
```

```
    <Attribute metaattribute.id="27C729AE3F4E004A"
metaattribute.name="exchange_format_version">
      <Value format="internal">000100</Value>
      <Value format="display">MEGA XML version 0.1</Value>
    </Attribute>
    <Attribute metaattribute.id="27C729AE3F4E0050" metaattribute.name="mega_version">
      <Value format="internal">25856</Value>
      <Value format="display">MEGA Version 2005 Pre release</Value>
    </Attribute>
    <!-- ... -->
  </Header>
  <Commands>
    <!-- ... -->
  </Commands>
</MegaExchange>
```

Document attributes list

- exchange_format_version : this attribute indicates the version of MEGA XML exchange format.
- mega_version : this attribute indicates the version of MEGA used at export.
- metamodel_language : this attribute specifies the language used to identify types in the attributes of *"metaclass.name"*, *"metaattribute.name"*, *"metaassociationend.name"* tags.
- model_default_language : this attribute specifies the default language used for translatable attribute values of objects and links when the language has not been specified by *"language.name"* or *"language.id"* attributes of the *<Value>* tag.
- author : the value of the *author* attribute identifies the user that executed export.
- source_database : the value of the *source_database* attribute identifies the repository from which export was executed.
- creation_date : specifies MEGA XML document creation date.

Glossary

XML attribute:

An XML attribute is unstructured text data contained in an XML tag.

Example: <attribute tag="value"/>

MEGA object attribute:

Object characteristics are called attributes. For a given object, attributes can take a value of which the type is defined for the attribute.

XML tag:

XML tags are syntax elements that structure XML documents. They mark opening and closing of XML document data definition.

Example: <tag>…</tag>

MEGA command:

The various actions possible that modify a MEGA repository are called commands. In particular these include creation, deletion and modification of objects and links.

XML element :

An XML element is structured data comprising an opening tag, attributes of this tag, a closing tag and text data and tags contained between opening and closing tags.

Example: <attribute tag="value">text<sub-tag/></tag>

File export

MEGA repository data can be exported in a file. From a MEGA repository we can export : objects (exported data describing exported objects), transactions (exported data describing transaction commands) or the complete repository (exported data describing all repository objects, this procedure also being known as logical backup). The various export functions are described in the "MEGA Help" CHM file and in the "MEGA Administration" PDF user guide in the following sections: "Administration > Managing Transactions > Managing Updates > Exporting the Transaction Logfile", "Administration > Managing Objects > Exporting MEGA objects", "Administration > Managing Repositories > Reorganizing a Repository > Logical Backup of a Repository".

MEGA APIs enable file export using Automation interfaces. Data export and logical backup functions using API are covered in the "MEGA Help" CHM file in the section "API > API - Reference Guide > Administration > MEGA Database" and in the "MEGA API" PDF user guide in the similarly named section.

IdAbs

Absolute identifier of data stored in the MEGA repository, idabs is generally represented in hexadecimal or base 64 form. The idabs is for example an object attribute (attribute "_idabs").

File import

Importing an XML document in a MEGA repository means executing or creating the commands it contains in this repository. MEGA can import files of type MGR, MGL, MGE and MEGA XML. The import function is covered in the "MEGA Help" CHM file in the section "Administration > Managing Repositories > Reorganizing a Repository > Updating a Repository (Importing)" and in the "MEGA Administration" PDF user guide in the similarly named section.

MEGA APIs enable file import using Automation interfaces. The API import function is covered in the "MEGA Help" CHM file in the section "API > API - Reference Guide > Administration > MEGA Database" and in the "MEGA API" PDF user guide in the similarly named section.

MEGA links :

For a given repository object, links enable definition of connected repository objects.

MEGA MetaAssociation :

This term indicates relationships existing between repository object types. MetaAssociations define repository links.

MEGA MetaClass :

This term indicates object types stored in the repository. MetaClasses define repository objects.

MEGA metamodel :

This comprises all MetaClasses and MetaAssociations defining objects that can be created and handled in the MEGA repository. The MEGA metamodel is covered in the "MEGA Help" CHM file in section "Metamodel and Glossaries > Metamodel".

MEGA object

Data stored in a MEGA repository are called repository objects. An object comprises attributes and can be connected to other repository objects. In addition, every object is identified uniquely by an attribute called the MEGA absolute identifier or idabs.

MEGA XML

Common name for MEGA data exchange XML format.

XML

Extendable tag language (eXtensible Markup Language), XML is a metalanguage describing languages of tagged hierarchical structure. XML specifications provide syntax basics of XML languages (ref. Extensible Markup Language (XML) 1.0: http://www.w3.org/TR/2004/REC-xml-20040204/).

REPORTING DATAMART

1. REPORTING DATAMART OVERVIEW

With the **Reporting Datamart** feature you can create a **Reporting Datamart**, which is a replicated RDBMS Database from an HOPEX repository content.

The **Reporting Datamart** is made up of data selected at creation and synchronized on regular basis, to keep the **Reporting Datamart** updated according to the HOPEX repository content.

The purpose of the **Reporting Datamart** feature is to be used as a source for any usage that needs HOPEX data (for example: reporting).

The **Reporting Datamart** feature is available with **HOPEX Datamart** license.

2. REPORTING DATAMART DESCRIPTION

2.1. Structure

The **Reporting Datamart** structure is initialized with the RDBMS database standard structure.

Tables are created and fed with the HOPEX repository and SystemDb repository data. If there is no data to be exported to a table, this table remains empty, so that the **Reporting Datamart** structure is always consistent. This ensure the user that any of his queries run properly on the **Reporting Datamart**.

Reporting Datamart table and column names are in English language.

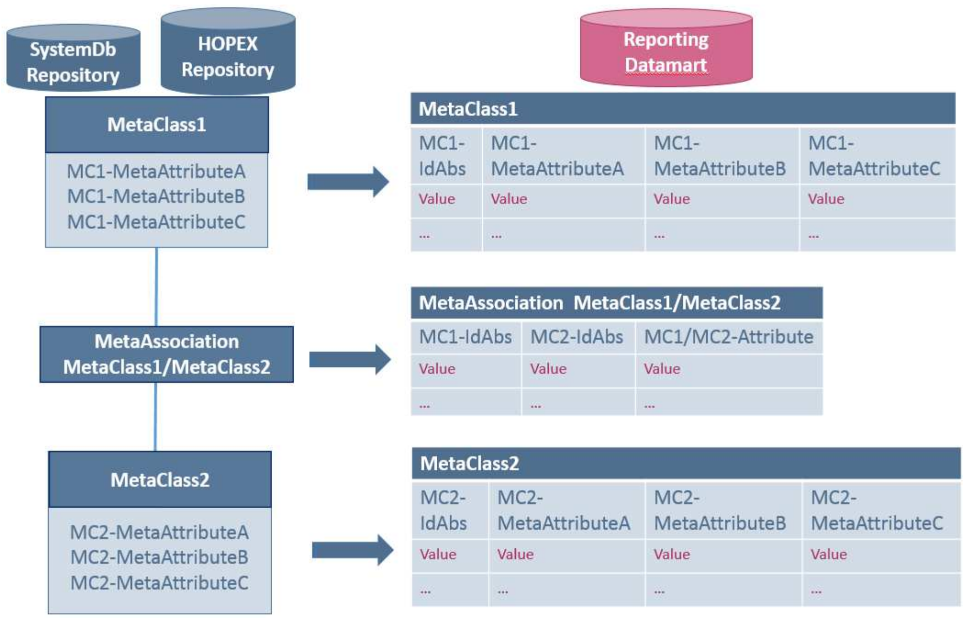
Reporting Datamart data is stored in the language selected at **Reporting Datamart** creation.

SQL database-based Reporting Datamart

For an SQL database-based **Reporting Datamart** the rule is as follows:

- an SQL Database Table is created for each HOPEX MetaClass
- an SQL Database Table is created for each HOPEX MetaAssociation
- an SQL Database Column is added in the SQL database Table for each HOPEX MetaAttribute

  See Reporting Datamart Detailed description for a detailed description.



2.2. Data

HOPEX repository

Most of the HOPEX repository data is exported to the **Reporting Datamart**, including:

- MetaClasses, MetaAssociations, and MetaAttributes
- diagrams exported in BLOB column of their object
- calculated MetaAttributes
- comments, in html format (instead of RTF format)

Logs and object history are not exported.

SystemDb repository

Few data is exported from the SystemDb repository:

- Logins
- Persons (system)
- Person groups

2.3. Content

With the **Reporting Datamart,** data is filtered so that the data amount is reduced to the data needed only:

- Only the last dispatched version of each object is kept in the Reporting Datamart.
- Data is filtered at Reporting Datamart creation according to a user rights (confidentiality) and his profile permissions.
- Data name is displayed (not Idabs) for an easy reading.
- Data is in the **Reporting Datamart** language.

  Note: if you need the **Reporting Datamart** data in multiple languages, you need to create as many **Reporting Datamart** as languages needed.

2.4. Excluding MetaAttribute values from the Reporting Datamart

In some cases, you might not need to export some specific MetaAttribute values in your **Reporting Datamart**.

To exclude MetaAttribute values from the Reporting Datamart:

1) In HOPEX, access the MetaAttribute properties.
2) Click the **Characteristics** tab, then **Advanced** subtab.
3) Click the Extended Properties field arrow and select Exclude from Reporting Datamart.

3. CREATING AND INITIALIZING A REPORTING DATAMART

You can create and initialize a **Reporting Datamart** from an RDBMS HOPEX repository of your HOPEX Environment as follows:

Step	**Reporting Datamart**		**Description**	**See**
1	Creation		- Reporting Datamart data language definition - User and profile selection	Creating a Reporting Datamart
2	Initialization	Structure (optional)	Creation of all the tables and columns according to the HOPEX repository metamodel	Initializing the Reporting Datamart structure
		Data	Creation and feeding of all the Reporting Datamart tables, columns and rows according to the HOPEX repository content	Initializing the Reporting Datamart data

Note: you can also initialize the Reporting Datamart using APIs, see HOPEX power Studio > Initializing and synchronizing a Reporting Datamart.

3.1. Creating a Reporting Datamart

At **Reporting Datamart** creation you define:

- the **Reporting Datamart** storage
- the data language of the **Reporting Datamart**

  Note: not translated data appears in its Idabs format.
- the user and its profile

  They both define and filter the HOPEX repository data according to their access rights (confidentiality: reading access areas of object occurrences and of linked objects for link occurrences) and permissions.

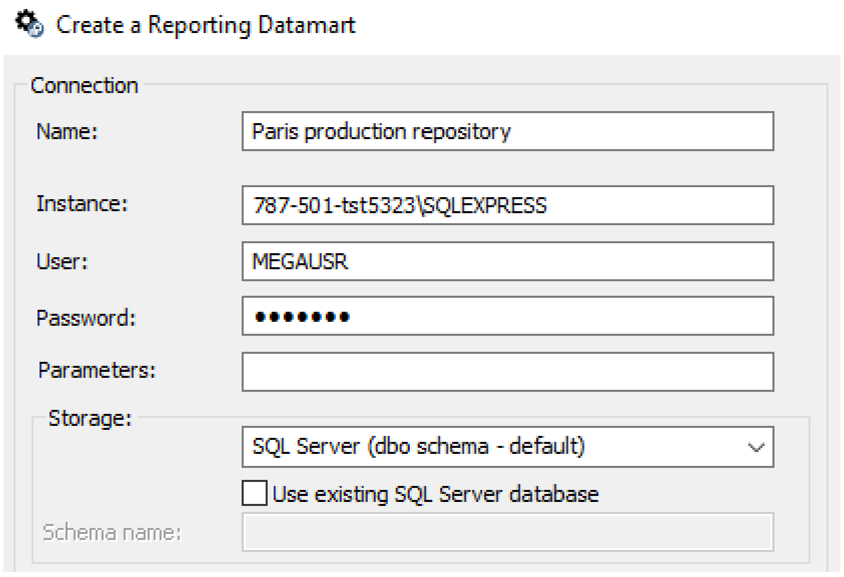
To create a Reporting Datamart (SQL Server format) from a HOPEX repository:

1) Launch **HOPEX Administration** application.
2) Connect to the environment from which you want to replicate the data.
3) In the **Repositories** folder, expand the repository folder concerned.
4) Right-click **Reporting Datamart** and select **New**.

   The Create a Reporting Datamart window opens.
5) In the **Connection** pane:
   a) In the **Name** field, enter your **Reporting Datamart** name.
   b) (If needed) Modify the HOPEX repository **Instance** set by default as well as its connection parameters (**User**, **Password**, **Parameters)**, see *RDBMS Repository Installation Guide* for detailed information.
   c) In the **Storage** pane, use the drop-down menu to select the **Reporting Datamart** storage type: “SQL Server (dbo schema - default)”.

d) (If you do not have the right to create a database) Select **Use existing SQL Server database**.

Note: if you selects **Use existing SQL Server database**, in the **Name** field you must enter the name of the existing database you want to use.



Create a Reporting Datamart

Connection

Name:	Paris production repository
Instance:	787-501-tst5323\SQLEXPRESS
User:	MEGAUSR
Password:	•••••••
Parameters:	
Storage:	SQL Server (dbo schema - default)
	☐ Use existing SQL Server database
Schema name:	

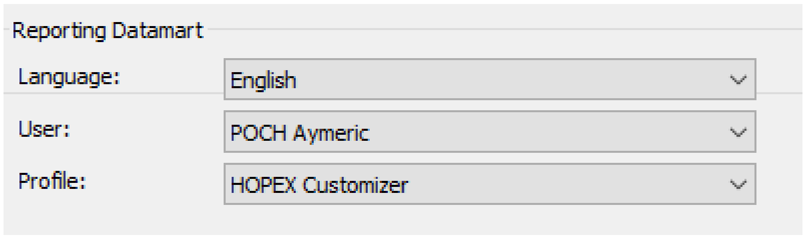
6) In the Reporting Datamart pane:

a) In the **Language** field, select the data language of the **Reporting Datamart**.

Note: select a language in which the data is available, so that you can read it.

b) In the **User** field, select the user that defines the access to HOPEX repository.

c) In the **Profile** field, select the profile that defines the access to HOPEX repository.

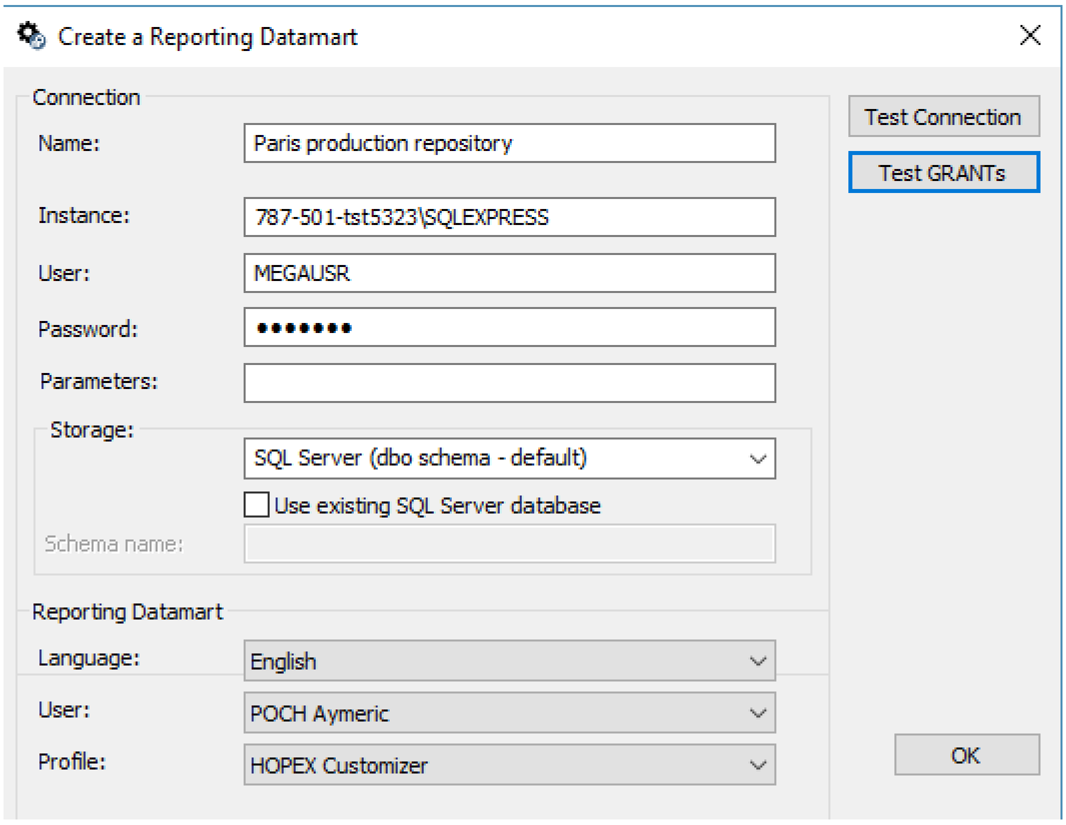


Reporting Datamart

Language:	English
User:	POCH Aymeric
Profile:	HOPEX Customizer

7) Click Test Connection.

8) Click Test GRANT's.


Create a Reporting Datamart

Connection

Name:	Paris production repository
Instance:	787-501-tst5323\SQLEXPRESS
User:	MEGAUSR
Password:	•••••••
Parameters:	
Storage:	SQL Server (dbo schema - default)
	☐ Use existing SQL Server database
Schema name:	

Reporting Datamart

Language:	English
User:	POCH Aymeric
Profile:	HOPEX Customizer

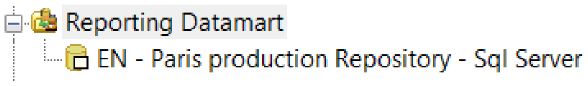
9) Click **OK**.

The Reporting Datamart is created.

The **Reporting Datamart** is added in the Reporting Datamart folder with the following format:

```
<Language> – <Reporting Datamart name> – <Storage type>
```

```
Example: EN – Paris production Repository – Sql Server
```


Reporting Datamart
EN - Paris production Repository - Sql Server

3.2. Initializing the Reporting Datamart structure

Once the **Reporting Datamart** is created you can initialize the structure with the HOPEX repository metamodel.

All the tables and columns are created even if there is no data to feed them. The table remains empty, so that the **Reporting Datamart** structure is always consistent. This ensures the user that any of his queries run properly on the **Reporting Datamart**.

To initialize the Reporting Datamart structure from the HOPEX repository:

1) Launch **HOPEX Administration** application.
2) Connect to the environment concerned.
3) In the **Repositories** folder, expand the repository folder concerned.
4) In the Reporting Datamart folder, right-click the Reporting Datamart concerned and select Initialize > Structure.

   The **Reporting Datamart** structure creation starts.

   The **Reporting Datamart** is initialized according to the HOPEX repository metamodel: all the tables (columns and rows) are created.

   If you have already initialized the **Reporting Datamart** data, only the empty tables and columns are added.

   Else initialize the **Reporting Datamart** data to feed the tables, rows and columns (see Initializing the Reporting Datamart data section).

3.3. Initializing the Reporting Datamart data

Once the **Reporting Datamart** is created you need to initialize the **Reporting Datamart** data with the HOPEX repository content according to the user/profile permissions and the object type selected.

To initialize the Reporting Datamart data from the HOPEX repository:

1) Launch **HOPEX Administration** application.
2) Connect to the environment concerned.
3) In the **Repositories** folder, expand the repository folder concerned.
4) In the **Reporting Datamart** folder, right-click the **Reporting Datamart** concerned and select **Initialize > Data**.
5) Click **Start**.

   The **Reporting Datamart** data initialization starts.

   The **Reporting Datamart** Data is initialized according to the HOPEX repository content.

4. SYNCHRONIZING THE REPORTING DATAMART WITH THE HOPEX REPOSITORY CONTENT

To keep your **Reporting Datamart** up-to-date with HOPEX repository updates, you can synchronize it with HOPEX Repository content thanks to the following provided triggers:

- Incremental synchronization
- Calculated MetaAttribute synchronization
- Diagram synchronization

Although, if needed, you can modify the **Reporting Datamart** data, these modifications are never synchronized with the HOPEX repository. Synchronization is only one way: from HOPEX repository to **Reporting Datamart**.

4.1. Synchronization frequency

The schedule policy of these synchronization triggers depends on the object volume in the HOPEX Repository and on the up-to-date data level you want to deliver to the Reporting Datamart users.

If you need a perfect synchronization between data, calculated MetaAttributes and Diagrams you can daily run the corresponding triggers at the same time.

Synchronization scheduling use: once scheduled the trigger launch the synchronization on all the repositories and for all their Reporting Datamarts.

To trigger the action on a specific repository for a specific Reporting Datamart, see HOPEX Power Studio > All about starting with APIs > Initializing and synchronizing a Reporting Datamart.

4.2. Launching an incremental synchronization

The HOPEX repository log must be activated (default value for an RDMS repository).

Launch an incremental synchronization to update the **Reporting Datamart** with all the dispatches performed after the last synchronization.

You can:

- at any time, manually launch an incremental synchronization
- define a scheduled trigger for the incremental synchronization

  Trigger: Reporting Datamart Synchronization

  Frequency: you can schedule the incremental synchronization with a high frequency (i.e.: every 10').

  For information regarding the scheduling, see *HOPEX Power Studio > Using the Scheduler > Scheduler*.

To manually launch an incremental synchronization:

1) Launch **HOPEX Administration** application.
2) Connect to the environment concerned.

3) In the **Repositories** folder, expand the repository folder concerned.

4) In the **Reporting Datamart** folder, right-click the Reporting Datamart concerned and select **Synchronize > Incremental update**.

To define the scheduled trigger for an incremental synchronization:

1) Launch **HOPEX Administration** application.
2) Connect to the environment concerned.
3) In the **Repositories** folder, expand the repository folder concerned.
4) Right-click Scheduler and select Manage triggers.
5) (If needed) In the **System Triggers** tab, right-click **Reporting Datamart Synchronization** and select **Update Scheduling** to modify the scheduling according to your needs.
6) In the System Triggers tab, right-click Reporting Datamart Synchronization and select Activate.

   The **Reporting Datamart Synchronization** is scheduled as defined (by default every 10').

4.3. Launching a calculated MetaAttribute synchronization

Launch a calculated MetaAttribute synchronization to scan all the objects and links of the HOPEX repository that can have calculated MetaAttribute values and put their values in the **Reporting Datamart**.

Do not launch or schedule a calculated MetaAttribute synchronization if you do not use the values of calculated MetaAttributes from the Reporting Datamart.

To exclude specific MetaAtribute values from this synchronization, see Excluding MetaAttribute values from the Reporting Datamart.

You can:

- manually launch a calculated MetaAttribute synchronization
- define a scheduled trigger for the diagram synchronization

  Trigger: Reporting Datamart Synchronization (Calculated MetaAttribute)

  Frequency: The synchronization may take time according to the HOPEX Repository calculated MetaAttribute volume. You should first manually launch a calculated MetaAttribute synchronization once to check the synchronization duration, so as to determine the synchronization scheduling frequency.

  When you schedule a "Reporting Datamart Synchronization (Calculated MetaAttribute)" the trigger is launched on all the repositories and for all their Reporting Datamarts.

  To trigger the action on a specific repository for a specific Reporting Datamart, see HOPEX Power Studio > All about starting with APIs > Initializing and synchronizing a Reporting Datamart.

  For information regarding the scheduling, see *HOPEX Power Studio > Using the Scheduler > Scheduler*.

To manually launch a calculated MetaAttribute synchronization:

1) Launch **HOPEX Administration** application.

2) Connect to the environment concerned.
3) In the **Repositories** folder, expand the repository folder concerned.
4) In the **Reporting Datamart** folder, right-click the Reporting Datamart concerned and select **Synchronize > Attributes calculated**.

To define the scheduled trigger for a calculated MetaAttribute synchronization:

1) Launch **HOPEX Administration** application.
2) Connect to the environment concerned.
3) In the **Repositories** folder, right-click **Scheduler** and select **Manage triggers.**
4) In the System Triggers tab, right-click Reporting Datamart Synchronization (Calculated MetaAttribute) and select Execute

   Check the synchronization duration.
5) In the System Triggers tab, right-click Reporting Datamart Synchronization (Calculated MetaAttribute) and select Update Scheduling.
6) Modify the scheduling according to your needs.
7) Click **OK**.
8) In the System Triggers tab, right-click Reporting Datamart Synchronization (Calculated MetaAttribute) and select Activate.

   The Reporting Datamart Synchronization (Calculated MetaAttribute) is scheduled as defined.

4.4. Launching a diagram synchronization

Launch a diagram synchronization to scan all the HOPEX repository diagrams and update the **Reporting Datamart** with their drawing representation in the SVG format.

Do not launch or schedule a diagram synchronization if you do not use diagrams.

You can:

- manually launch a diagram synchronization
- define a scheduled trigger for the diagram synchronization

  Trigger: Reporting Datamart Synchronization (Diagrams)

  Frequency: The synchronization may take time according to the HOPEX Repository diagram volume. You should first manually launch a diagram synchronization once to check the synchronization duration, so as to determine the synchronization scheduling frequency.

When you schedule a “Reporting Datamart Synchronization (Diagrams)” the trigger is launched on all the repositories and for all their Reporting Datamarts.

To trigger the action on a specific repository for a specific Reporting Datamart, see HOPEX Power Studio > All about starting with APIs > Initializing and synchronizing a Reporting Datamart.

For information regarding the scheduling, see *HOPEX Power Studio > Using the Scheduler > Scheduler*.

To manually launch a diagram synchronization:

1) Launch **HOPEX Administration** application.

2) Connect to the environment concerned.
3) In the **Repositories** folder, expand the repository folder concerned.
4) In the **Reporting Datamart** folder, right-click the Reporting Datamart concerned and select **Synchronize > Diagrams**.

To define the scheduled trigger for a diagram synchronization:

1) Launch **HOPEX Administration** application.
2) Connect to the environment concerned.
3) In the **Repositories** folder, right-click **Scheduler** and select **Manage triggers.**
4) In the System Triggers tab, right-click Reporting Datamart Synchronization (Diagrams) and select Execute.

   Check the synchronization duration.
5) In the System Triggers tab, right-click Reporting Datamart Synchronization (Diagrams) and select Update Scheduling.
6) Modify the scheduling according to your needs.
7) Click **OK**.
8) In the System Triggers tab, right-click Reporting Datamart Synchronization (Diagrams) and select Activate.

   The Reporting Datamart Synchronization (Diagrams) is scheduled as defined.

5. DELETING A REPORTING DATAMART

You can delete a **Reporting Datamart**.

To delete a Reporting Datamart:

1) Launch **HOPEX Administration** application.
2) Connect to the environment concerned.
3) In the **Repositories** folder, expand the repository folder concerned.
4) In the **Reporting Datamart** folder, right-click the reporting Datamart concerned and select **Delete**.

6. USING THE REPORTING DATAMART

6.1. Usage

The **Reporting Datamart** can be used as:

- a data source to generate reports, through queries on a person data according to a specific profile.
- a repository to build custom static web sites with external CMS tools
- a repository to build a custom application, or an outbound interface to another tool
- a repository for mobile applications.

6.2. Advantages

Advantages of using the **Reporting Datamart** are that:

- data amount is reduced to comply with the creation parameters (user and profile rights) (see Content section)
- data access is simplified
- it contains up-to-date information (see Synchronizing the Reporting Datamart with the HOPEX Repository content section)
- it may run on a separate server
- it does not affect HOPEX repository performance while using it

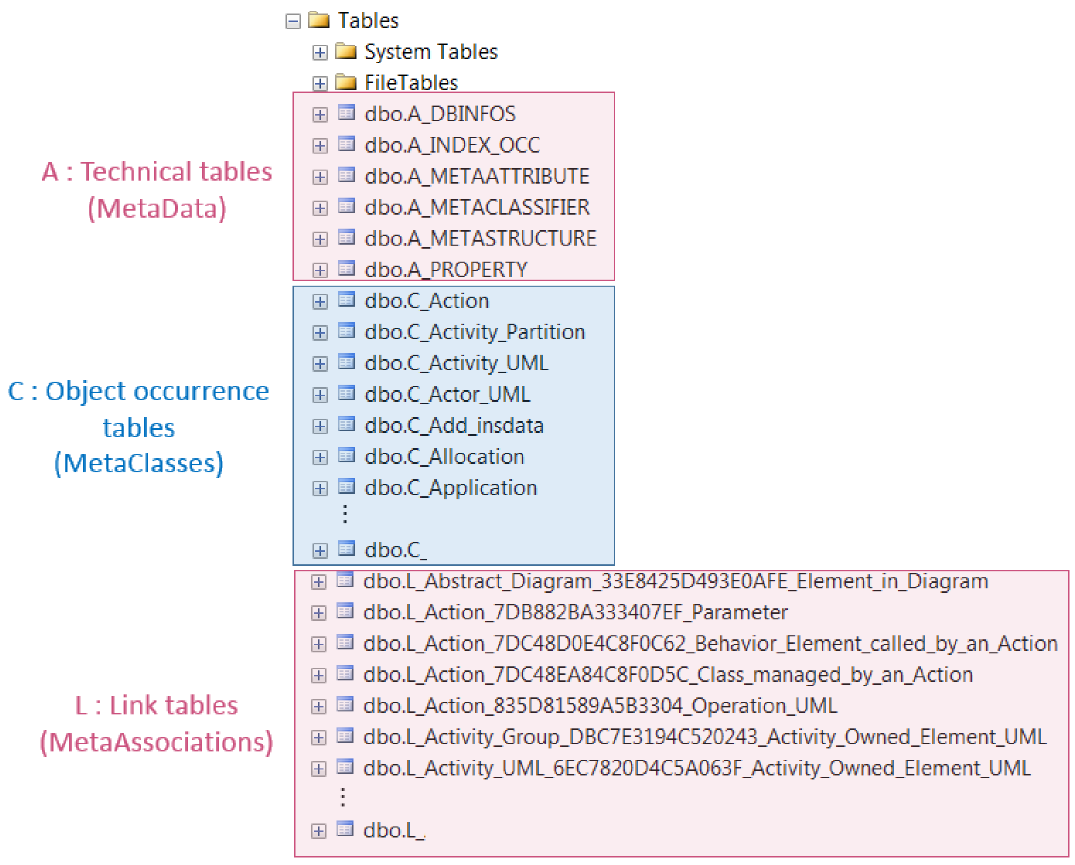
  However, see Launching a diagram synchronization and Launching a calculated MetaAttribute synchronization sections for HOPEX repository performance issues.

7. REPORTING DATAMART DETAILED DESCRIPTION

7.1. Reporting Datamart table classification

In the **Tables** folder, tables are classified according to their content:

- Technical data (MetaData)
- Object occurrences (MetaClass occurrences)
- Links (MetaAssociations)


A : Technical tables (MetaData)
C : Object occurrence tables (MetaClasses)
L : Link tables (MetaAssociations)

```
Tables
  System Tables
  FileTables
  dbo.A_DBINFOS
  dbo.A_INDEX_OCC
  dbo.A_METAATTRIBUTE
  dbo.A_METACLASSIFIER
  dbo.A_METASTRUCTURE
  dbo.A_PROPERTY
  dbo.C_Action
  dbo.C_Activity_Partition
  dbo.C_Activity_UML
  dbo.C_Actor_UML
  dbo.C_Add_insdata
  dbo.C_Allocation
  dbo.C_Application
  ⋮
  dbo.C_
  dbo.L_Abstract_Diagram_33E8425D493E0AFE_Element_in_Diagram
  dbo.L_Action_7DB882BA333407EF_Parameter
  dbo.L_Action_7DC48D0E4C8F0C62_Behavior_Element_called_by_an_Action
  dbo.L_Action_7DC48EA84C8F0D5C_Class_managed_by_an_Action
  dbo.L_Action_835D81589A5B3304_Operation_UML
  dbo.L_Activity_Group_DBC7E3194C520243_Activity_Owned_Element_UML
  dbo.L_Activity_UML_6EC7820D4C5A063F_Activity_Owned_Element_UML
  ⋮
  dbo.L_
```

7.2. Reporting Datamart table name format

Each table name format is as follows:

dbo.<letter>_<table name>

dbo: data base owner

<letter>: **A** for Technical tables

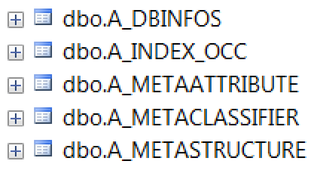
C for object occurrence tables

L for link tables

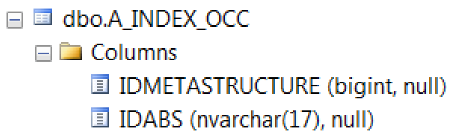
<table name>: describes the table content

7.2.1. Technical tables

Technical table name is prefixed with: **dbo.A_**.

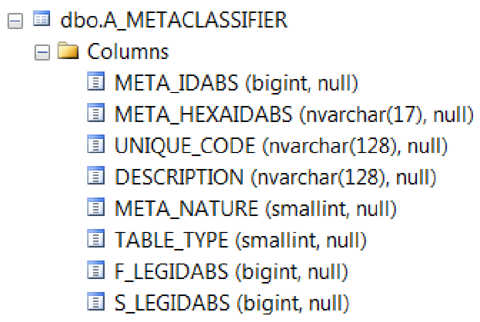

dbo.A_DBINFOS
dbo.A_INDEX_OCC
dbo.A_METAATTRIBUTE
dbo.A_METACLASSIFIER
dbo.A_METASTRUCTURE

- dbo.A_**DBINFOS** table is for internal use only (source repository information).
- dbo.A_**INDEX_OCC** table details all the object IdAbs with their corresponding MetaStructure Idabs. It enables to find out to which table belongs an object.


dbo.A_INDEX_OCC
Columns
IDMETASTRUCTURE (bigint, null)
IDABS (nvarchar(17), null)

For an example see Finding information on an attribute section.

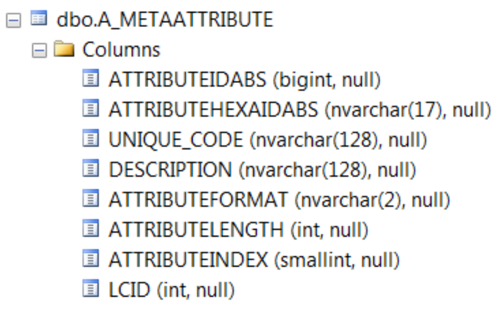
- dbo.A_**METACLASSIFIER** table describes the tables


dbo.A_METACLASSIFIER
Columns
META_IDABS (bigint, null)
META_HEXAIDABS (nvarchar(17), null)
UNIQUE_CODE (nvarchar(128), null)
DESCRIPTION (nvarchar(128), null)
META_NATURE (smallint, null)
TABLE_TYPE (smallint, null)
F_LEGIDABS (bigint, null)
S_LEGIDABS (bigint, null)

- META_IDABS and META_HEXAIDABS are the Metamodel object (MetaClass or MetaAssociation) IdAbs in HOPEX, in numerical (integer) and hexadecimal formats.
- UNIQUE_CODE is the table name
- DESCRIPTION is the Metamodel object (MetaClass or MetaAssociation) name in HOPEX
- F_LEGIDABS and S_LEGIDABS give the MetaAssociationEnd Idabs

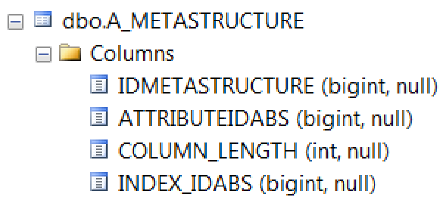
For examples see Finding the MetaClass, and Finding information on an attribute sections.

- dbo.A_**METATTRIBUTE** describes the columns


dbo.A_METAATTRIBUTE
Columns
ATTRIBUTEIDABS (bigint, null)
ATTRIBUTEHEXAIDABS (nvarchar(17), null)
UNIQUE_CODE (nvarchar(128), null)
DESCRIPTION (nvarchar(128), null)
ATTRIBUTEFORMAT (nvarchar(2), null)
ATTRIBUTELENGTH (int, null)
ATTRIBUTEINDEX (smallint, null)
LCID (int, null)

- ATTRIBUTEIDABS and ATTRIBUTEHEXAIDABS are the MetaAttribute IdAbs in HOPEX
- UNIQUE_CODE is the column name
- DESCRIPTION is the MetaAttribute name in HOPEX

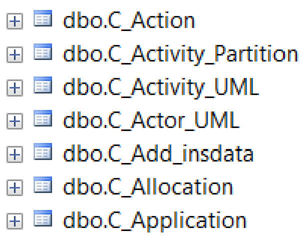
- dbo.A_**METASTRUCTURE** describes the Table x Column matrix


dbo.A_METASTRUCTURE
Columns
IDMETASTRUCTURE (bigint, null)
ATTRIBUTEIDABS (bigint, null)
COLUMN_LENGTH (int, null)
INDEX_IDABS (bigint, null)

- IDMETASTRUCTURE gives the META_IdAbs of the dbo.A_METACLASSIFIER table
- ATTRIBUTEIDABS gives the MetaAttribute IdAbs in HOPEX.

7.2.2. MetaClass occurrence tables

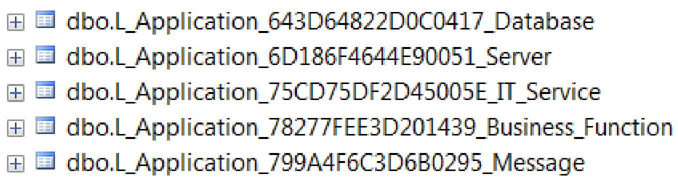
MetaClass occurrence table name is prefixed with: **dbo.C_**.


dbo.C_Action
dbo.C_Activity_Partition
dbo.C_Activity_UML
dbo.C_Actor_UML
dbo.C_Add_insdata
dbo.C_Allocation
dbo.C_Application

7.2.3. MetaAssociation tables

MetaAssociaton table name:

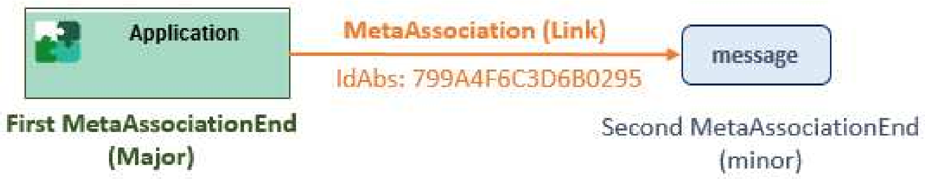
- is prefixed with: **dbo.L_**
- is made up of the MetaAssociation IdAbs between the major MetaAssociationEnd name (the first one) and the minor MetaAssociationEnd name (the second one)


dbo.L_Application_643D64822D0C0417_Database
dbo.L_Application_6D186F4644E90051_Server
dbo.L_Application_75CD75DF2D45005E_IT_Service
dbo.L_Application_78277FEE3D201439_Business_Function
dbo.L_Application_799A4F6C3D6B0295_Message

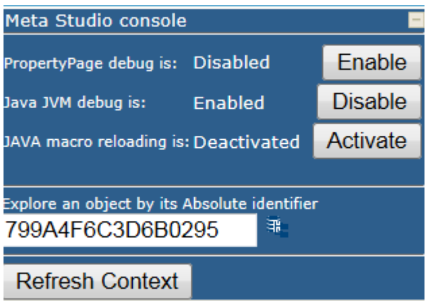
Example:

dbo.L_**Application**_799A4F6C3D6B0295_**Message**:

- dbo.L: indicates it is a link (MetaAssociation) table
- **Application**: the first MetaAssociationEnd (major)
- 799A4F6C3D6B0295: the "Application-Message" MetaAssociation IdAbs
- **Message**: the second MetaAssociationEnd (minor)


Application
MetaAssociation (Link)
IdAbs: 799A4F6C3D6B0295
message
First MetaAssociationEnd (Major)
Second MetaAssociationEnd (minor)

In HOPEX, with the MetaStudio console, in **Explore an object by its absolute identifier**, enter the 799A4F6C3D6B0295 IdAbs.


Meta Studio console

PropertyPage debug is: Disabled

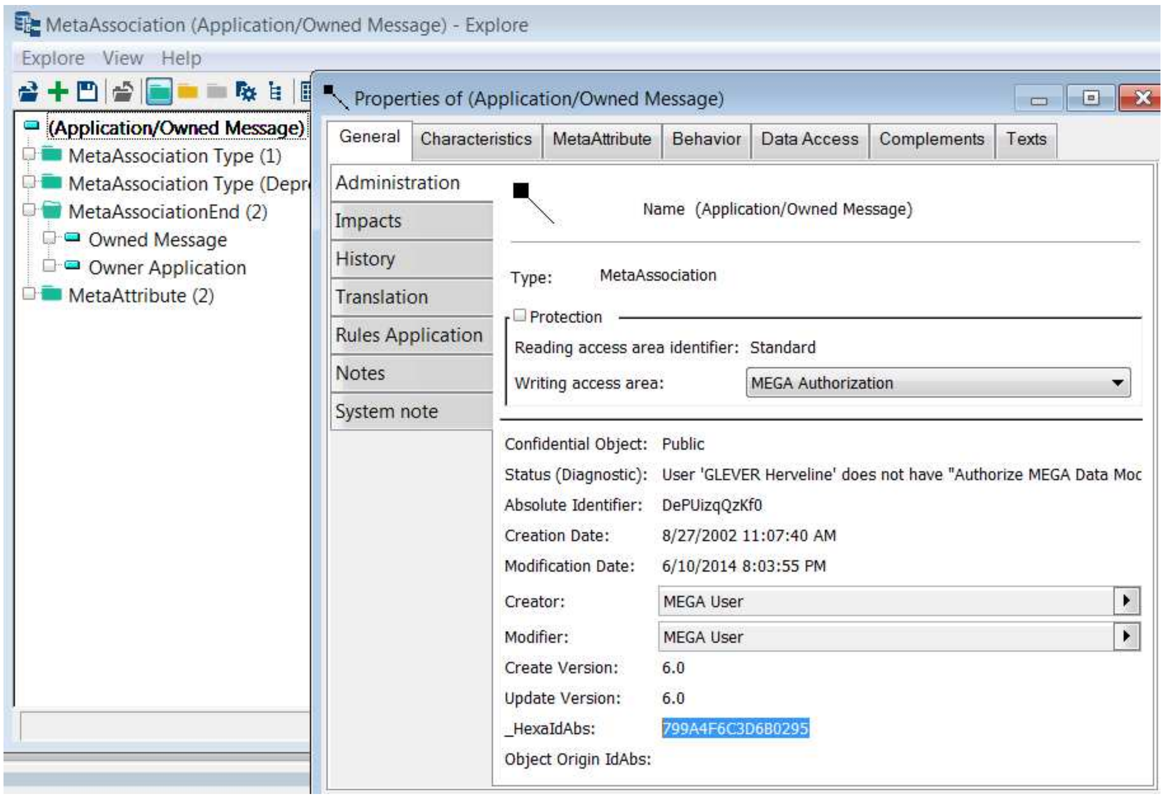
Java JVM debug is: Enabled

JAVA macro reloading is: Deactivated

Explore an object by its Absolute identifier

799A4F6C3D6B0295

You get the MetaAssociation:


MetaAssociation (Application/Owned Message) - Explore

(Application/Owned Message)
- MetaAssociation Type (1)
- MetaAssociation Type (Depr
- MetaAssociationEnd (2)
  - Owned Message
  - Owner Application
- MetaAttribute (2)

Properties of (Application/Owned Message)

Name (Application/Owned Message)

Type: MetaAssociation

Protection

Reading access area identifier: Standard

Writing access area: MEGA Authorization

Confidential Object: Public

Status (Diagnostic): User 'GLEVER Herveline' does not have "Authorize MEGA Data Moc

Absolute Identifier: DePUizqQzKf0

Creation Date: 8/27/2002 11:07:40 AM

Modification Date: 6/10/2014 8:03:55 PM

Creator: MEGA User

Modifier: MEGA User

Create Version: 6.0

Update Version: 6.0

_HexaIdAbs: 799A4F6C3D6B0295

Object Origin IdAbs:

7.3. Reporting Datamart columns

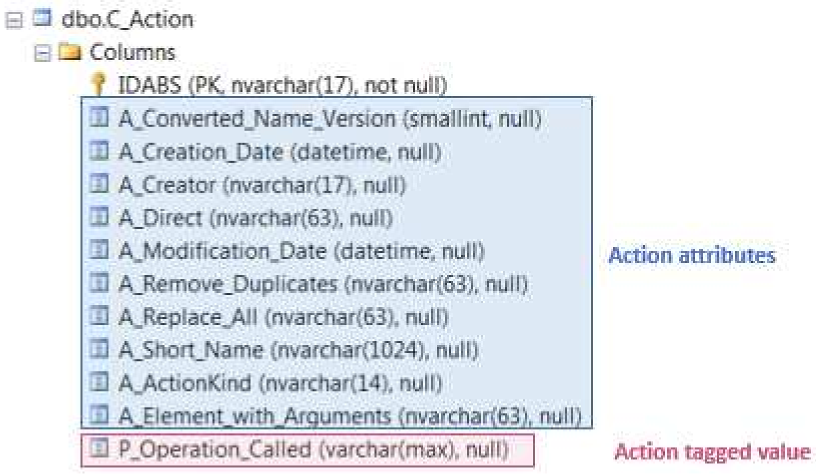
7.3.1. MetaClass occurrence table columns

In **dbo.C_**<MetaClass name>, the **Columns** folder details the attributes or tagged values belonging to the MetaClass occurrence concerned. Each column name is prefixed as follows:

- **A_** for attributes
- **P_** for tagged values

Example:

dbo.C_Action table includes in **Columns** folder all the attributes and tagged values belonging to the Actions.



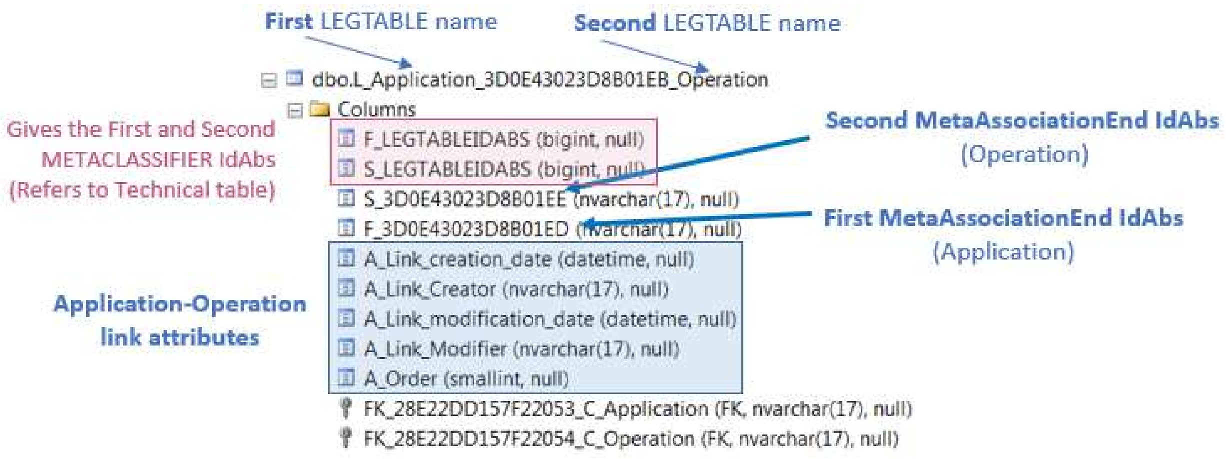
```
dbo.C_Action
  Columns
    IDABS (PK, nvarchar(17), not null)
    A_Converted_Name_Version (smallint, null)
    A_Creation_Date (datetime, null)
    A_Creator (nvarchar(17), null)
    A_Direct (nvarchar(63), null)
    A_Modification_Date (datetime, null)
    A_Remove_Duplicates (nvarchar(63), null)
    A_Replace_All (nvarchar(63), null)
    A_Short_Name (nvarchar(1024), null)
    A_ActionKind (nvarchar(14), null)
    A_Element_with_Arguments (nvarchar(63), null)
    P_Operation_Called (varchar(max), null)
```

7.3.2. Link table columns

In **dbo.L_<...>**, the **Columns** folder details the attributes, and the first (major) and second (minor) MetaAssociationEnd IdAbs belonging to the MetaAssociation concerned. Each column name is prefixed as follows:

- F_LEGTABLEIDABS: indicates the first (major) table IdAbs
- S_LEGTABLEIDABS: indicates the second (minor) table IdAbs
- F_< IdAbs>: indicates the first (major) MetaAssociationEnd IdAbs
- S_< IdAbs>: indicates the second (minor) MetaAssociationEnd IdAbs
- A_<...> for attributes

Example:


First LEGTABLE name
Second LEGTABLE name
Gives the First and Second METACLASSIFIER IdAbs (Refers to Technical table)
Second MetaAssociationEnd IdAbs (Operation)
First MetaAssociationEnd IdAbs (Application)
Application-Operation link attributes

dbo.L_Application_3D0E43023D8B01EB_Operation
Columns
F_LEGTABLEIDABS (bigint, null)
S_LEGTABLEIDABS (bigint, null)
S_3D0E43023D8B01EE (nvarchar(17), null)
F_3D0E43023D8B01ED (nvarchar(17), null)
A_Link_creation_date (datetime, null)
A_Link_Creator (nvarchar(17), null)
A_Link_modification_date (datetime, null)
A_Link_Modifier (nvarchar(17), null)
A_Order (smallint, null)
FK_28E22DD157F22053_C_Application (FK, nvarchar(17), null)
FK_28E22DD157F22054_C_Operation (FK, nvarchar(17), null)

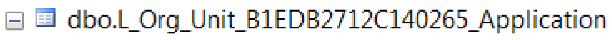
In dbo.L_**Application**_3D0E43023D8B01EB_**Operation** link table, the **Columns** folder is made up of:

- **Application** is the first MetaAssociationEnd (major)

  F_3D0E43023D8B01ED is the Application occurrence IdAbs
- **Operation** is the second MetaAssociationEnd (minor)

  S_3D0E43023D8B01EE is the Operation occurrence IdAbs

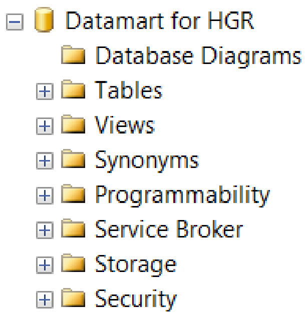
7.4. Use case: reading the Reporting Datamart through a link


dbo.L_Org_Unit_B1EDB2712C140265_Application

7.4.1. Accessing your Reporting Datamart tables

To access your Reporting Datamart:

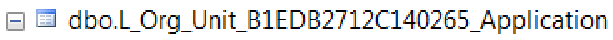
1) Launch Microsoft SQL Server Management Studio.
2) Access the server where you saved your reporting Datamart.
3) In the server tree view, expand **Databases** folder.
4) Expand your Reporting Datamart folder.


Datamart for HGR
Database Diagrams
Tables
Views
Synonyms
Programmability
Service Broker
Storage
Security

5) Expand the **Tables** folder.

7.4.2. Understanding a link table

Link example: Org_Unit - Application.


dbo.L_Org_Unit_B1EDB2712C140265_Application

Dbo.L: “Org_Unit - Application” link

Org_Unit: major MetaAssociationEnd

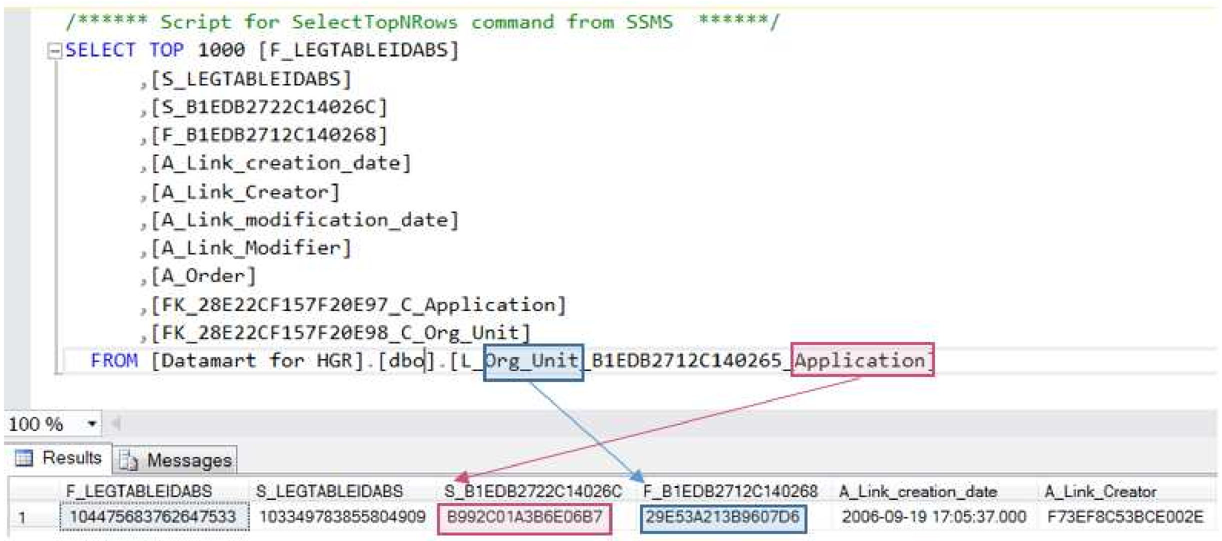
B1EDB2712C140265: MetaAssociation IdAbs

Application: minor MetaAssociationEnd

To understand the dbo.L_Org_Unit_B1EDB2712C140265_Application table information:

1) In your Reporting Datamart tree view, **Tables** folder, right-click **dbo_Org_Unit_B1EDB2712C140265_Application** table and select **Select Top 1000 rows**.

   The result is displayed in the right pane.



```
/****** Script for SelectTopNRows command from SSMS  ******/
SELECT TOP 1000 [F_LEGTABLEIDABS]
      ,[S_LEGTABLEIDABS]
      ,[S_B1EDB2722C14026C]
      ,[F_B1EDB2712C140268]
      ,[A_Link_creation_date]
      ,[A_Link_Creator]
      ,[A_Link_modification_date]
      ,[A_Link_Modifier]
      ,[A_Order]
      ,[FK_28E22CF157F20E97_C_Application]
      ,[FK_28E22CF157F20E98_C_Org_Unit]
  FROM [Datamart for HGR].[dbo].[L_Org_Unit_B1EDB2712C140265_Application]
```

	F_LEGTABLEIDABS	S_LEGTABLEIDABS	S_B1EDB2722C14026C	F_B1EDB2712C140268	A_Link_creation_date	A_Link_Creator
1	104475683762647533	103349783855804909	B992C01A3B6E06B7	29E53A213B9607D6	2006-09-19 17:05:37.000	F73EF8C53BCE002E

- 104475683762647533 is the **Org Unit** MetaClass IdAbs (for the first MetaAssociationEnd).

  See Finding the MetaClass corresponding to an IdAbs section.
- 103349783855804909 is the **Application** MetaClass IdAbs (for the second MetaAssociationEnd).
- 29E53A213B9607D6 is the **Org Unit** (first MetaAssociationEnd occurrence) IdAbs.

  See Finding attributes or tagged values belonging to a MetaClass occurrence section.
- B992C01A3B6E06B7 is the **Application** (second MetaAssociationEnd occurrence) IdAbs.

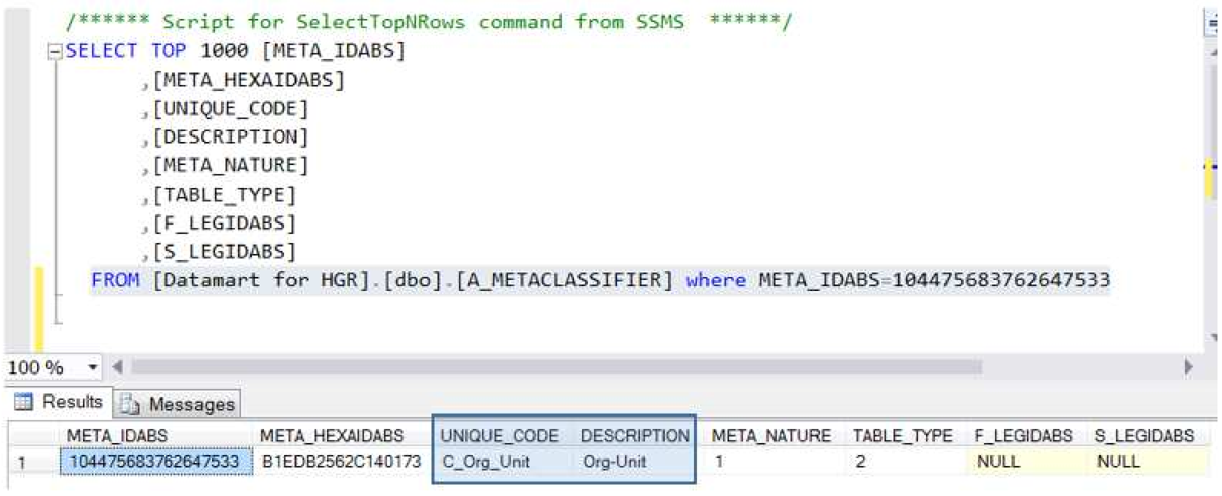
7.4.3. Finding the MetaClass corresponding to an IdAbs

To find out the MetaClass corresponding to the 104475683762647533 IdAbs use the **dbo.A_METACLASSIFIER** table.

To find out the MetaClass corresponding to 104475683762647533 IdAbs:

1) In your Reporting Datamart tree view, right-click **dbo.A_METACLASSIFIER** and select **Select Top 1000 rows**.
2) In the right pane complete the query with : where META_IDABS=104475683762647533

   ```
   FROM [Datamart for HGR].[dbo].[A_METACLASSIFIER] where
   META_IDABS=104475683762647533
   ```
3) **Execute** the query.



```
/****** Script for SelectTopNRows command from SSMS  ******/
SELECT TOP 1000 [META_IDABS]
      ,[META_HEXAIDABS]
      ,[UNIQUE_CODE]
      ,[DESCRIPTION]
      ,[META_NATURE]
      ,[TABLE_TYPE]
      ,[F_LEGIDABS]
      ,[S_LEGIDABS]
  FROM [Datamart for HGR].[dbo].[A_METACLASSIFIER] where META_IDABS=104475683762647533
```

	META_IDABS	META_HEXAIDABS	UNIQUE_CODE	DESCRIPTION	META_NATURE	TABLE_TYPE	F_LEGIDABS	S_LEGIDABS
1	104475683762647533	B1EDB2562C140173	C_Org_Unit	Org-Unit	1	2	NULL	NULL

The query result indicates that 104475683762647533 is the **Org-Unit** MetaClass IdAbs.

7.4.4. Finding attributes or tagged values belonging to a MetaClass occurrence

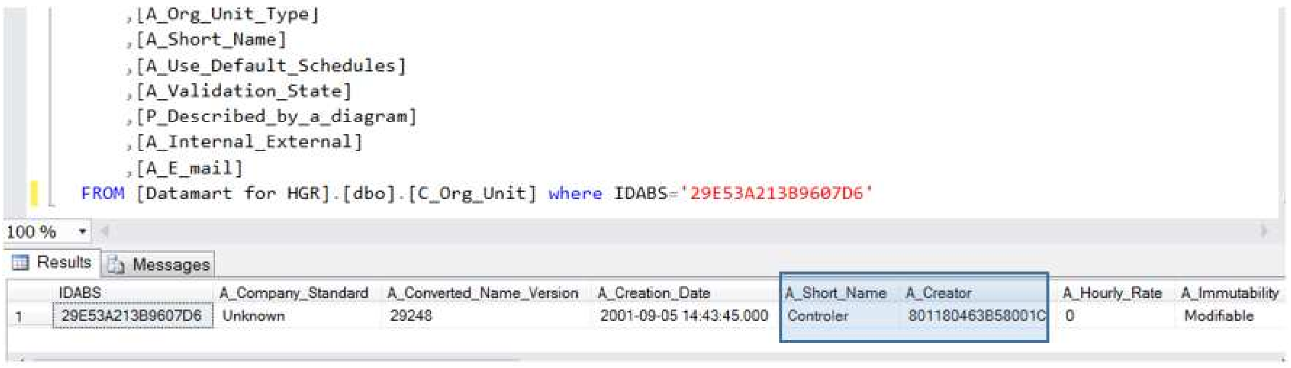
Use the **dbo.C_<MetaClass name>** table to find the attributes and tagged values belonging to a MetaClass occurrence.

For example 29E53A213B9607D6 is the IdAbs of an Org_Unit occurrence.

To find the Org_Unit table which IdAbs is 29E53A213B9607D6:

1) In your Reporting Datamart tree view, right-click **dbo.C_Org_Unit** and select **Select Top 1000 rows**.
2) In the right pane complete the query with: `where IDABS='29E53A213B9607D6'`

   ```
   FROM [Datamart for HGR].[dbo].[C_Org_Unit] where IDABS='29E53A213B9607D6'
   ```
3) **Execute** the query.



```
      ,[A_Org_Unit_Type]
      ,[A_Short_Name]
      ,[A_Use_Default_Schedules]
      ,[A_Validation_State]
      ,[P_Described_by_a_diagram]
      ,[A_Internal_External]
      ,[A_E_mail]
  FROM [Datamart for HGR].[dbo].[C_Org_Unit] where IDABS='29E53A213B9607D6'
```

	IDABS	A_Company_Standard	A_Converted_Name_Version	A_Creation_Date	A_Short_Name	A_Creator	A_Hourly_Rate	A_Immutability
1	29E53A213B9607D6	Unknown	29248	2001-09-05 14:43:45.000	Controler	801180463B58001C	0	Modifiable

The query result details all the Org-Unit attributes (IdAbs: 29E53A213B9607D6, Short Name: Controler). It indicates, in particular that its creator IdAbs is: 801180463B58001C.

7.4.5. Finding information on an attribute

You want information on the Creator attribute whose idabs is 801180463B58001C.

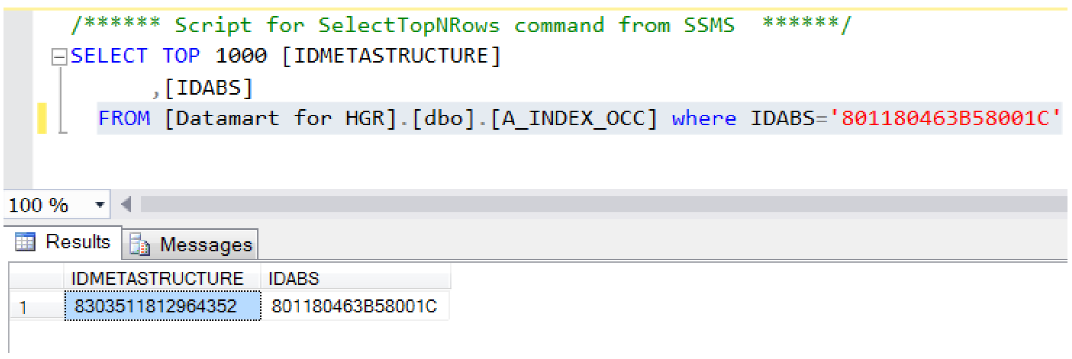
You need to:

- use the **dbo.A_INDEX_OCC** table to get the MetaClass IdAbs to which the Creator belongs to.
- use the **dbo.A_METACLASSIFIER** table to get the MetaClass name from its Idabs
- use the **dbo.C_<MetaClass name>** table to get the Creator (801180463B58001C IdAbs) information

To find the MetaClass IdAbs of the MetaClass occurrence whose IdAbs is 801180463B58001C:

1) To find out to which MetaClass belongs the occurrence with 801180463B58001C IdAbs, in your Reporting Datamart tree view, right-click **dbo.A_INDEX_OCC** and select **Select Top 1000 rows**.
2) In the right pane complete the query with: `where IDABS='801180463B58001C'`

   ```
   FROM [Datamart for HGR].[dbo].[A_INDEX_OCC] where IDABS='801180463B58001C'
   ```
3) **Execute** the query.



```
/****** Script for SelectTopNRows command from SSMS  ******/
SELECT TOP 1000 [IDMETASTRUCTURE]
      ,[IDABS]
  FROM [Datamart for HGR].[dbo].[A_INDEX_OCC] where IDABS='801180463B58001C'
```

100 %

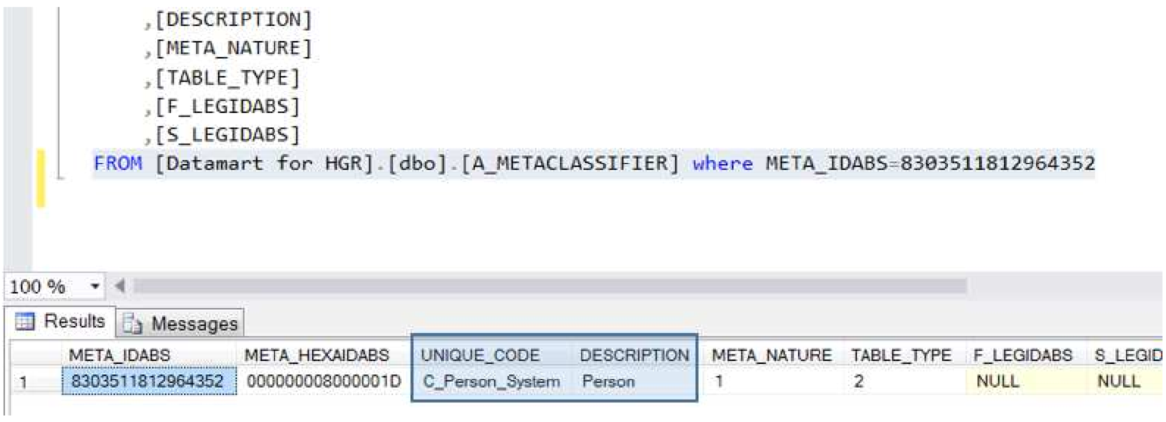
Results | Messages

	IDMETASTRUCTURE	IDABS
1	8303511812964352	801180463B58001C

The query result indicates that the IDMETASTRUCTURE of the 801180463B58001C IdAbs is: 8303511812964352.

4) To find out which MetaClass has the 8303511812964352 IdAbs, in your Reporting Datamart tree view, right-click **dbo.A_METACLASSIFIER** and select **Select Top 1000 rows**.
5) In the right pane complete the query with: `where META_IDABS=8303511812964352`

   ```
   FROM [Datamart for HGR].[dbo].[A_METACLASSIFIER] where
   META_IDABS=8303511812964352
   ```
6) **Execute** the query.



```
      ,[DESCRIPTION]
      ,[META_NATURE]
      ,[TABLE_TYPE]
      ,[F_LEGIDABS]
      ,[S_LEGIDABS]
  FROM [Datamart for HGR].[dbo].[A_METACLASSIFIER] where META_IDABS=8303511812964352
```

100 %

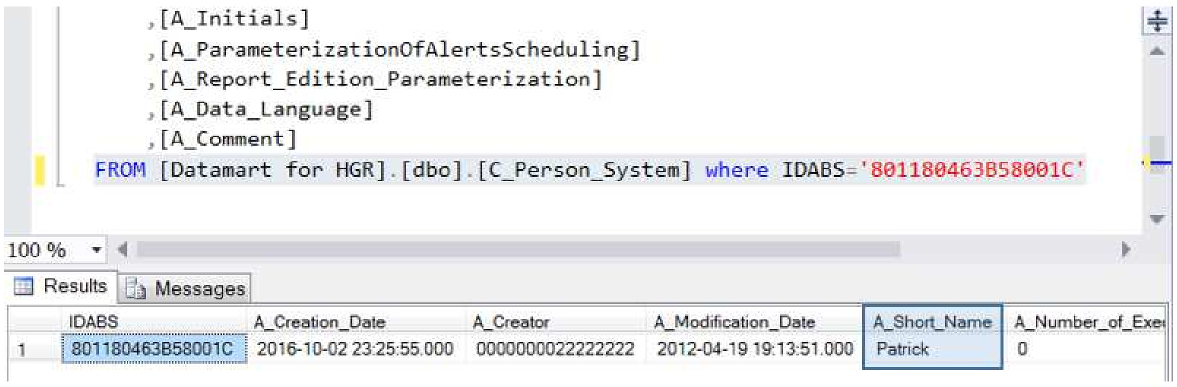
Results | Messages

	META_IDABS	META_HEXAIDABS	UNIQUE_CODE	DESCRIPTION	META_NATURE	TABLE_TYPE	F_LEGIDABS	S_LEGID
1	8303511812964352	000000008000001D	C_Person_System	Person	1	2	NULL	NULL

The query result indicates that 8303511812964352 IdAbs is the **Person System** MetaClass IdAbs.

7) To find out which Person has the 801180463B58001C IdAbs, in your Reporting Datamart tree view, right-click **dbo.C_Person_System** and select **Select Top 1000 rows**.
8) In the right pane complete the query with: `where IDABS='801180463B58001C'`

```
FROM [Datamart for HGR].[dbo].[C_Person_System] where
IDABS='801180463B58001C'
```



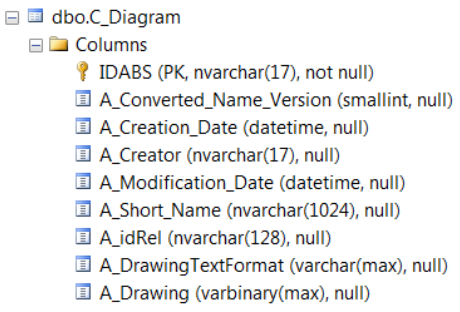
```
      ,[A_Initials]
      ,[A_ParameterizationOfAlertsScheduling]
      ,[A_Report_Edition_Parameterization]
      ,[A_Data_Language]
      ,[A_Comment]
  FROM [Datamart for HGR].[dbo].[C_Person_System] where IDABS='801180463B58001C'
```

```
   IDABS             A_Creation_Date          A_Creator          A_Modification_Date      A_Short_Name   A_Number_of_Exe
1  801180463B58001C  2016-10-02 23:25:55.000  0000000022222222   2012-04-19 19:13:51.000  Patrick        0
```

The query result indicates that the 801180463B58001C IdAbs belongs to Patrick.

7.5. Use case: saving the diagram drawings

In **dbo.C_Diagrams** table, the diagrams drawings are in the **A_Drawing** column.



dbo.C_Diagram
Columns
IDABS (PK, nvarchar(17), not null)
A_Converted_Name_Version (smallint, null)
A_Creation_Date (datetime, null)
A_Creator (nvarchar(17), null)
A_Modification_Date (datetime, null)
A_Short_Name (nvarchar(1024), null)
A_idRel (nvarchar(128), null)
A_DrawingTextFormat (varchar(max), null)
A_Drawing (varbinary(max), null)

To save the diagram drawings:

1) Launch Microsoft SQL Server Management Studio.
2) Access the server where you saved your reporting Datamart.
3) Check in your configuration that the following features are unlock (= 1):

```
/*
  CONFIGURATION
  =============
*/
 ---- check the configuration for 'Ole Automation Procedures'. It must be set
to 1
EXEC sp_configure 'Ole Automation Procedures';
GO
 ---- setting 'Ole Automation Procedures' to 1
EXEC sp_configure 'show advanced options', 1;
GO
RECONFIGURE;
GO
EXEC sp_configure 'Ole Automation Procedures', 1;
GO
RECONFIGURE;
 /*
  EXECUTION
  =========

*/
```

4) Create a folder where you want to retrieve the diagram drawings.

   `For example: C:\Pictures.`

5) Enter the following code:

```
DECLARE @SQLIMG VARCHAR(MAX),
                @IDABS NVARCHAR(17),
                @NAME NVARCHAR(1024),
                @IMG_PATH VARBINARY(MAX),
                @FILENAME VARCHAR(MAX),
                @ObjectToken INT

DECLARE IMGPATH CURSOR FAST_FORWARD FOR
    SELECT [IDABS], [A_Short_Name], [A_Drawing] from [dbo].[C_Diagram]
OPEN IMGPATH
```

```
FETCH NEXT FROM IMGPATH INTO @IDABS, @NAME, @IMG_PATH

WHILE @@FETCH_STATUS = 0
  BEGIN
    SET @FILENAME = 'C:\Pictures\'+ @IDABS + '_' + @NAME + '.png'

    EXEC sp_OACreate 'ADODB.Stream', @ObjectToken OUTPUT
    EXEC sp_OASetProperty @ObjectToken, 'Type', 1
    EXEC sp_OAMethod @ObjectToken, 'Open'
    EXEC sp_OAMethod @ObjectToken, 'Write', NULL, @IMG_PATH
    EXEC sp_OAMethod @ObjectToken, 'SaveToFile', NULL, @FILENAME, 2
    EXEC sp_OAMethod @ObjectToken, 'Close'
    EXEC sp_OADestroy @ObjectToken

    FETCH NEXT FROM IMGPATH INTO @IDABS, @NAME, @IMG_PATH
 END

CLOSE IMGPATH
DEALLOCATE IMGPATH
```

RDBMS Environment Duplication

INTRODUCTION

The goal of this document is to give detailed instructions as to how duplicating a HOPEX environment (hosted on an **SQL Server**).

We will take the hypothesis that the new databases of the duplicated environment are hosted on a separate SQL Server instance. This way, it gives the appropriate details in case you move data from one server to another.

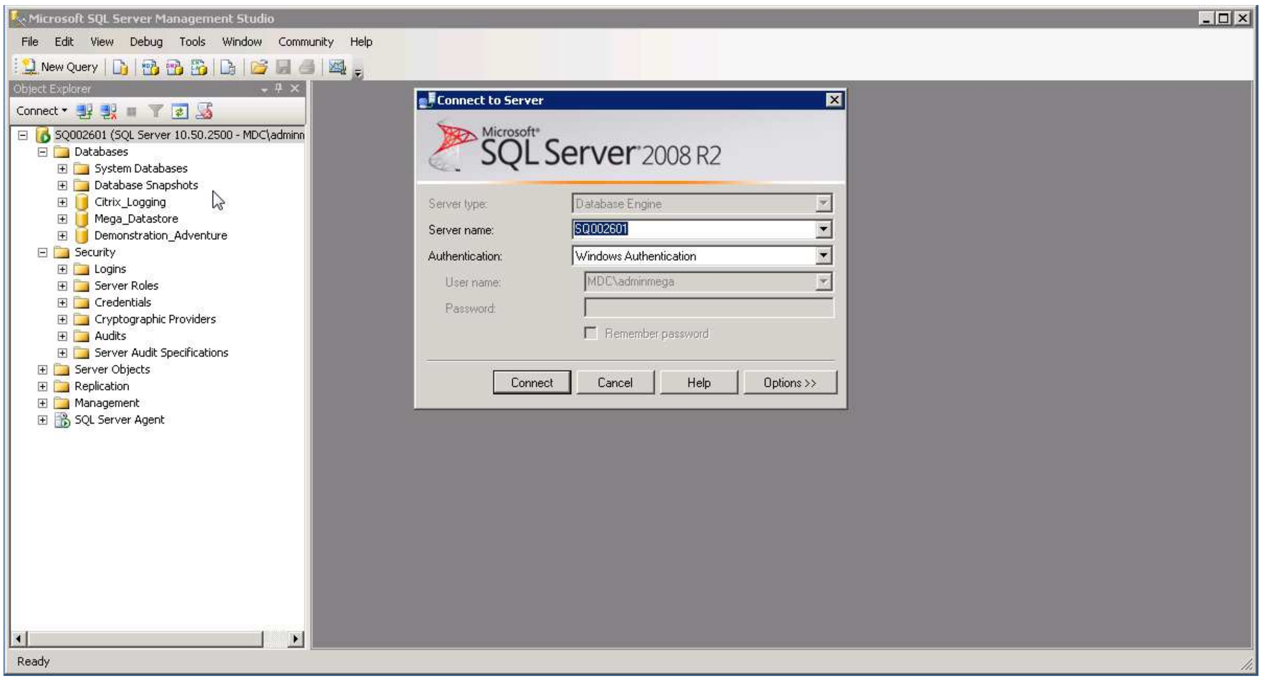
PREREQUISITES

The SQL Server account

Before you start, you have to make sure that, where the duplicated databases are going to be hosted, you have a user with the proper rights, either:

- you are given the sysadmin right (the super admin) on the instance, and you perform those actions yourself, or
- you have to ask the DBA to create it and grant it.

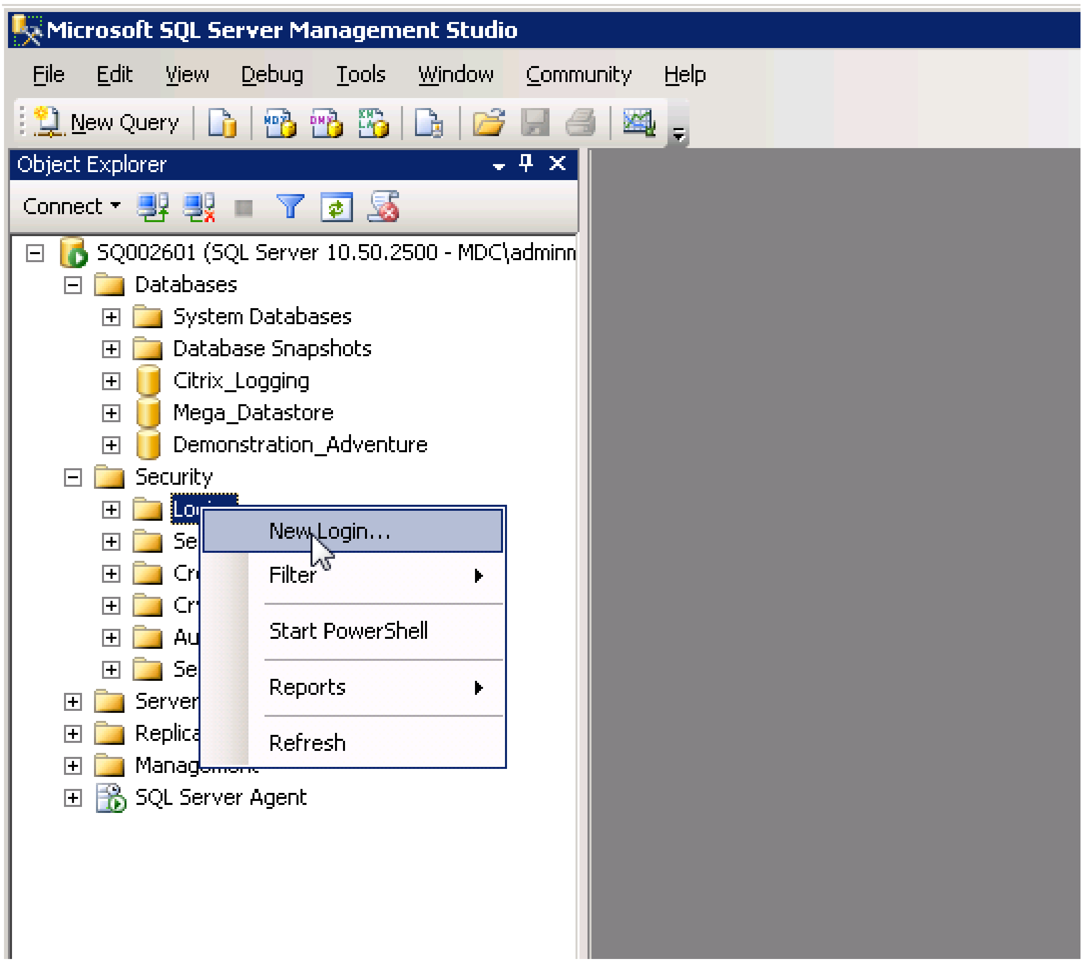
1. Launch the "Microsoft SQL Server Management Studio" tool.
2. Connect to the SQL Server instance with the user having the sysadmin right.
3. If your Windows account is the one having the sysadmin right, stay in « Windows Authentication » mode. Otherwise, switch to "SQL Server Authentication", and provide the username and password. The field "Server Name" is the name of the instance.



Connect to Server

Server type: Database Engine
Server name: SQ002601
Authentication: Windows Authentication
User name: MDC\adminmega
Password:
Remember password

4. In the left pane, expand **Security** file.
5. Right-click **Logins** and select **New Login**.

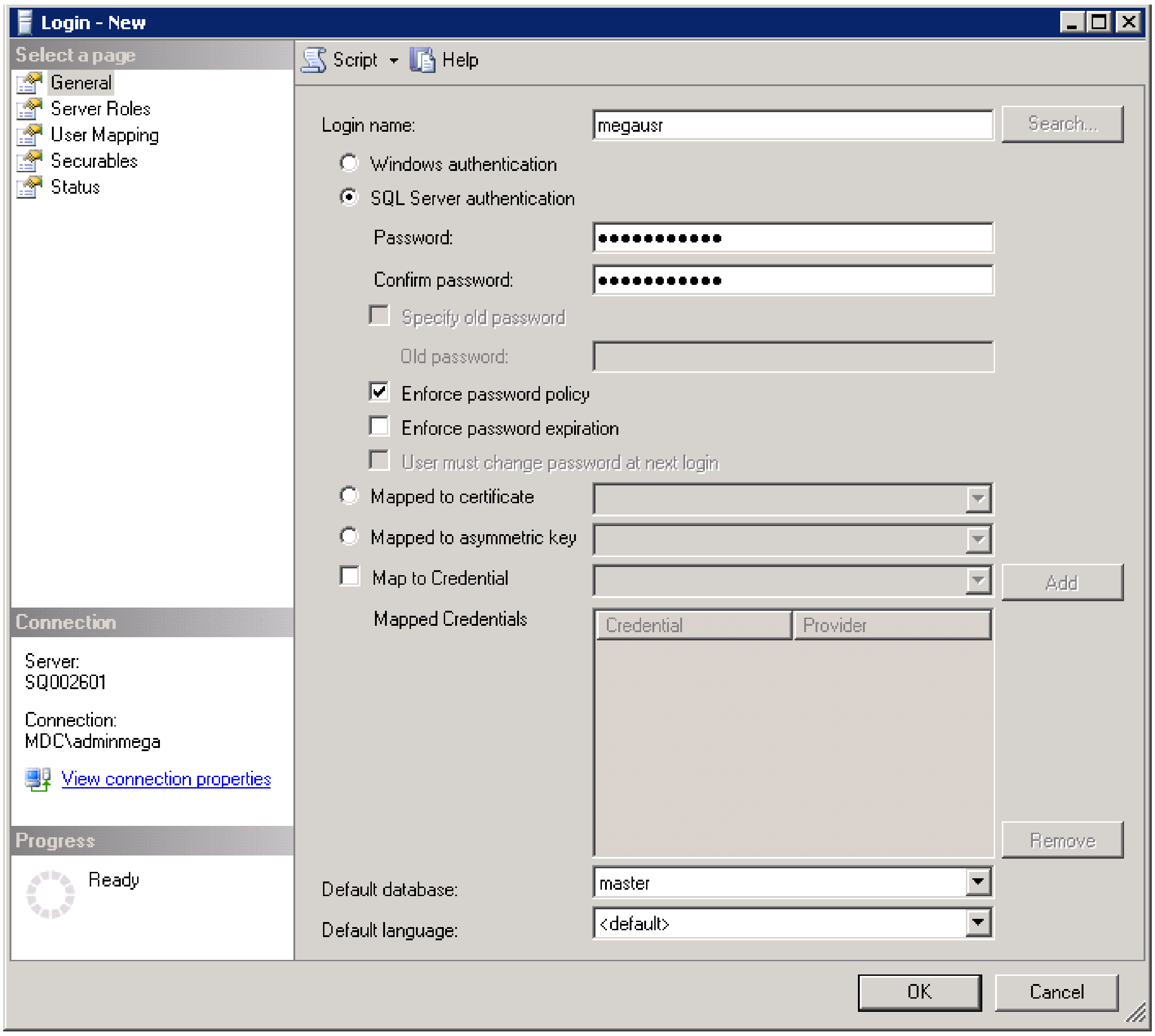


The **Login - New** window appears and displays and the **General** page.

6. In the right pane:
   a. In the field **Login name** enter the login, for example « megausr ».
   b. Select **SQL Server authentication** mode.
   c. In the field **Password**, enter a password that complies with security requirements, for example :

```
Mega2k8!usr
```

7. Unselect **Enforce password expiration** and User must change password at next login.



Login name: megausr

[ ] Windows authentication

[✓] SQL Server authentication

Password: ●●●●●●●●●●●●

Confirm password: ●●●●●●●●●●●●

[ ] Specify old password

Old password:

[✓] Enforce password policy

[ ] Enforce password expiration

[ ] User must change password at next login

[ ] Mapped to certificate

[ ] Mapped to asymmetric key

[ ] Map to Credential

Mapped Credentials

Credential	Provider

Default database: master

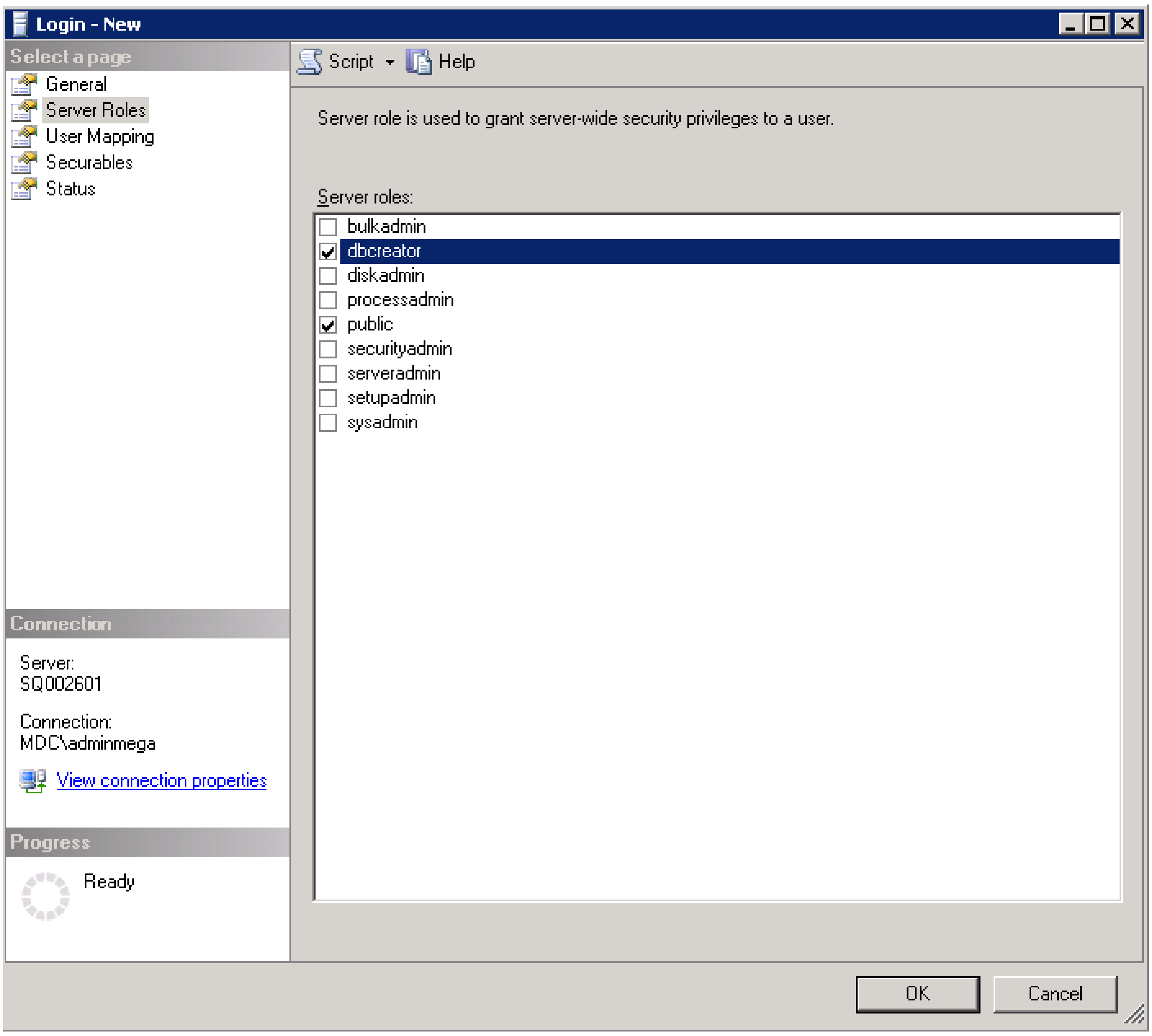
Default language: <default>

Server:
SQ002601

Connection:
MDC\adminmega

Ready

8. In the **Select a page** pane, select **Server Roles**.
9. In the right pane select **dbcreator**.



Server role is used to grant server-wide security privileges to a user.

Server roles:

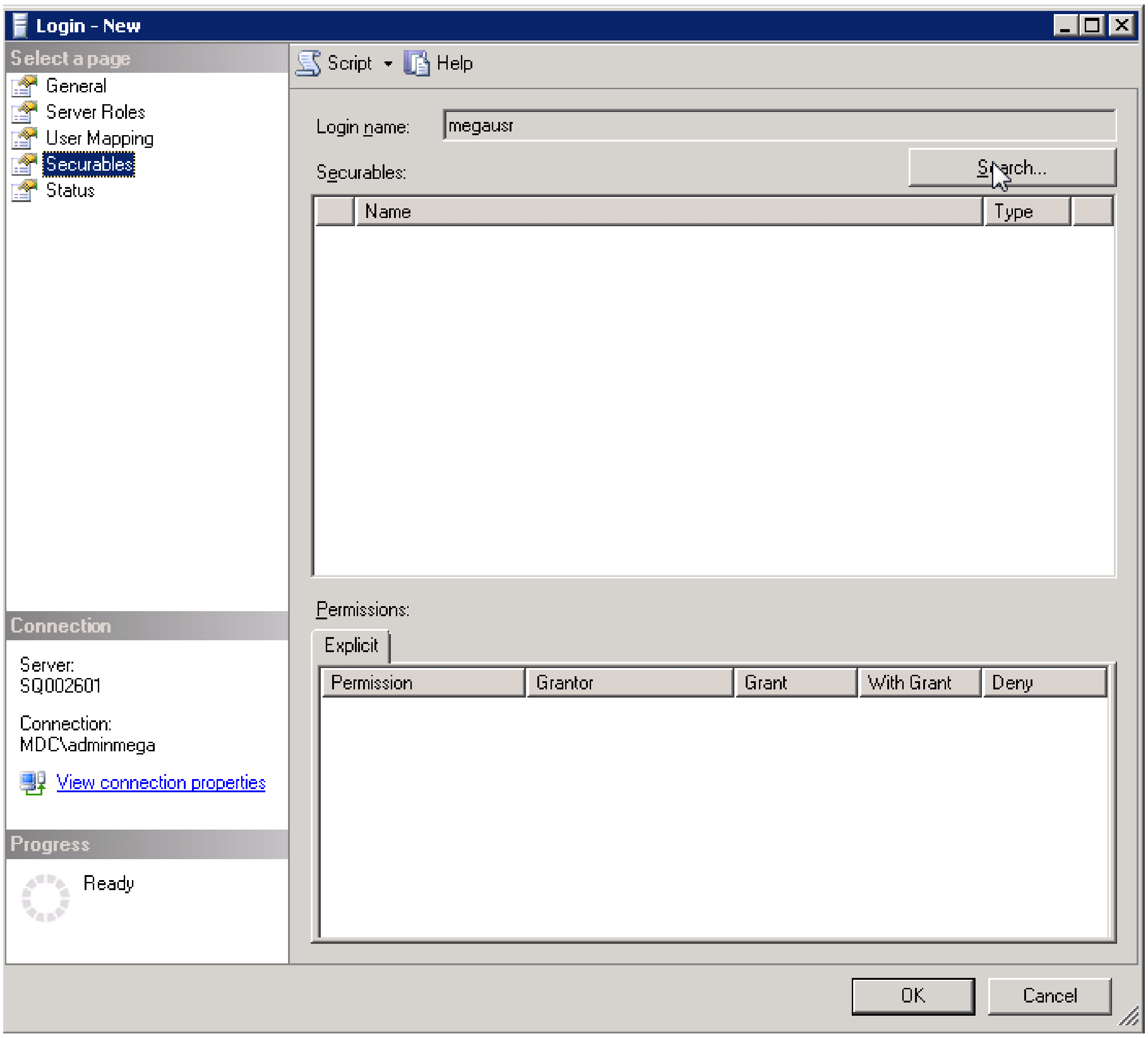
- [ ] bulkadmin
- [✓] dbcreator
- [ ] diskadmin
- [ ] processadmin
- [✓] public
- [ ] securityadmin
- [ ] serveradmin
- [ ] setupadmin
- [ ] sysadmin

Server:
SQ002601

Connection:
MDC\adminmega

Ready

10. In the **Select a page** pane, select **Securables**.
11. In the right pane, click **Search**.



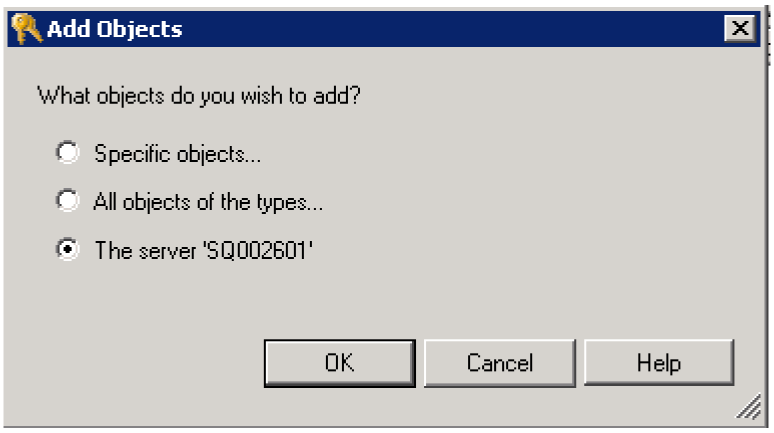
Login name: megausr

Securables:

Server:
SQ002601

Connection:
MDC\adminmega

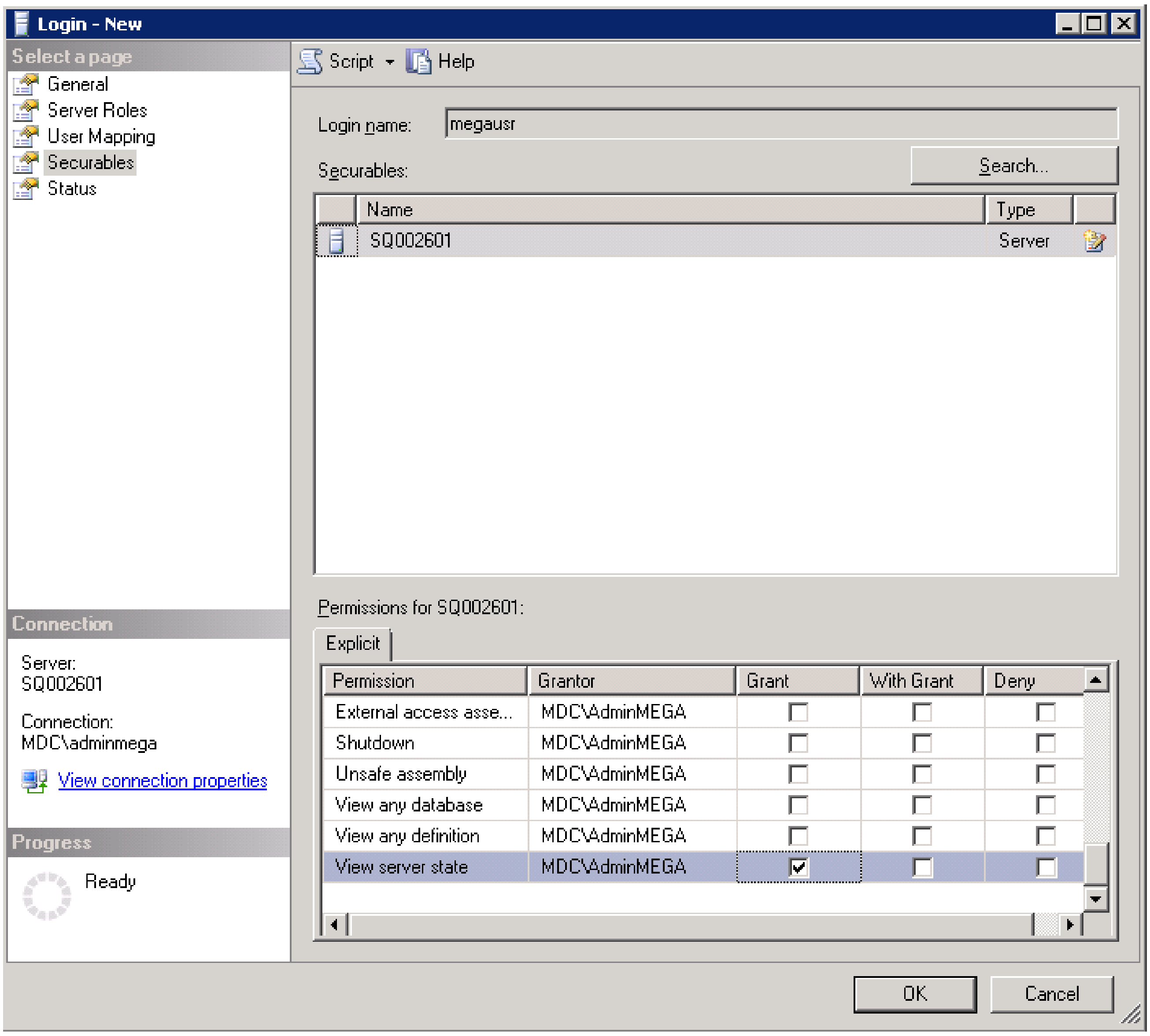
12. Select the instance to browser for its objects. Here we were on instance SQL002601 :



What objects do you wish to add?

- ○ Specific objects...
- ○ All objects of the types...
- [✓ The server 'SQ002601']

13. Click **OK**.
14. In the **Explicit** tab, for the **View Server state** permission select « Grant ».



Login name: megausr

Securables:

Name	Type
SQ002601	Server

Server: SQ002601

Connection: MDC\adminmega

Permissions for SQ002601:

Permission	Grantor	Grant	With Grant	Deny
External access asse...	MDC\AdminMEGA	☐	☐	☐
Shutdown	MDC\AdminMEGA	☐	☐	☐
Unsafe assembly	MDC\AdminMEGA	☐	☐	☐
View any database	MDC\AdminMEGA	☐	☐	☐
View any definition	MDC\AdminMEGA	☐	☐	☐
View server state	MDC\AdminMEGA	☑	☐	☐

15. Click **OK** to create the user.

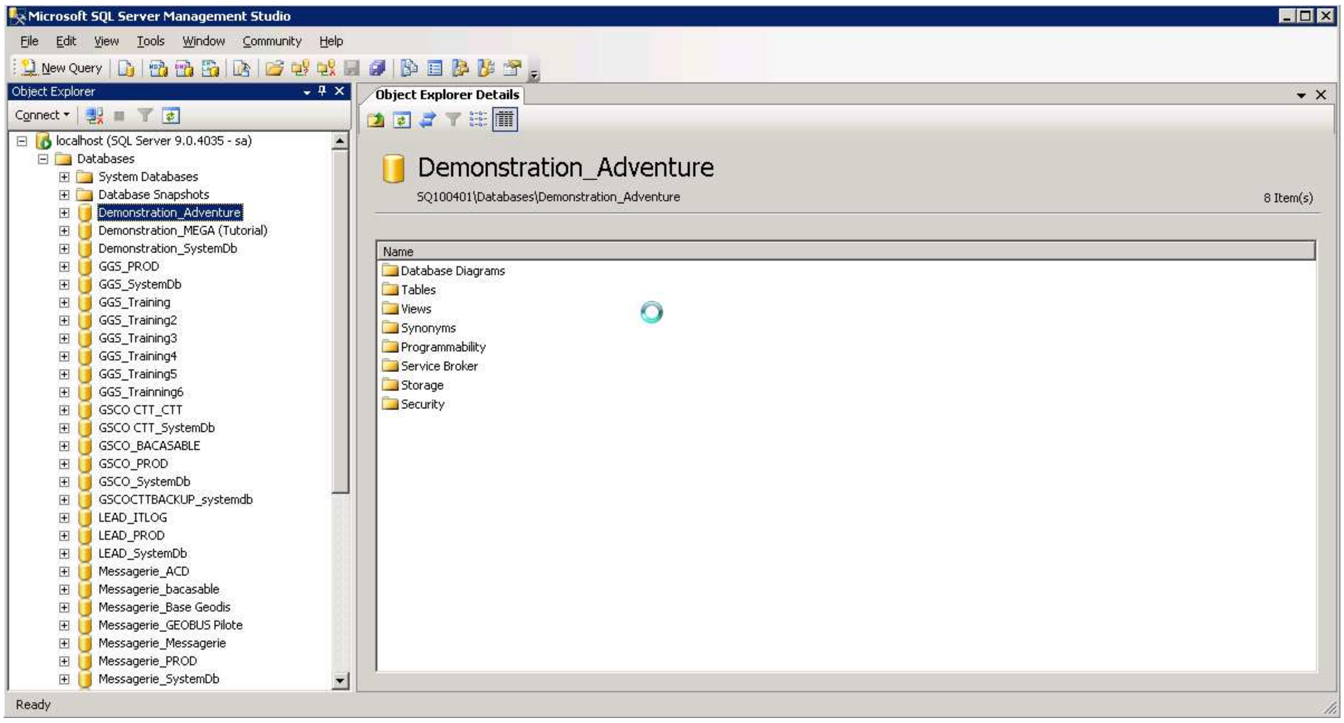
Backup and file transfer

1. Connect to the server hosting the source database.

   Example: SQ100401.

2. Launch **Microsoft SQL Server Management Studio**.

   If possible, connect with a Windows user that will have been granted sysadmin rights on the instance. Otherwise, you might encounter issues when creating the backup file to a specific location, or later, when you need to restore the database on the target instance.



localhost (SQL Server 9.0.4035 - sa)
- Databases
  - System Databases
  - Database Snapshots
  - Demonstration_Adventure
  - Demonstration_MEGA (Tutorial)
  - Demonstration_SystemDb
  - GGS_PROD
  - GGS_SystemDb
  - GGS_Training
  - GGS_Training2
  - GGS_Training3
  - GGS_Training4
  - GGS_Training5
  - GGS_Trainning6
  - GSCO CTT_CTT
  - GSCO CTT_SystemDb
  - GSCO_BACASABLE
  - GSCO_PROD
  - GSCO_SystemDb
  - GSCOCTTBACKUP_systemdb
  - LEAD_ITLOG
  - LEAD_PROD
  - LEAD_SystemDb
  - Messagerie_ACD
  - Messagerie_bacasable
  - Messagerie_Base Geodis
  - Messagerie_GEOBUS Pilote
  - Messagerie_Messagerie
  - Messagerie_PROD
  - Messagerie_SystemDb

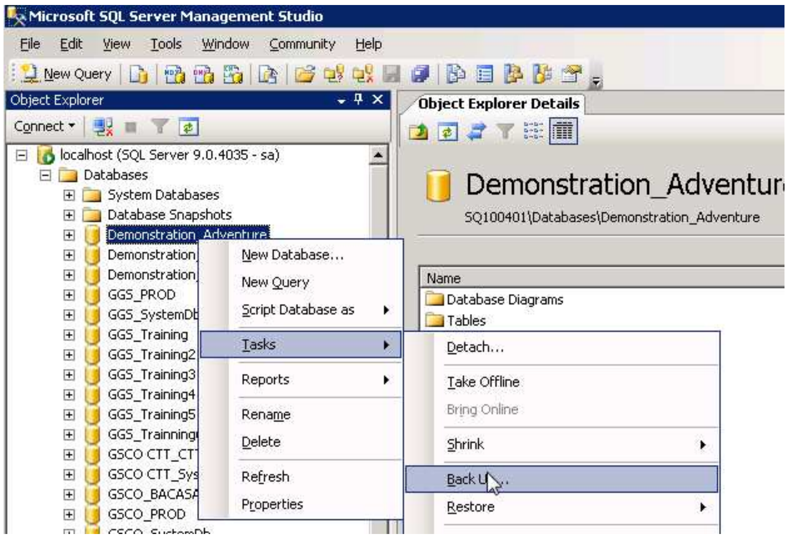
Demonstration_Adventure

SQ100401\Databases\Demonstration_Adventure

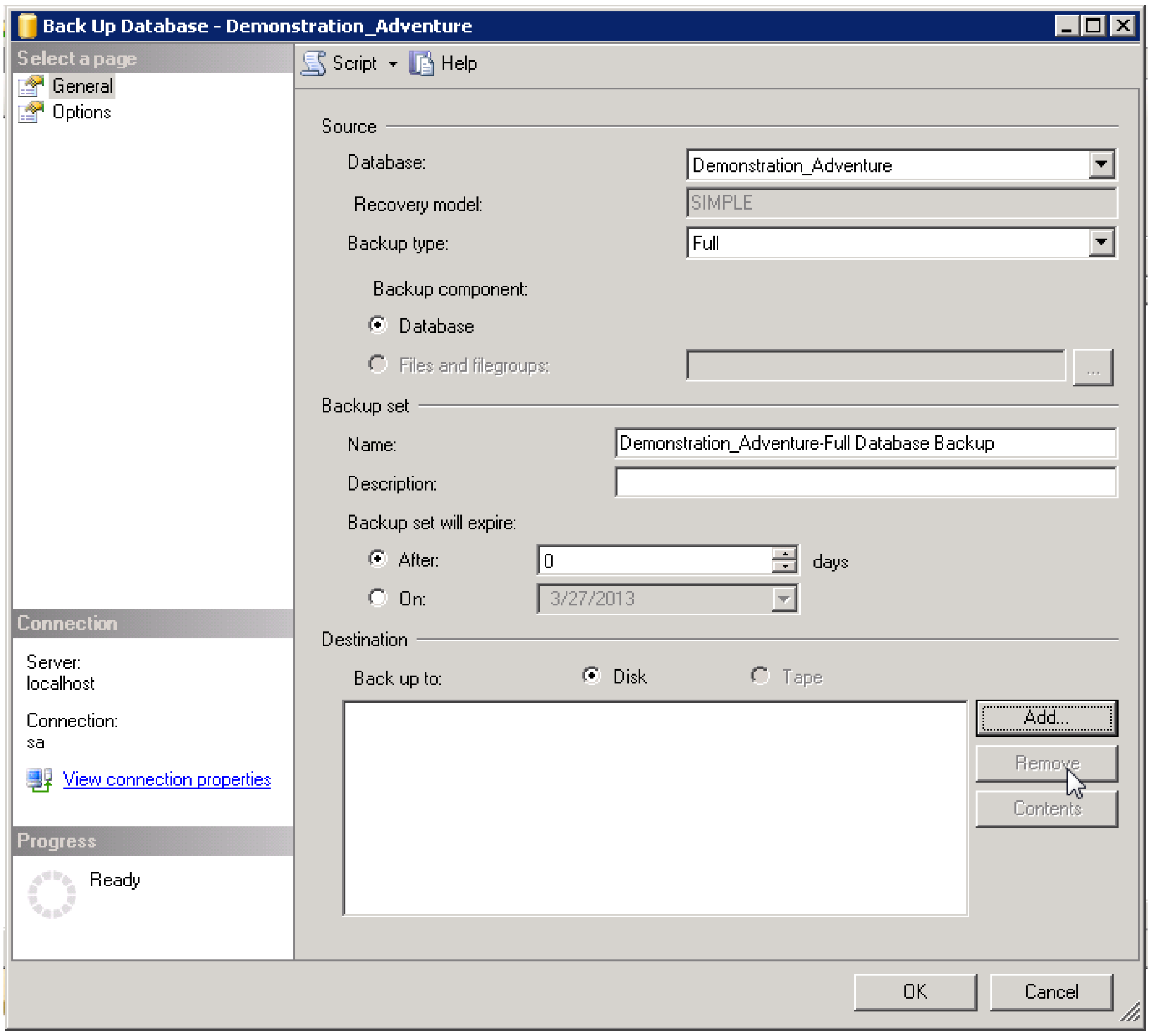
8 Item(s)

Name
Database Diagrams
Tables
Views
Synonyms
Programmability
Service Broker
Storage
Security

3. To make a backup of the source database (for example, Demonstration_Adventure), right-click the database and select **Tasks > Back Up**.



The **Back Up Database** window appears.


Back Up Database - Demonstration_Adventure

Source

Database:	Demonstration_Adventure
Recovery model:	SIMPLE
Backup type:	Full

Backup component: [✓ Database] Files and filegroups:

Backup set

Name:	Demonstration_Adventure-Full Database Backup
Description:	

Backup set will expire: [✓ After: 0 days] On: 3/27/2013

Destination

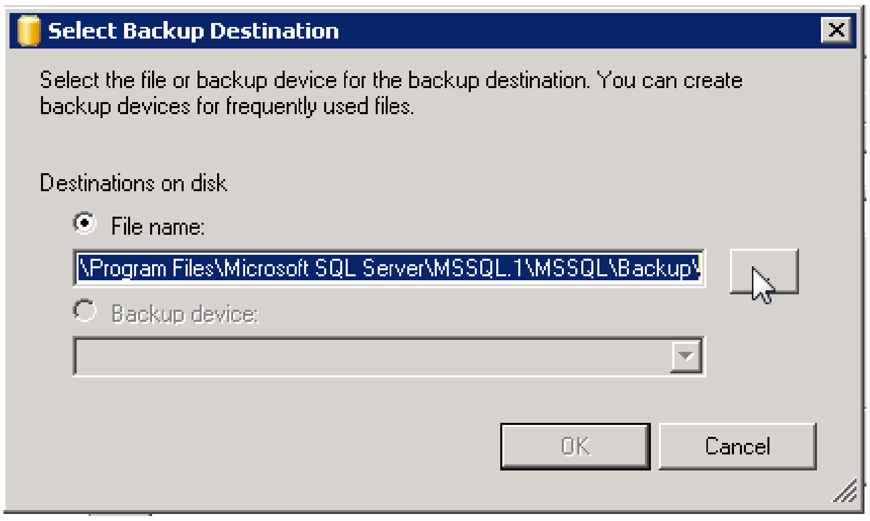
Back up to: [✓ Disk] Tape

Connection
Server: localhost
Connection: sa

Progress
Ready

4. In the right pane, in the **Destination** pane, make sure that the destination list is empty. If it isn't, select each line and click **Remove**. Once it is empty, click **Add**.

5. In the **Select BackUp Destination**, click ....



Select the file or backup device for the backup destination. You can create backup devices for frequently used files.

Destinations on disk

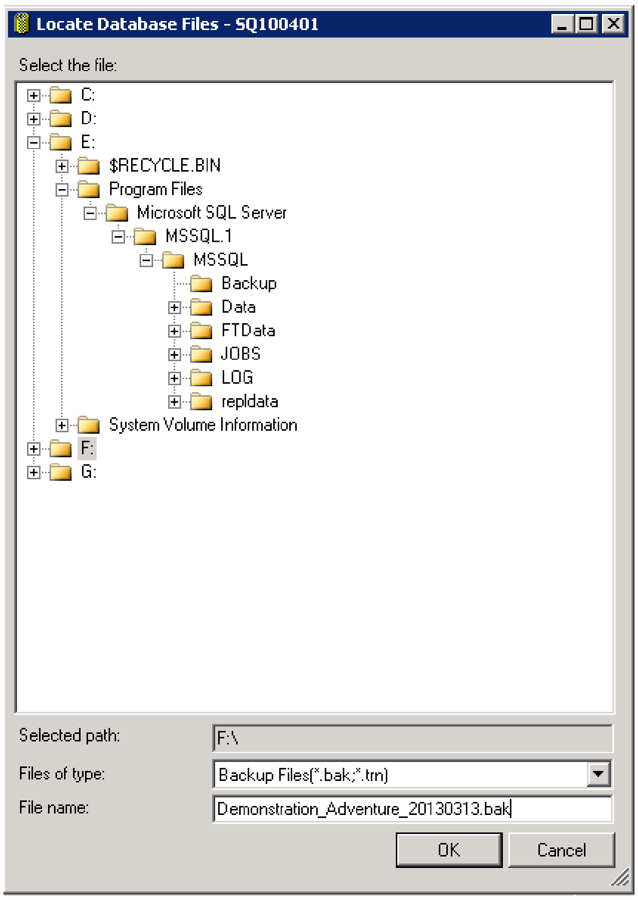
(●) File name:

\Program Files\Microsoft SQL Server\MSSQL.1\MSSQL\Backup\

( ) Backup device:

6. Choose a location where you know that the user you authenticated with, has rights to write on. In this example in « F:\SQL_BACKUP » of the F drive, and give the name of the backup file to create (here Demonstration_Adventure_20130313.bak).

   Please note that the known format of a full backup in SQL Server is **.BAK** files. You have to explicitely put it in the file name, otherwise it will not have any format.



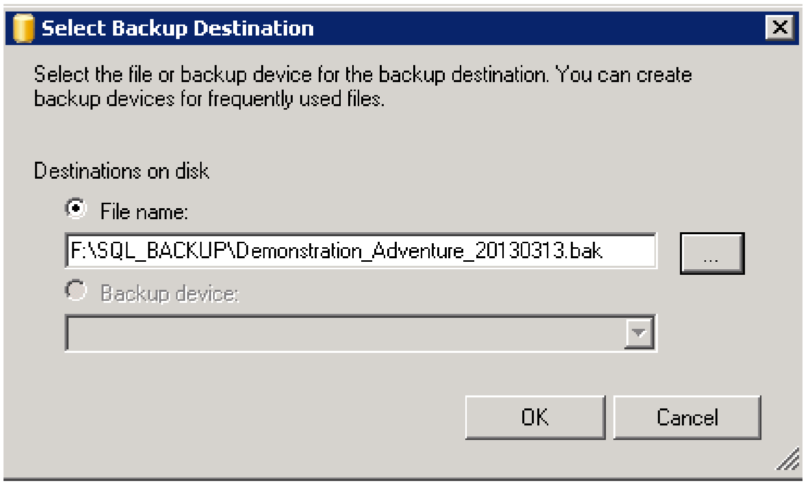
Select the file:

Selected path: F:\

Files of type: Backup Files(*.bak;*.trn)

File name: Demonstration_Adventure_20130313.bak

7. Click **OK**.



Select Backup Destination

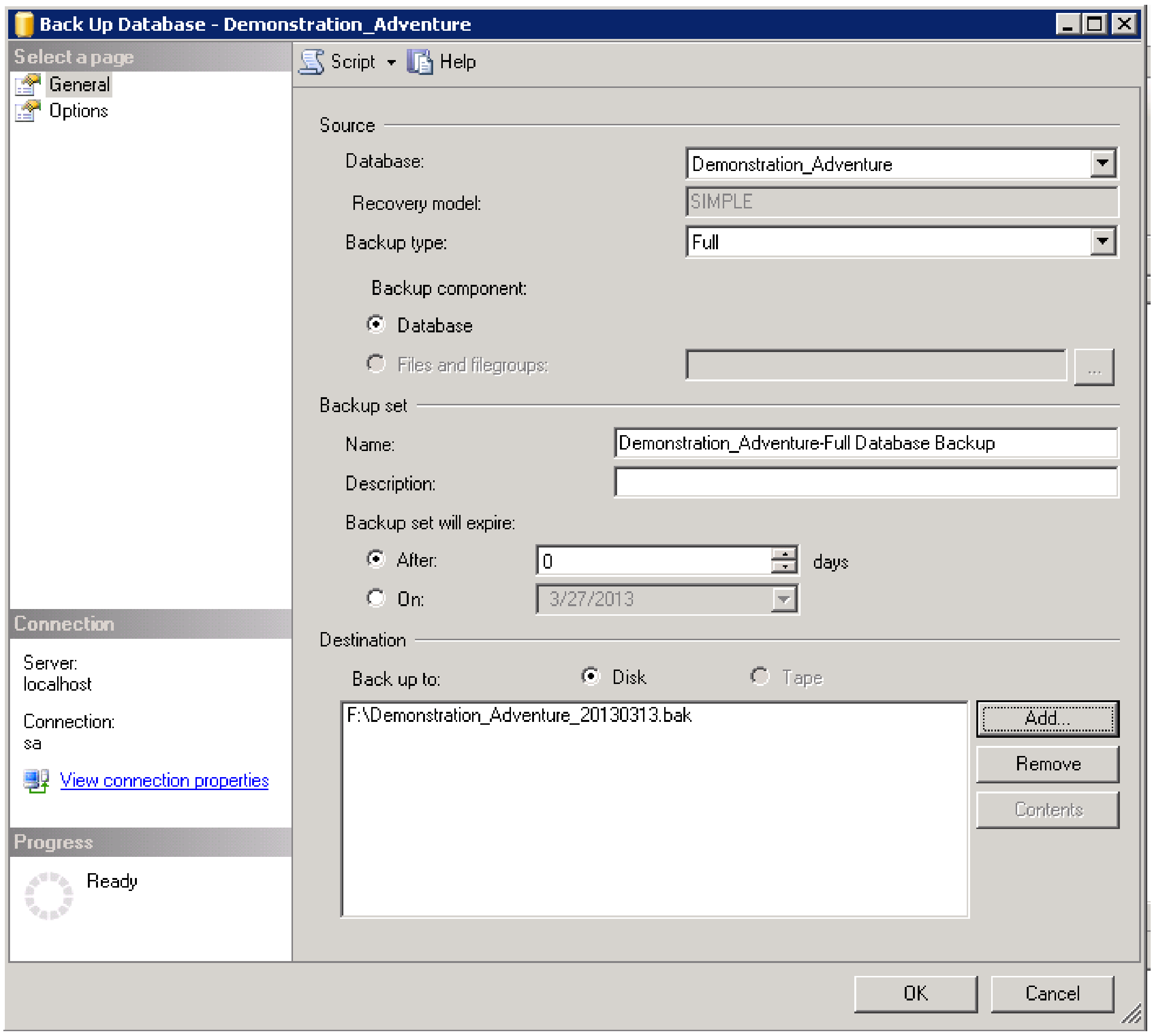
Select the file or backup device for the backup destination. You can create backup devices for frequently used files.

Destinations on disk

[✓ File name:] F:\SQL_BACKUP\Demonstration_Adventure_20130313.bak

Backup device:

8. Click **OK**.



Back Up Database - Demonstration_Adventure

Source

Database: Demonstration_Adventure

Recovery model: SIMPLE

Backup type: Full

Backup component:

[✓ Database] Files and filegroups:

Backup set

Name: Demonstration_Adventure-Full Database Backup

Description:

Backup set will expire:

[✓ After:] 0 days

On: 3/27/2013

Destination

Back up to: [✓ Disk] Tape

F:\Demonstration_Adventure_20130313.bak

Connection

Server:
localhost

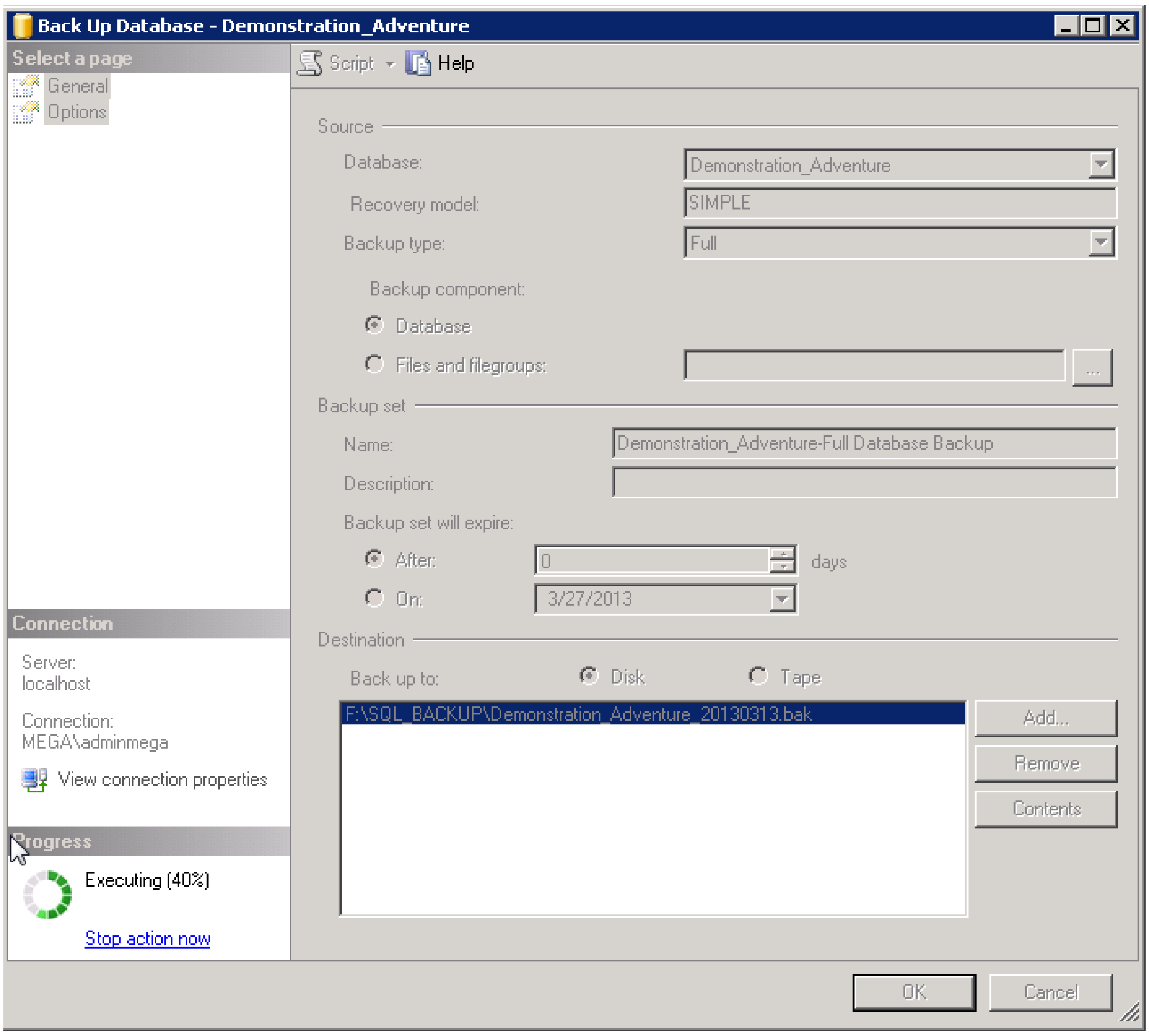
Connection:
sa

View connection properties

Progress

Ready

9. Click **OK** and check the progress of the restore by looking at the left-bottom section of this window (here, we are 40% done with the restore).


Back Up Database - Demonstration_Adventure

Select a page: General, Options

Source
Database: Demonstration_Adventure
Recovery model: SIMPLE
Backup type: Full
Backup component:
(●) Database
( ) Files and filegroups:

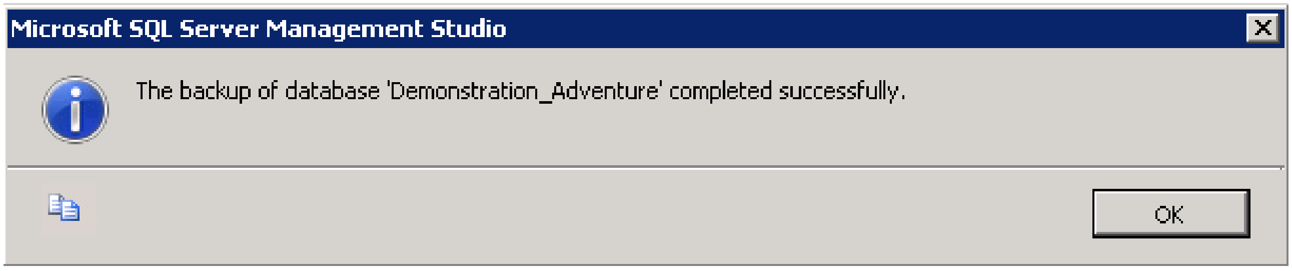
Backup set
Name: Demonstration_Adventure-Full Database Backup
Description:
Backup set will expire:
(●) After: 0 days
( ) On: 3/27/2013

Destination
Back up to: (●) Disk ( ) Tape
F:\SQL_BACKUP\Demonstration_Adventure_20130313.bak

Connection
Server: localhost
Connection: MEGA\adminmega

Progress
Executing (40%)
Stop action now

10. Check that the database was fully restored, and click **OK**.

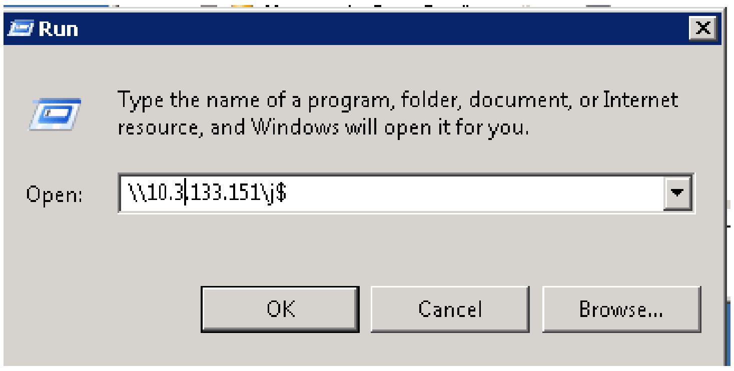

Microsoft SQL Server Management Studio

The backup of database 'Demonstration_Adventure' completed successfully.

11. Transfer the backup file created on the SQL Server server, to the target server (for example, here it is SQL002601).

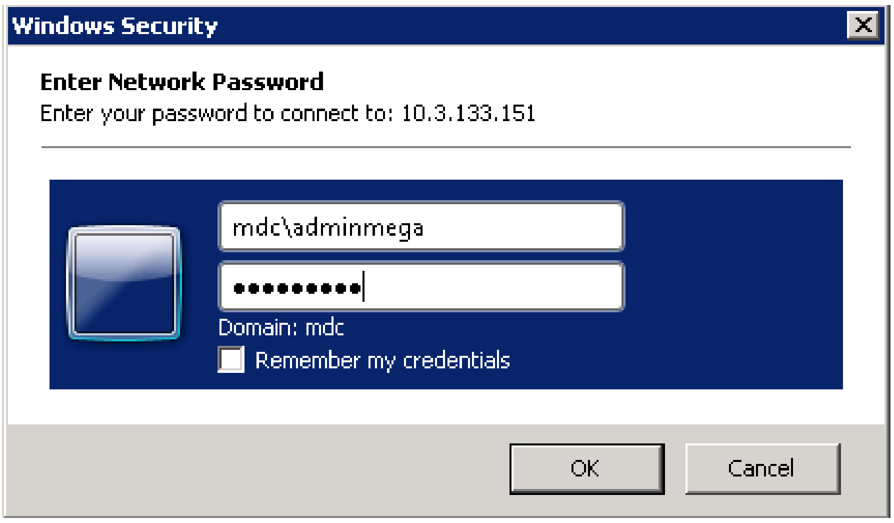
In this example, the drive hosting the databases on the target instance, as well as the daily backups, is the J drive.

We used its IP address instead of its name, as we were working over two different domains, that did not see each other.


Run
Type the name of a program, folder, document, or Internet resource, and Windows will open it for you.
OK
Cancel
Browse...

Open: \\10.3.133.151\j$

This is also the reason why we had to authenticate with another user.



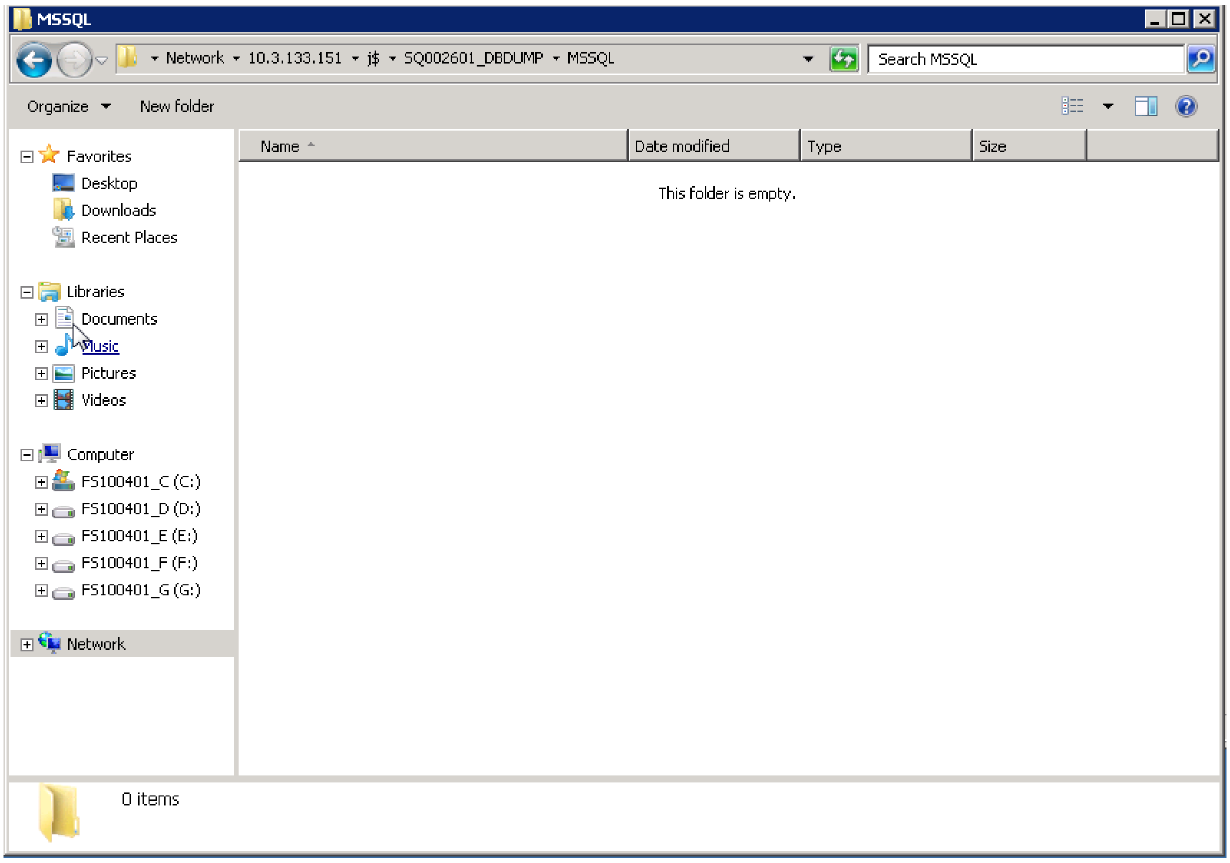
Enter your password to connect to: 10.3.133.151

mdc\adminmega

••••••••••

Domain: mdc

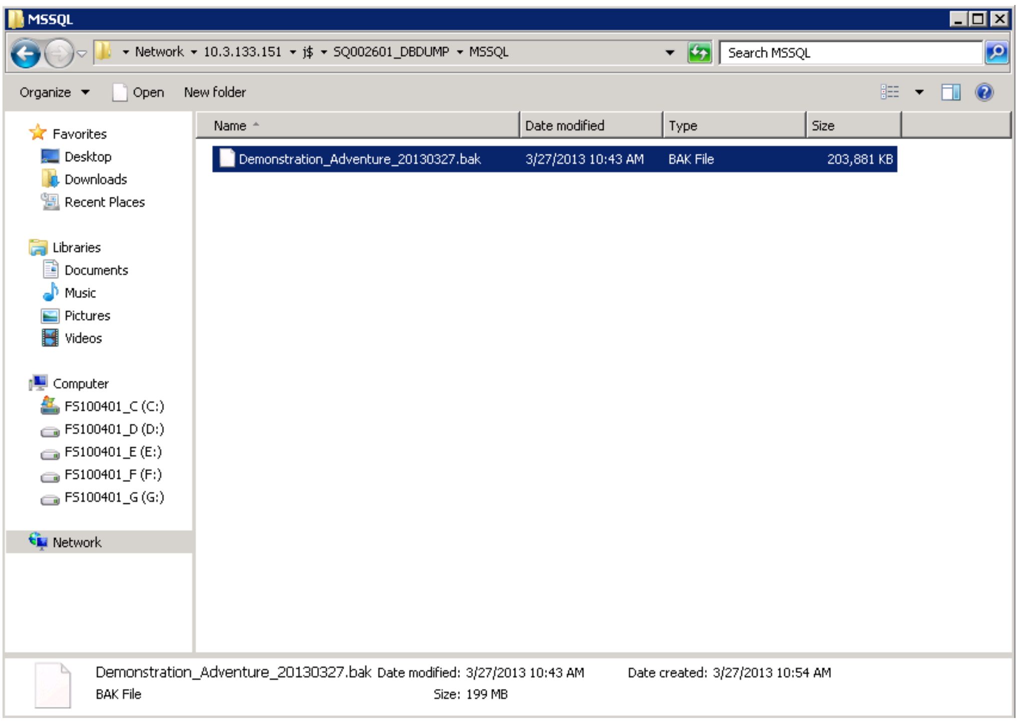
12. Go to the subfolder hosting the daily backups.
    (for example: « SQ002601_DBDUMP\MSSQL » on the J drive)



This folder is empty.

0 items

13. Copy the file from the source folder to this one.

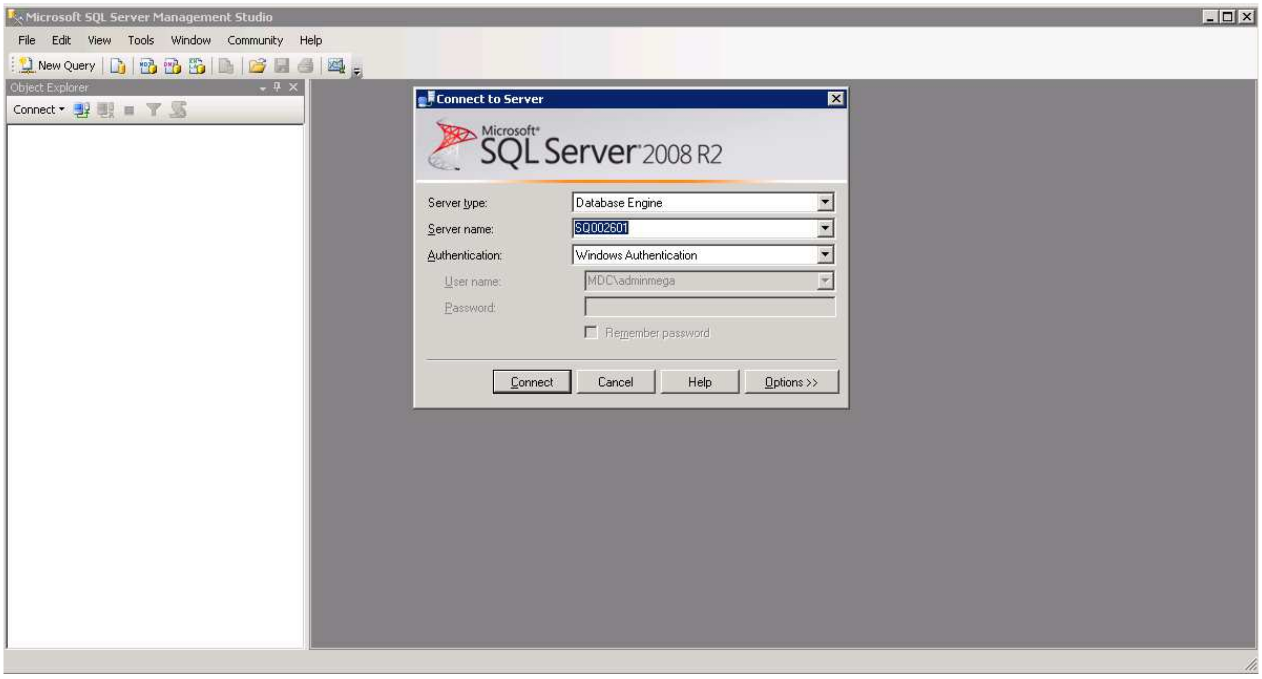


Name	Date modified	Type	Size
Demonstration_Adventure_20130327.bak	3/27/2013 10:43 AM	BAK File	203,881 KB

Demonstration_Adventure_20130327.bak Date modified: 3/27/2013 10:43 AM Date created: 3/27/2013 10:54 AM
BAK File Size: 199 MB

Restore

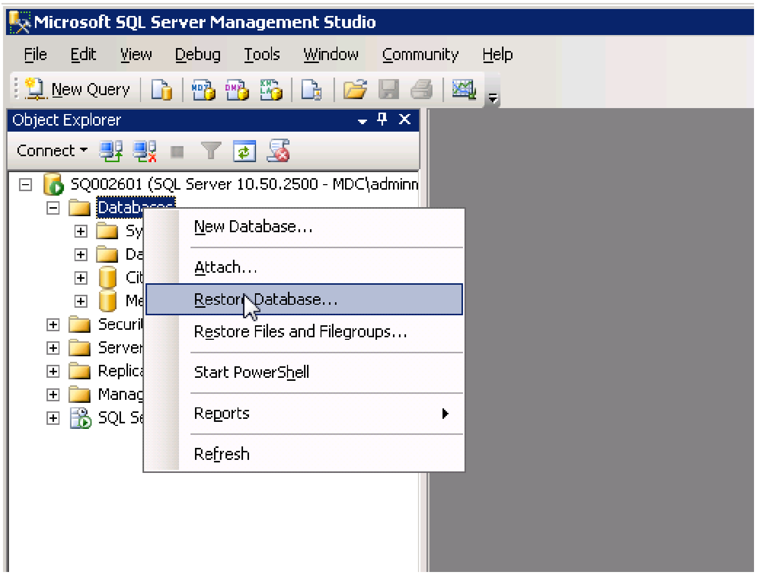
1. Connect to the target SQL Server server (example: SQL002601).
2. Launch the Management Studio, and connect to the instance using, if possible, a Windows account that is both sysadmin, and has access rights to the folder where the backup file was copied (example: mdc\adminmega).


Microsoft SQL Server Management Studio
File Edit View Tools Window Community Help
New Query
Object Explorer
Connect
Connect to Server
Microsoft SQL Server 2008 R2

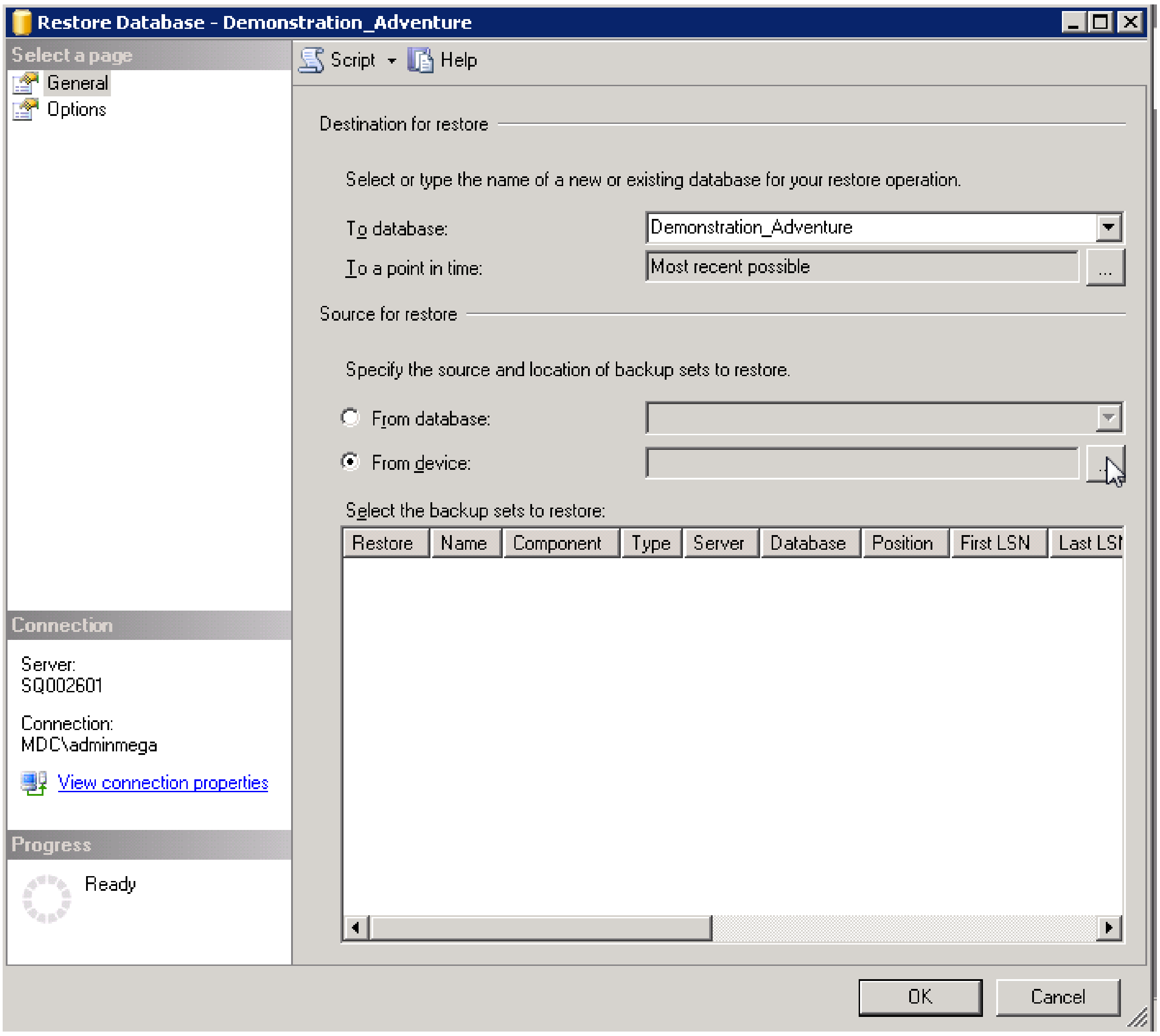
Server type: Database Engine
Server name: SQ002601
Authentication: Windows Authentication
User name: MDC\adminmega
Password:
Remember password

Connect
Cancel
Help
Options >>

3. Right-click **Databases**, and select **Restore Database**.


Microsoft SQL Server Management Studio
File Edit View Debug Tools Window Community Help
New Query
Object Explorer
Connect
SQ002601 (SQL Server 10.50.2500 - MDC\adminn
Databases
Sy
Da
Cit
Me
Securi
Server
Replica
Manag
SQL Se
New Database...
Attach...
Restore Database...
Restore Files and Filegroups...
Start PowerShell
Reports
Refresh

4. From the **General** page, in the **Destination for restore** pane, in the **To database** field, provide the name of the database that will be created (example: Demonstration_Adventure).
5. In the **Source for restore** pane, select **From Device** option and click ....



Destination for restore

Select or type the name of a new or existing database for your restore operation.

To database: Demonstration_Adventure

To a point in time: Most recent possible

Source for restore

Specify the source and location of backup sets to restore.

( ) From database:

(●) From device:

Select the backup sets to restore:

Restore	Name	Component	Type	Server	Database	Position	First LSN	Last LSI

Connection

Server:
SQ002601

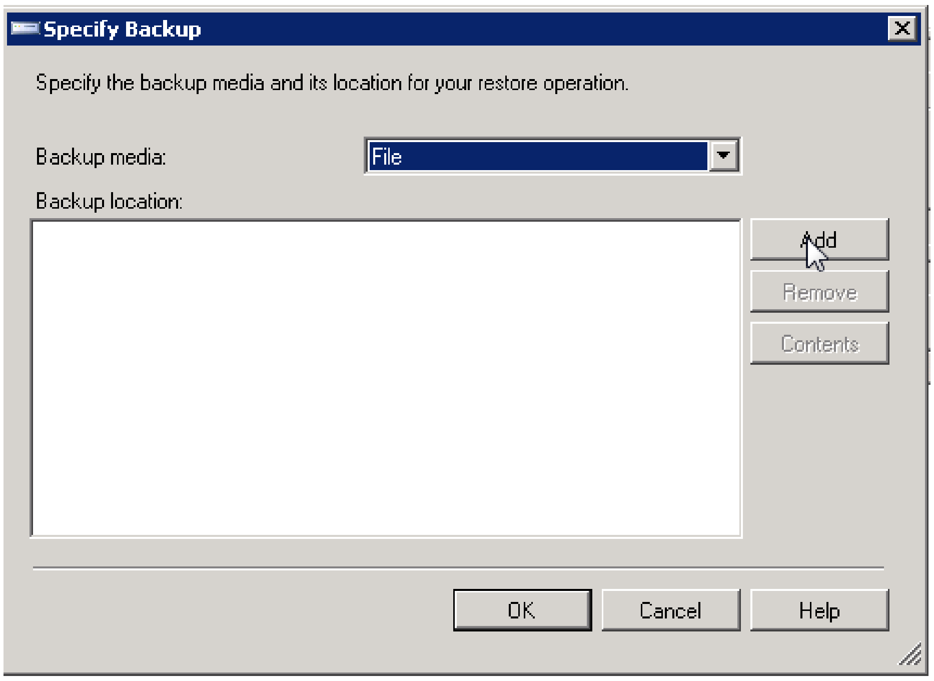
Connection:
MDC\adminmega

View connection properties

Progress

Ready

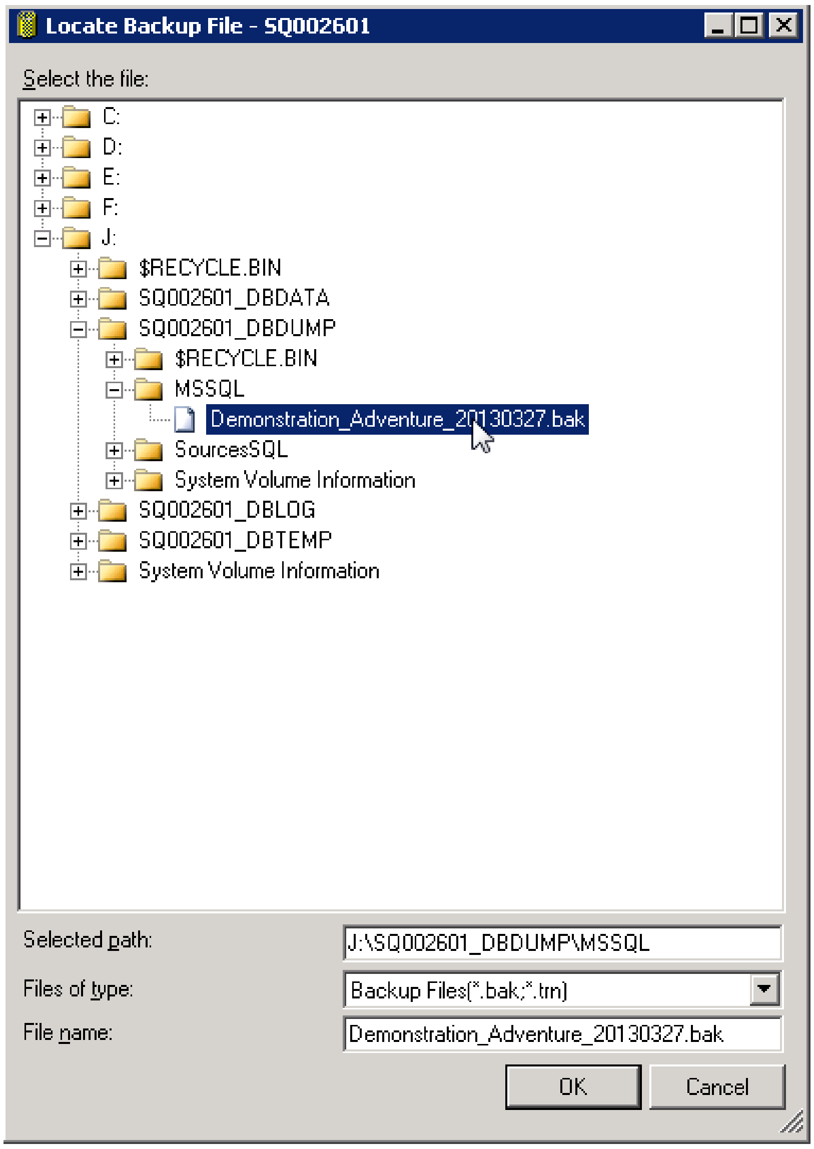
6. Click **Add**.



7. Check that we are correctly put in the folder « J:\SQ002601_DBDUMP\MSSQL».

   Otherwise, browse the folders to get to it. Then, click the .bak file that you just copied on the server (for example: « Demonstration_Adventure_20130327.bak »).

   Check that the « File Name » field is correctly filled, and click **OK**.


Locate Backup File - SQ002601

Select the file:

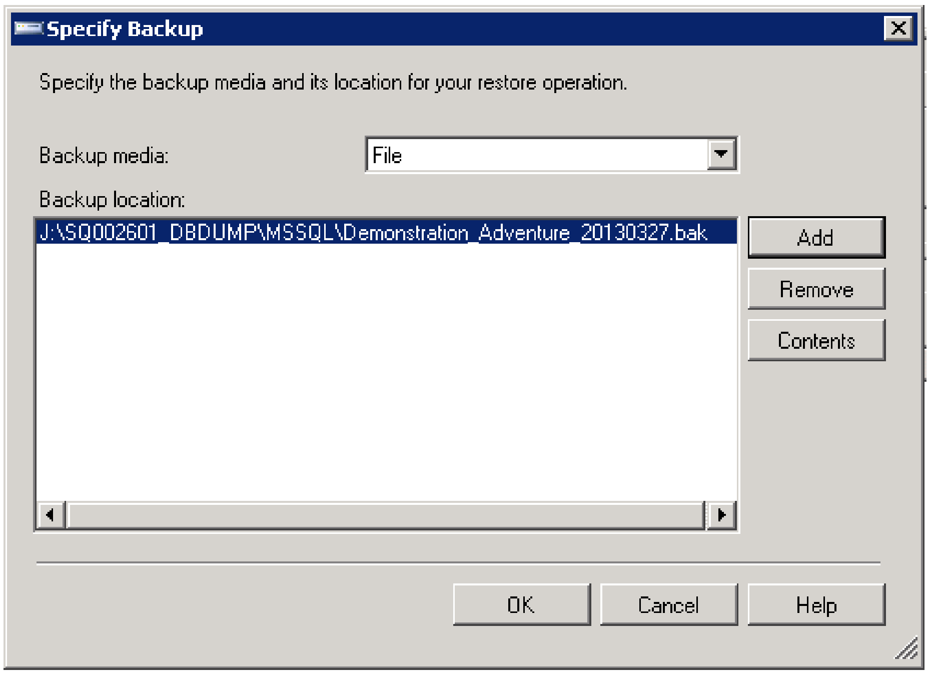
```
C:
D:
E:
F:
J:
  $RECYCLE.BIN
  SQ002601_DBDATA
  SQ002601_DBDUMP
    $RECYCLE.BIN
    MSSQL
      Demonstration_Adventure_20130327.bak
    SourcesSQL
    System Volume Information
  SQ002601_DBLOG
  SQ002601_DBTEMP
  System Volume Information
```

Selected path: J:\SQ002601_DBDUMP\MSSQL

Files of type: Backup Files(*.bak;*.trn)

File name: Demonstration_Adventure_20130327.bak

8. Click **OK.**


Specify Backup

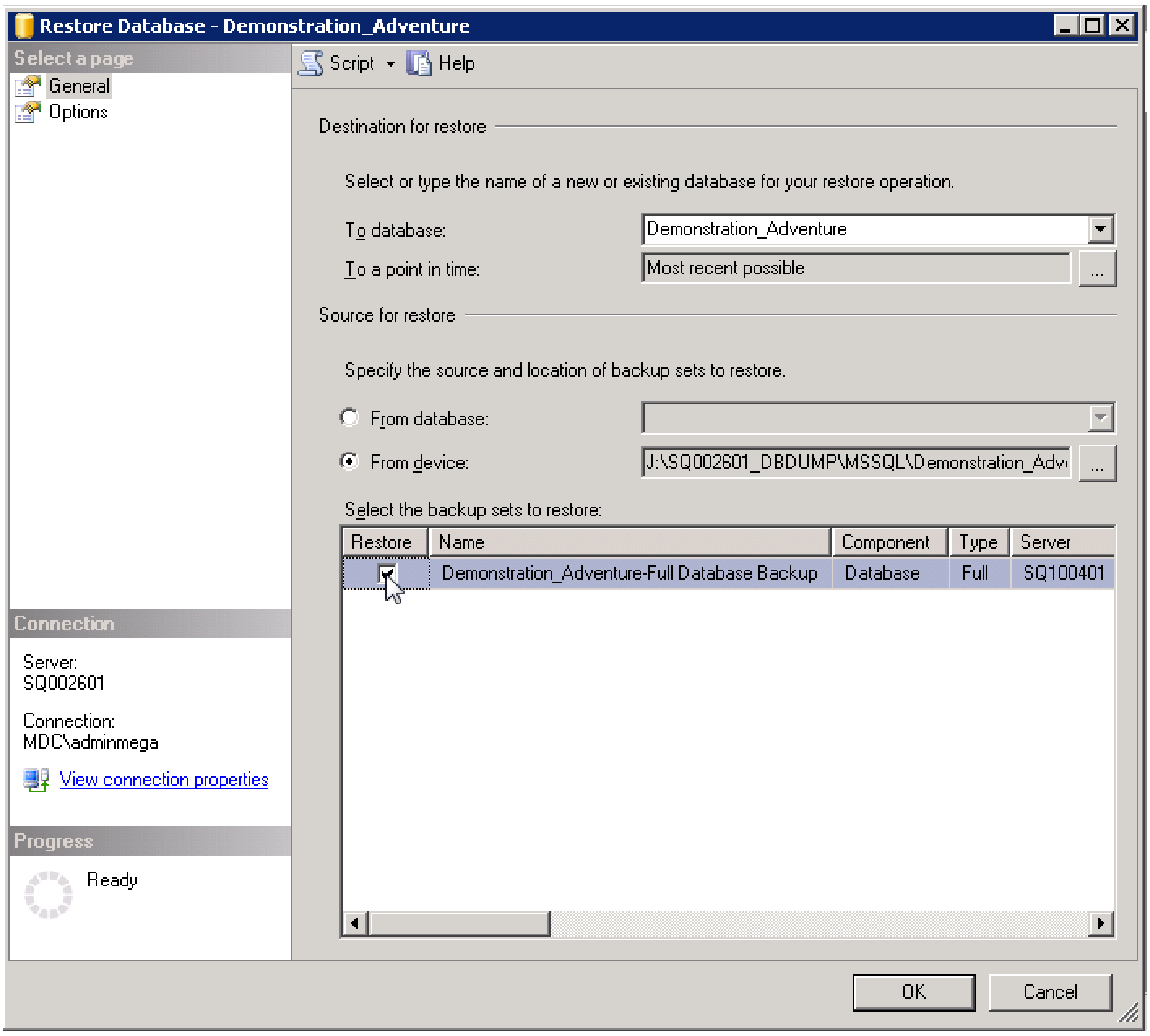
Specify the backup media and its location for your restore operation.

Backup media: File

Backup location:

J:\SQ002601_DBDUMP\MSSQL\Demonstration_Adventure_20130327.bak

9. In the **Source for restore** pane, in the **Select the backup sets to restore** table select **Restore** at the beginning of the line.



Restore Database - Demonstration_Adventure

Destination for restore

Select or type the name of a new or existing database for your restore operation.

To database: Demonstration_Adventure

To a point in time: Most recent possible

Source for restore

Specify the source and location of backup sets to restore.

From database:

From device: J:\SQ002601_DBDUMP\MSSQL\Demonstration_Adv

Select the backup sets to restore:

Restore	Name	Component	Type	Server
✓	Demonstration_Adventure-Full Database Backup	Database	Full	SQ100401

Connection

Server: SQ002601

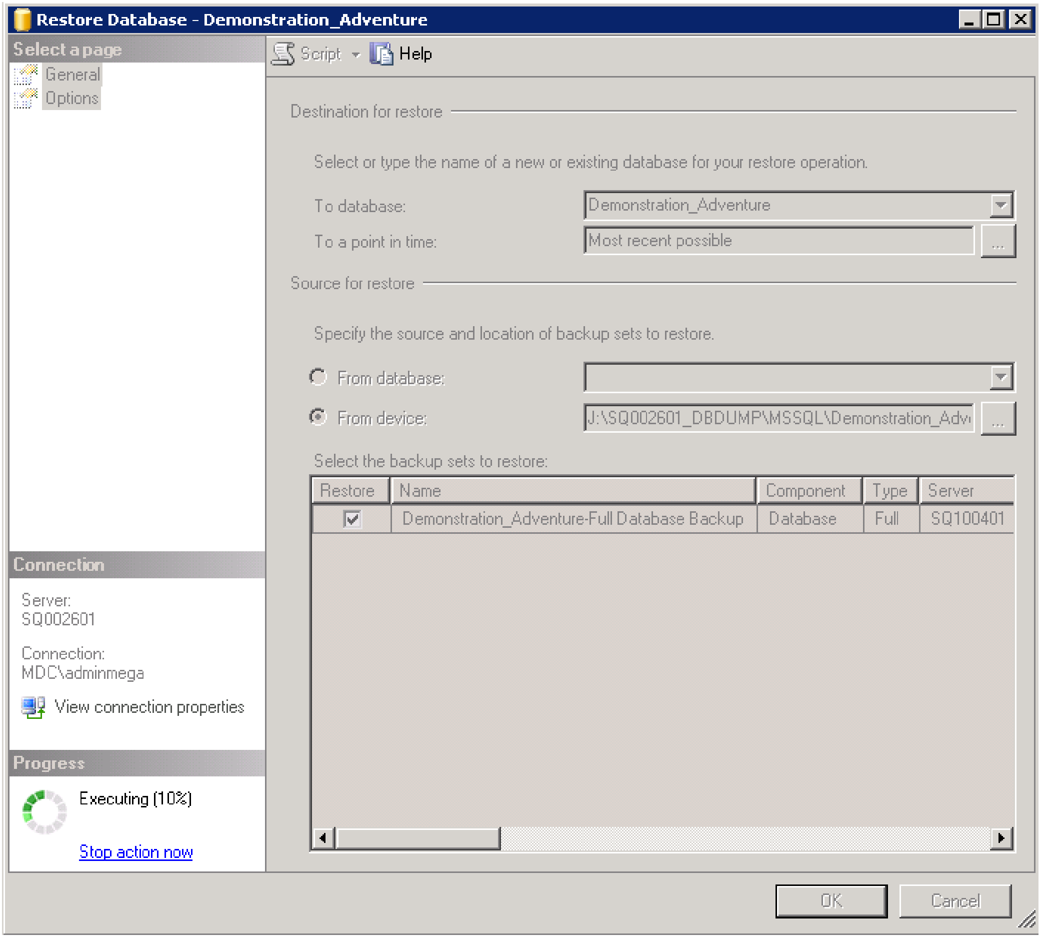
Connection: MDC\adminmega

View connection properties

Progress

Ready

10. click **OK**.


Restore Database - Demonstration_Adventure

Select a page: General, Options

Destination for restore

Select or type the name of a new or existing database for your restore operation.

To database: Demonstration_Adventure

To a point in time: Most recent possible

Source for restore

Specify the source and location of backup sets to restore.

From database:

From device: J:\SQ002601_DBDUMP\MSSQL\Demonstration_Adv

Select the backup sets to restore:

Restore	Name	Component	Type	Server
✓	Demonstration_Adventure-Full Database Backup	Database	Full	SQ100401

Connection

Server: SQ002601

Connection: MDC\adminmega

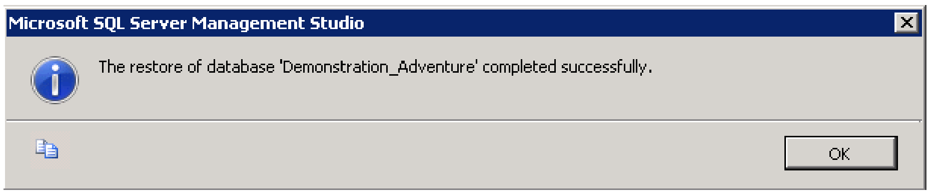
View connection properties

Progress

Executing (10%)

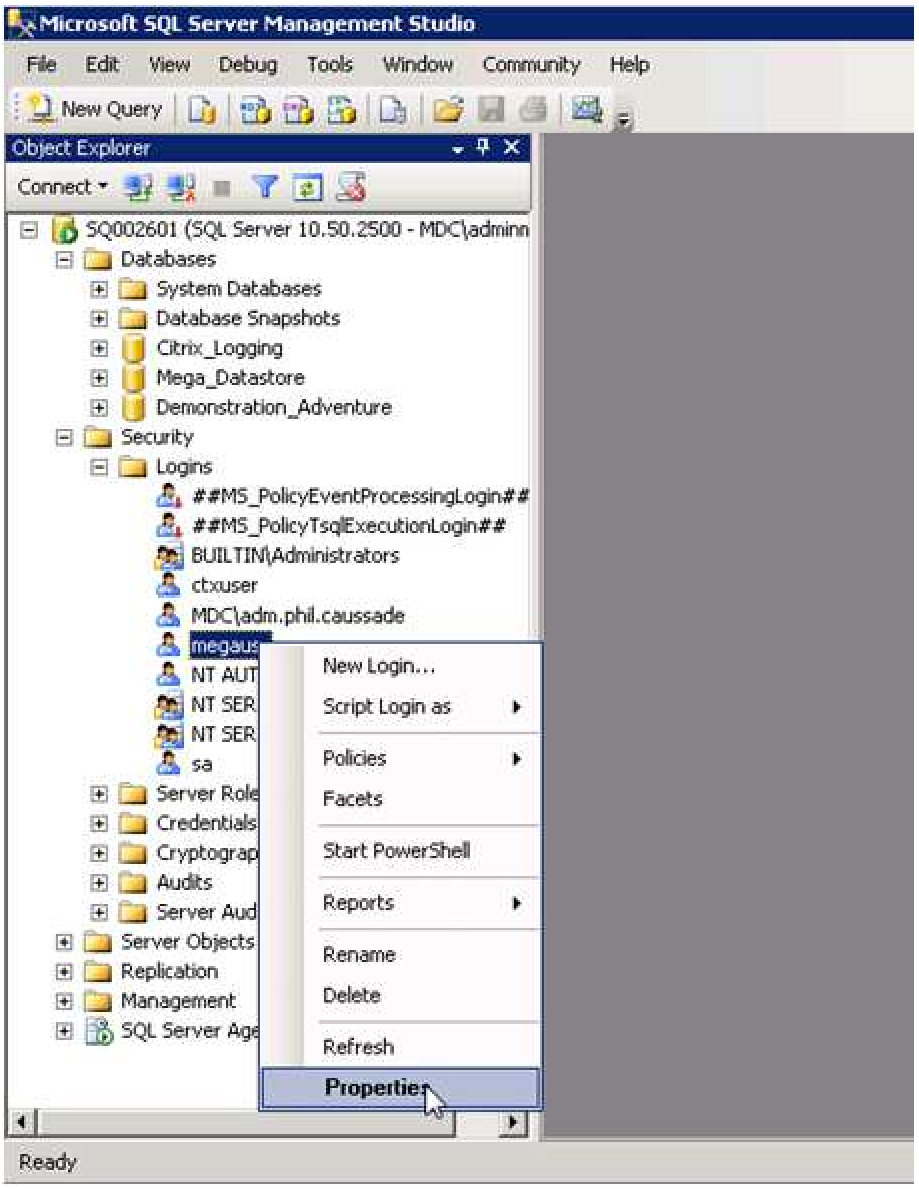
Stop action now

11. Once the restore completed successfully, click **OK**.

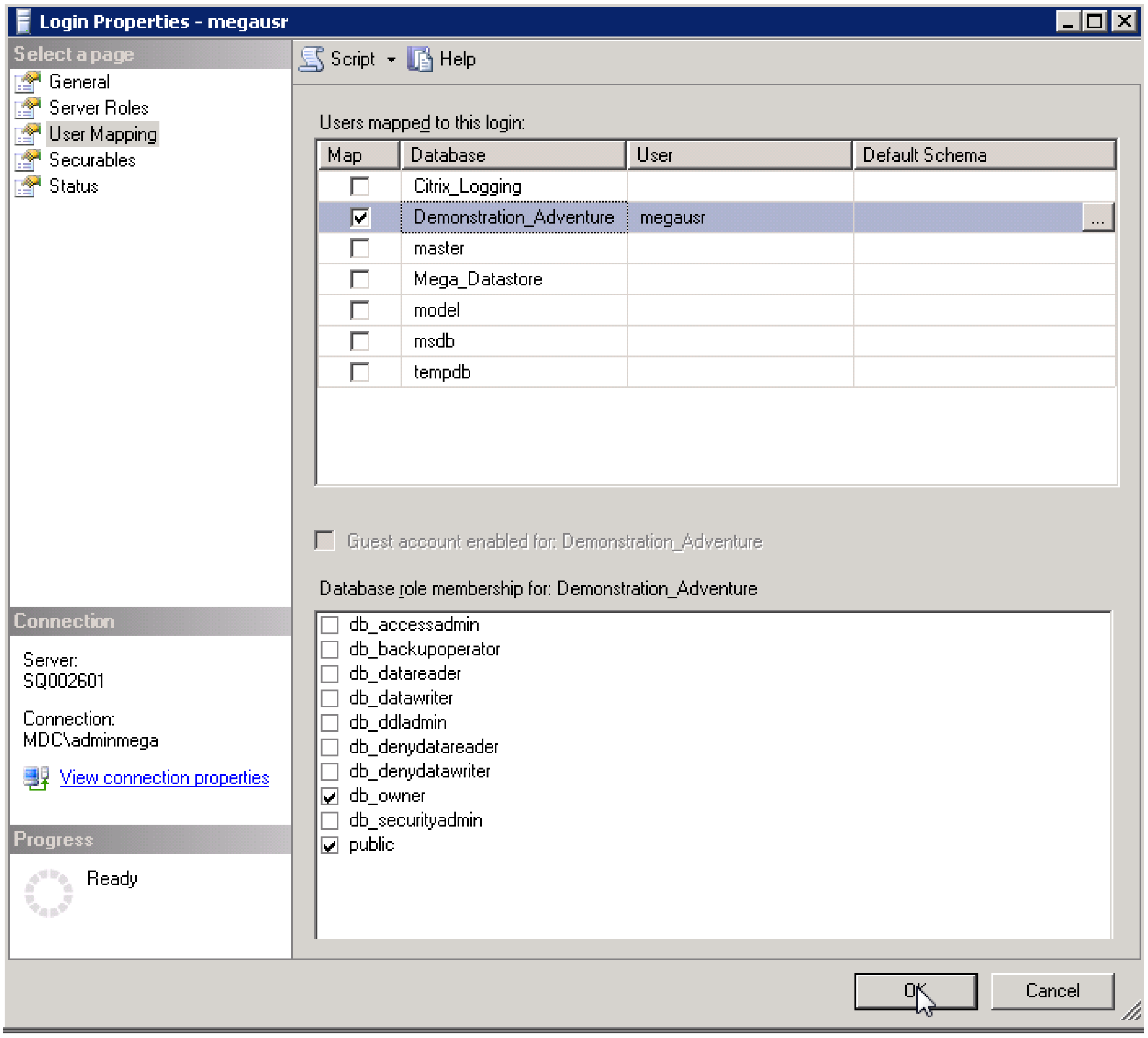

Microsoft SQL Server Management Studio

The restore of database 'Demonstration_Adventure' completed successfully.

12. In ManagementStudio, expand **Security** folder, and **Logins** folder.
13. Right-click the account that you will use to connect the application to SQL Server (for example: login « megausr ») and select **Properties**.



14. In the **Login Properties - <login>** window, select **User Mapping** page, and select the **Map** corresponding to the database lin you just restored.
15. In the **Database role membering for: <database name>** pane, select « db_owner ».
16. Click **OK**.



Login Properties - megausr

Select a page: General, Server Roles, User Mapping, Securables, Status

Users mapped to this login:

Map	Database	User	Default Schema
☐	Citrix_Logging		
✓	Demonstration_Adventure	megausr	
☐	master		
☐	Mega_Datastore		
☐	model		
☐	msdb		
☐	tempdb		

☐ Guest account enabled for: Demonstration_Adventure

Database role membership for: Demonstration_Adventure

- ☐ db_accessadmin
- ☐ db_backupoperator
- ☐ db_datareader
- ☐ db_datawriter
- ☐ db_ddladmin
- ☐ db_denydatareader
- ☐ db_denydatawriter
- ✓ db_owner
- ☐ db_securityadmin
- ✓ public

Connection

Server:
SQ002601

Connection:
MDC\adminmega

Progress: Ready

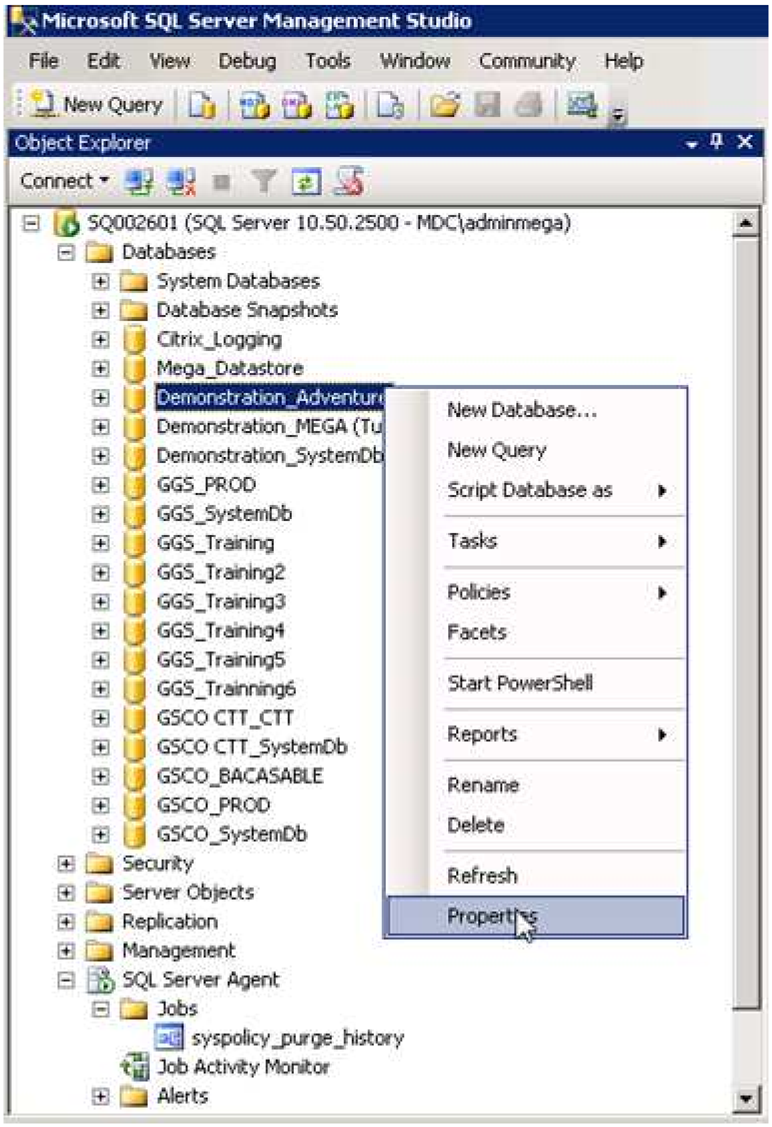
IF you moved from a version of SQL Server to a more recent one, you have to upgrade the compatibility level of your database.

In this example, the source database came from an SQL Server 2005 instance, and was restore on an SQL Server 2008 instance.

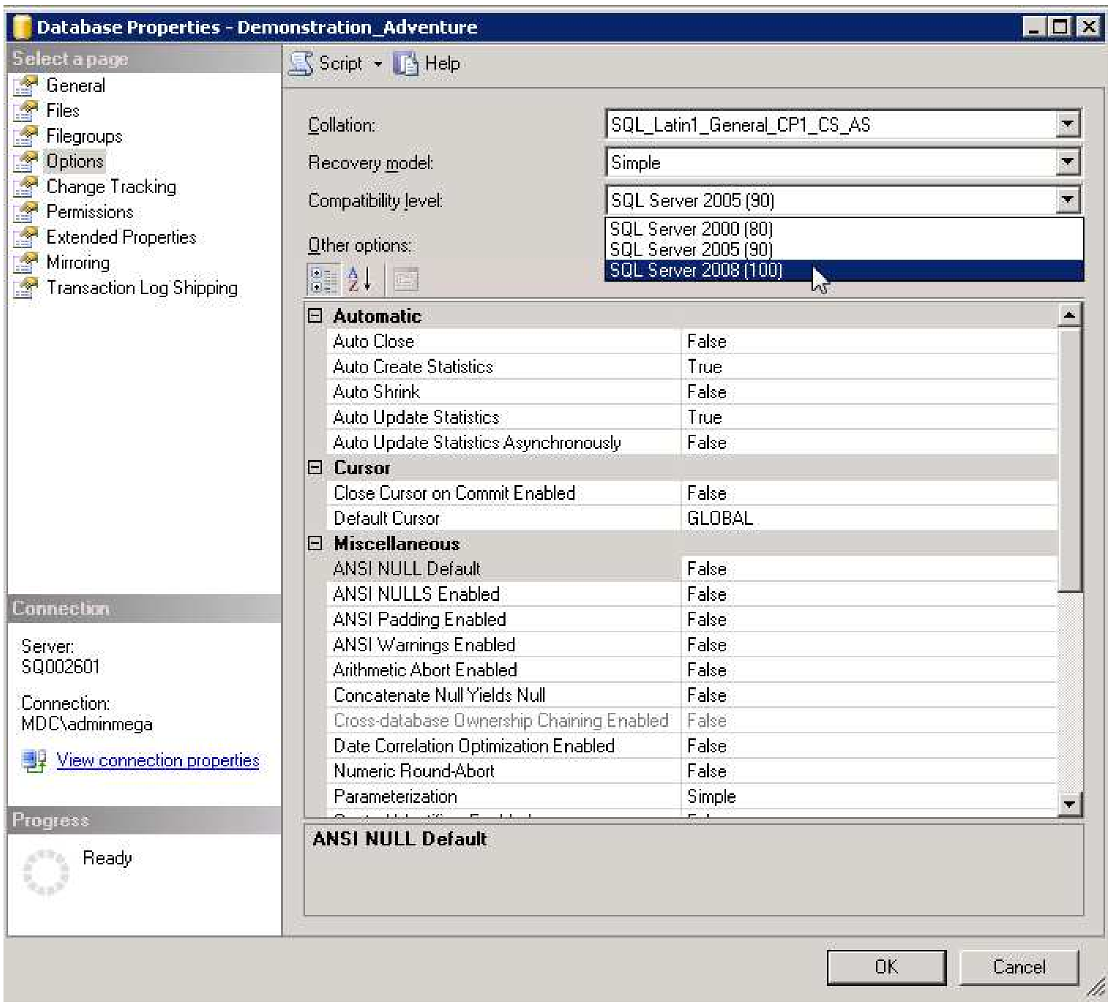
If you stay on the same version, go directly to the next section.

To upgrade:

1. Right-click the database and select **Properties**.



2. Select the **Options** page, in the **Compatibility level** drop down list select the appropriate version (for example « SQL Server 2008 (100) »).


Database Properties - Demonstration_Adventure

Collation: SQL_Latin1_General_CP1_CS_AS

Recovery model: Simple

Compatibility level: SQL Server 2005 (90)

SQL Server 2000 (80)
SQL Server 2005 (90)
SQL Server 2008 (100)

Other options:

Automatic	
Auto Close	False
Auto Create Statistics	True
Auto Shrink	False
Auto Update Statistics	True
Auto Update Statistics Asynchronously	False
Cursor	
Close Cursor on Commit Enabled	False
Default Cursor	GLOBAL
Miscellaneous	
ANSI NULL Default	False
ANSI NULLS Enabled	False
ANSI Padding Enabled	False
ANSI Warnings Enabled	False
Arithmetic Abort Enabled	False
Concatenate Null Yields Null	False
Cross-database Ownership Chaining Enabled	False
Date Correlation Optimization Enabled	False
Numeric Round-Abort	False
Parameterization	Simple

ANSI NULL Default

Server:
SQ002601

Connection:
MDC\adminmega

3. Click **OK**.

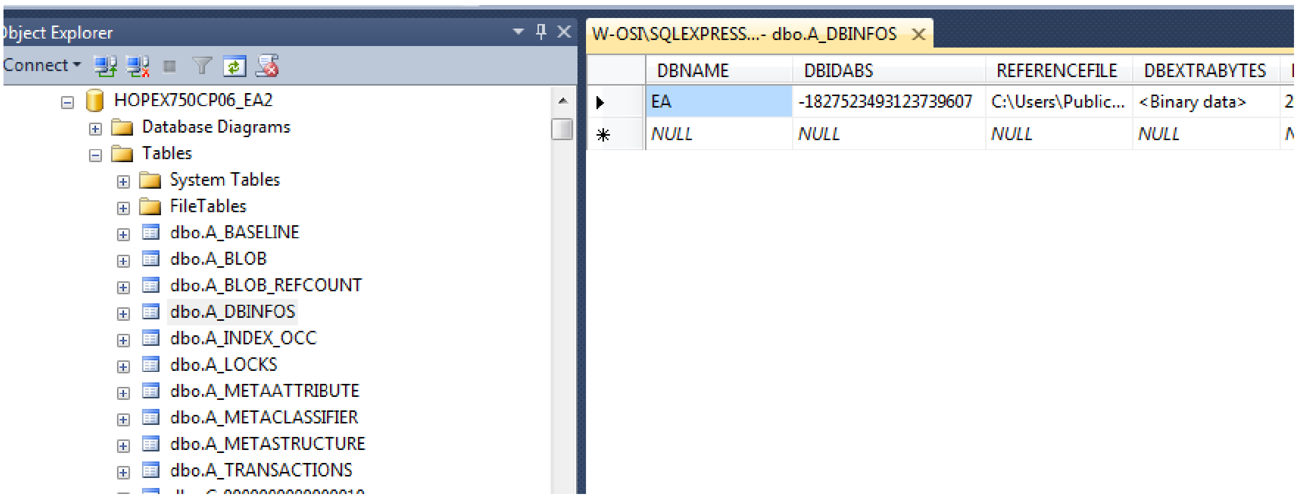
When duplicating a repository in an environment – Expert mode

This section is only if you are very confident about your HOPEX and RDBMS skills, since you have to modify data directly in some tables of your database.

In case you want, within an environment, duplicate a specific repository, you need to tweak the data manually in two tables of your restored database.

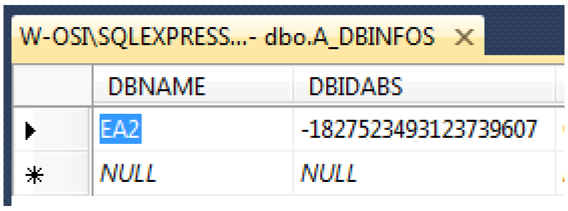
For example: you want to duplicate repository “EA” to have a repository “EA2” in the environment “HOPEX750CP06”.

1. You have restored database EA twice, in databases “HOPEX750CP06_EA” and “HOPEX750CP06_EA2”.
2. Using SQL Server Management Studio, open database “HOPEX750CP06_EA2”, and edit the table “dbo.A_DBINFOS”:



DBNAME	DBIDABS	REFERENCEFILE	DBEXTRABYTES
EA	-1827523493123739607	C:\Users\Public...	<Binary data>
NULL	NULL	NULL	NULL

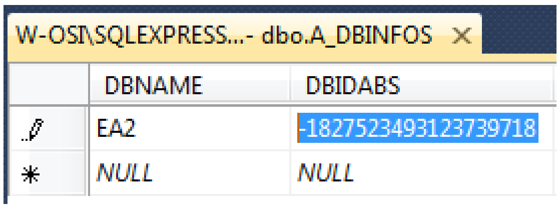
3. Modify the **DBNAME** field of that line, to enter the name of your duplicate, i.e. how you want it to appear in your environment (for example: here, “EA2”).



DBNAME	DBIDABS
EA2	-1827523493123739607
NULL	NULL

4. Modify the **DBIDABS** field that is the unique identifier of the repository within your environment.

   You need to change its value, and make sure that it is unique for all other repositories, so you need to check in all schemas. Make sure that when you change one or two characters, it will create a string that is not used by any other (for example: here, we modified the last 3 digits from “ 607” to “718”).



DBNAME	DBIDABS
EA2	-1827523493123739718
NULL	NULL

5. Save your updates and close the table.

POST INSTALLATION TASKS

This section describes the tasks you need to carry out after duplicating the HOPEX data that were stored in the SQL Server database.

The first part details what to do to reattach to the set of duplicated data from HOPEX point of view.

As all is not stored inside the SQL Server, you need to copy from the original HOPEX environment to the new HOPEX environment pointing at the duplicate SQL Server data.

Create/Attach an environment in HOPEX

Creating an environment

To create an environment:

1. Connect to **HOPEX Administration**.
2. In the navigation tree, right-click **Environments** and select **New**.
3. Enter the **Name** of the new environment.

   The environment name must respect Windows folder naming constraints.

   It is the beginning of the name of the system database. It should be called <environment>_SystemDb, where <environment is the string that you enter in the HOPEX admin tool. In this example, the environment is called "GGS".
4. (Optional) Location of the environment is specified by default; you can modify it if necessary using the **Browse** button.

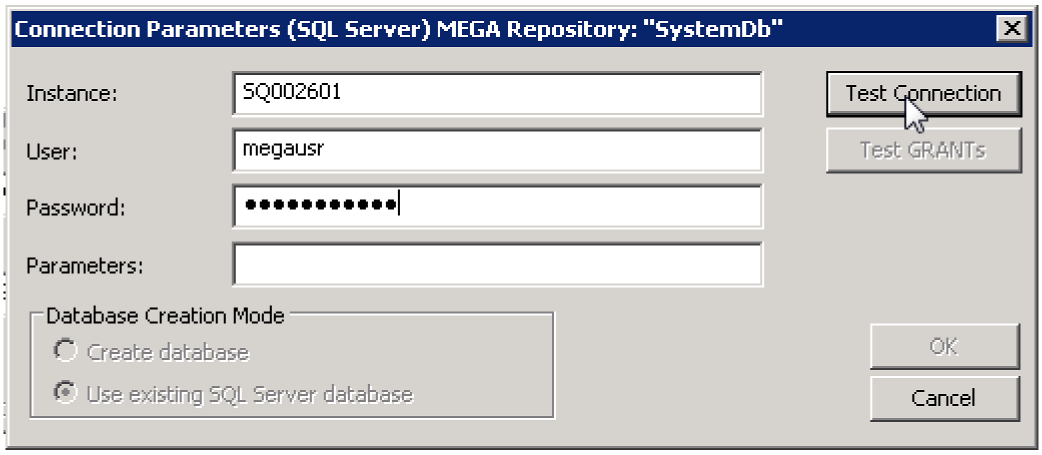
   In this example, we created a share on the server hosting the SQL Server instance, that will host the environment files, and that is called \\sq002601\EnvironnementsMega.

   The RDBMS repository server type for the new environment is already selected: SQL Server.
5. Select **Restore** which allows to connect to an existing SQL Server database on an instance.
6. Click **OK**.
7. Enter the connection parameters for the **Instance**.

   Do not enter anything in the **Parameters** field.

   Note:

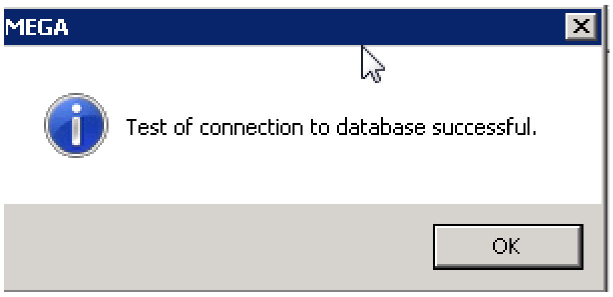
   The syntax for a named instance on **SQL Server** is: server_name\instance_name. In this example, the instance is the default one, without a specific name. That is why we only provide the name of the server where the instance is running.


Connection Parameters (SQL Server) MEGA Repository: "SystemDb"

Instance: SQ002601
User: megausr
Password: ●●●●●●●●●●●●
Parameters:

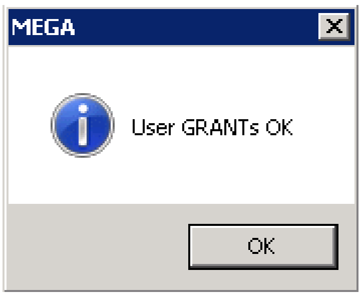
Database Creation Mode
( ) Create database
(●) Use existing SQL Server database

8. Click **Test Connection**.


MEGA

Test of connection to database successful.

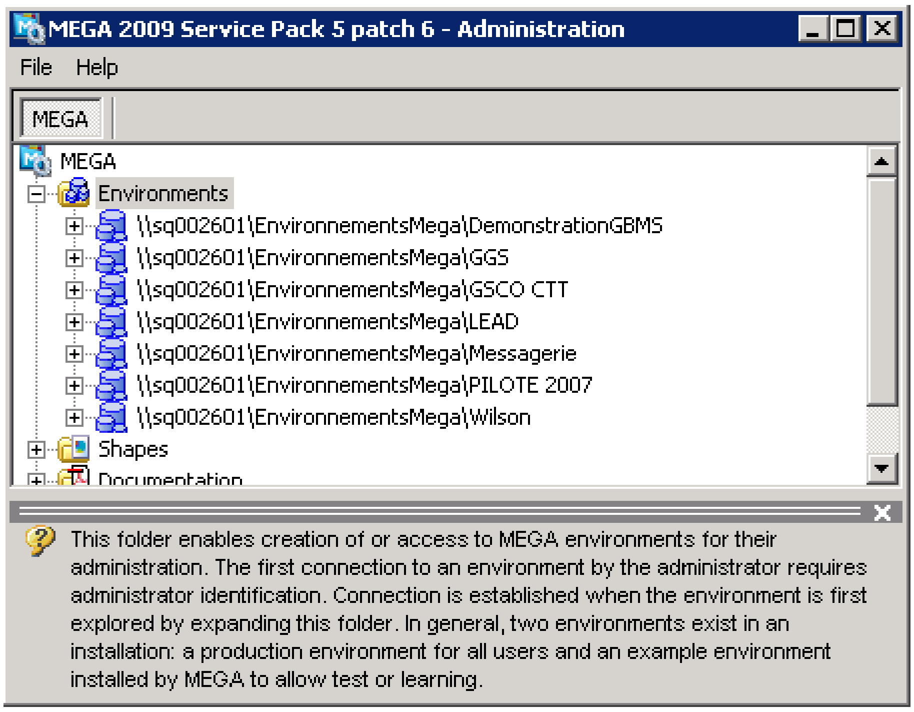
9. Click **Test GRANTS**.


MEGA

User GRANTs OK

10. Click **OK**.

    A popup indicating that the environment was successfully created appears, and the new environment is displayed in the list (for example here « \\sql002601\EnvironmentsMega\GGS).


MEGA 2009 Service Pack 5 patch 6 - Administration
File
Help
MEGA
MEGA
Environments
\\sq002601\EnvironnementsMega\DemonstrationGBMS
\\sq002601\EnvironnementsMega\GGS
\\sq002601\EnvironnementsMega\GSCO CTT
\\sq002601\EnvironnementsMega\LEAD
\\sq002601\EnvironnementsMega\Messagerie
\\sq002601\EnvironnementsMega\PILOTE 2007
\\sq002601\EnvironnementsMega\Wilson
Shapes
Documentation

This folder enables creation of or access to MEGA environments for their administration. The first connection to an environment by the administrator requires administrator identification. Connection is established when the environment is first explored by expanding this folder. In general, two environments exist in an installation: a production environment for all users and an example environment installed by MEGA to allow test or learning.

Attaching the working database(s) to the environment

Once your environment is created, you still have to attach the working database(s) to this environment.

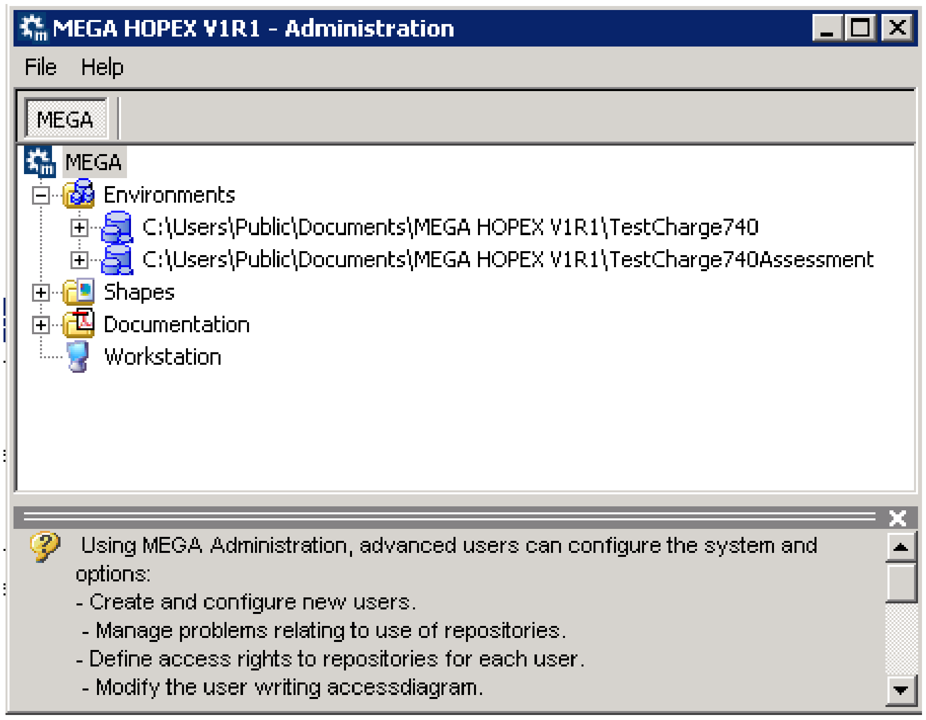
In **Repositories** you only have the SystemDb.

Now you have to connect to the environment, and add the HOPEX repositories. The first time you connect, you receive warning message(s) telling you that database "X" ("X" being the name of the repository that existed in the source environment, and most likely one of the repositories you are trying to attach) is not referenced.

This is because the SystemDb database contains some information about the working repositories. You can discard this warning, and continue.

To attach a database, previously restored in SQL Server Management Studio:

1. Connect to **HOPEX Administration**.


MEGA HOPEX V1R1 - Administration

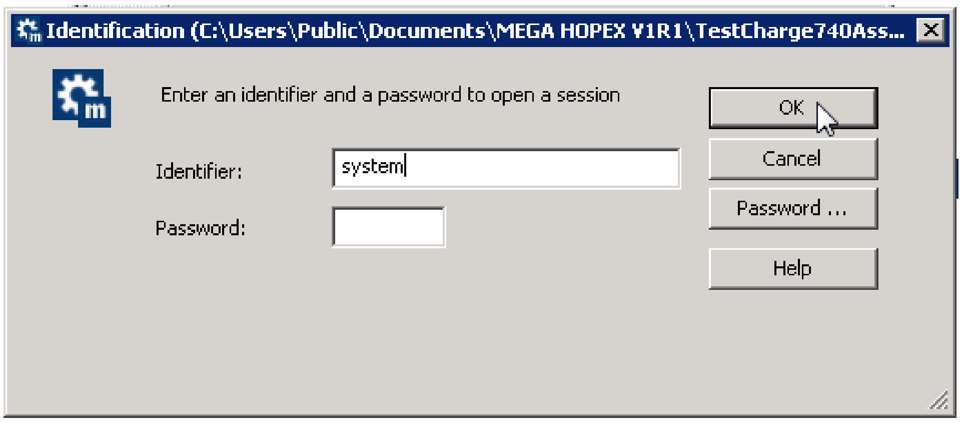
File Help

MEGA

- MEGA
  - Environments
    - C:\Users\Public\Documents\MEGA HOPEX V1R1\TestCharge740
    - C:\Users\Public\Documents\MEGA HOPEX V1R1\TestCharge740Assessment
  - Shapes
  - Documentation
  - Workstation

Using MEGA Administration, advanced users can configure the system and options:
- Create and configure new users.
- Manage problems relating to use of repositories.
- Define access rights to repositories for each user.
- Modify the user writing accessdiagram.

2. Connect to the wanted environment (for example "TestEnv", with **System** user.

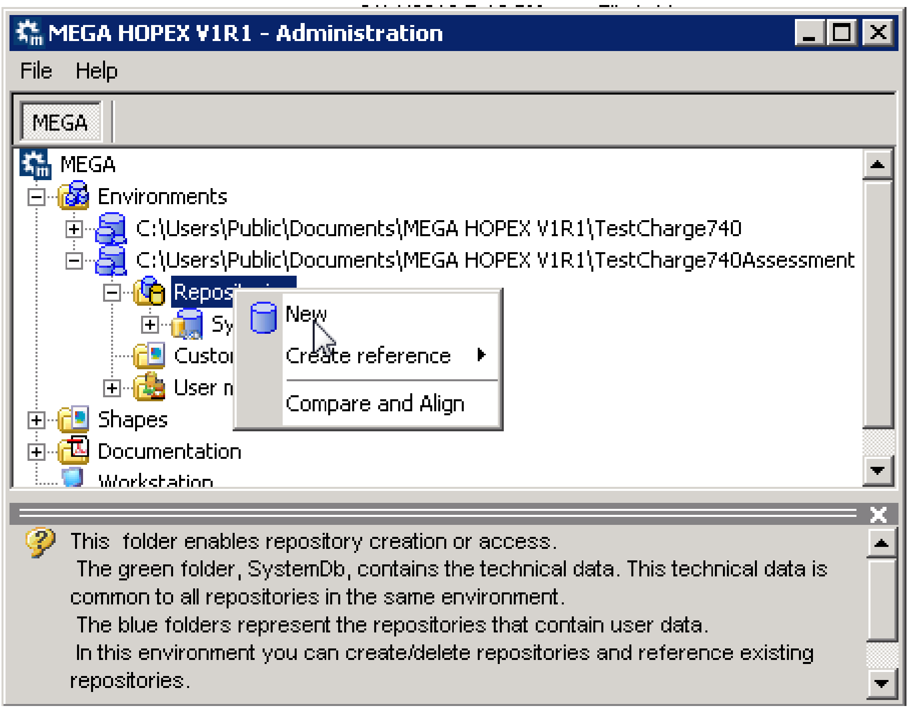

Identification (C:\Users\Public\Documents\MEGA HOPEX V1R1\TestCharge740Ass...

Enter an identifier and a password to open a session

Identifier: system

Password:

3. Right-click **Repositories** and select **New**.


MEGA HOPEX V1R1 - Administration

File Help

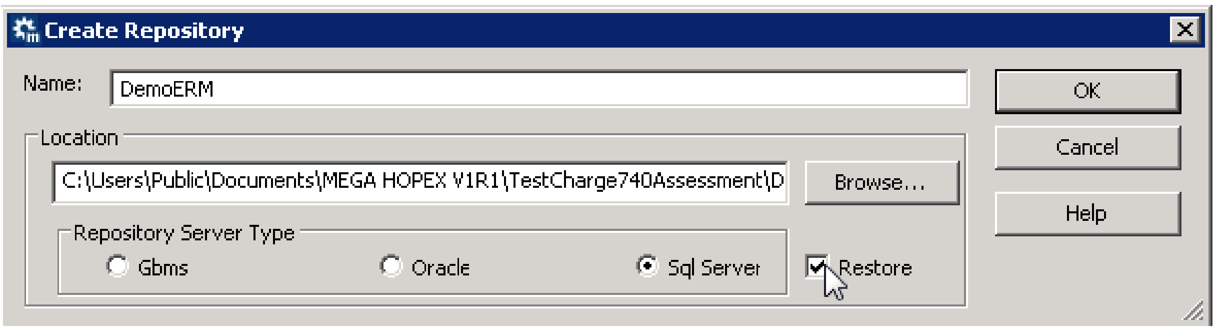
MEGA

- MEGA
  - Environments
    - C:\Users\Public\Documents\MEGA HOPEX V1R1\TestCharge740
    - C:\Users\Public\Documents\MEGA HOPEX V1R1\TestCharge740Assessment
      - Reposi
        - Sy
      - Custo
      - User n
  - Shapes
  - Documentation
  - Workstation

This folder enables repository creation or access.
The green folder, SystemDb, contains the technical data. This technical data is common to all repositories in the same environment.
The blue folders represent the repositories that contain user data.
In this environment you can create/delete repositories and reference existing repositories.

4. Enter the Name of the Repository (in this example "DemoERM")

   In **SQL Server**, you should have an existing database called "TestEnv_ DemoERM", with the native SQL Server user db_owner of the database.
5. In the **Repository Server Type** pane, select **SQL Server**.
6. Select **Restore**.


Create Repository

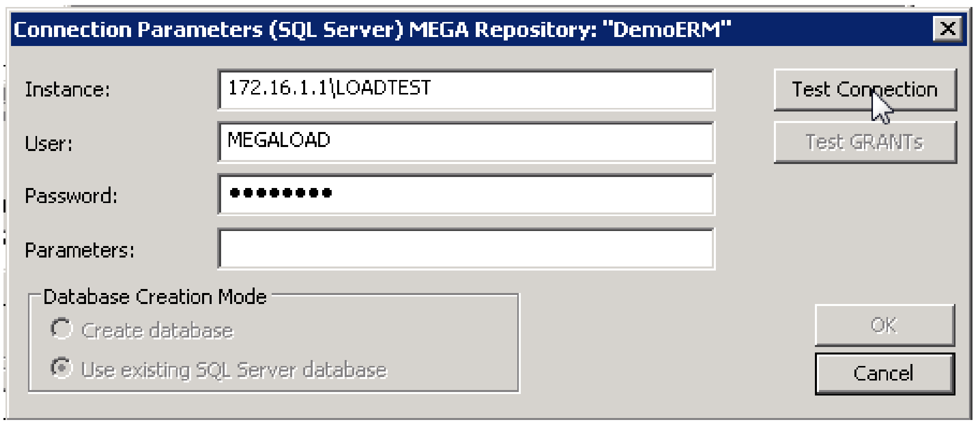
Name: DemoERM

Location: C:\Users\Public\Documents\MEGA HOPEX V1R1\TestCharge740Assessment\D

Repository Server Type: ( ) Gbms ( ) Oracle (●) Sql Server

[✓] Restore

7. Click **OK**.
8. The connection parameters are already set. Check that they are correct.


Connection Parameters (SQL Server) MEGA Repository: "DemoERM"

Instance: 172.16.1.1\LOADTEST

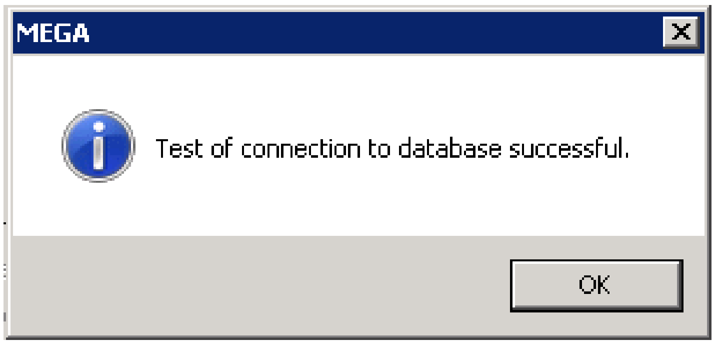
User: MEGALOAD

Password: ••••••••

Parameters:

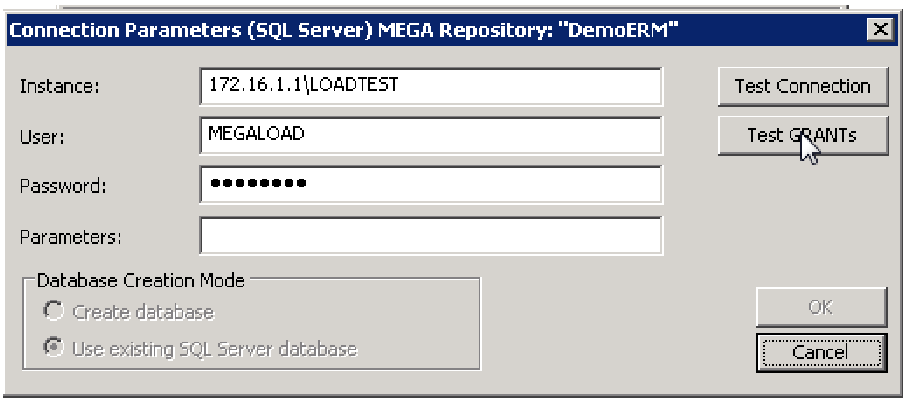
Database Creation Mode: ( ) Create database (●) Use existing SQL Server database

9. Click **Test Connection**.

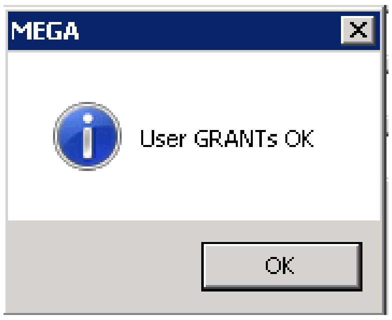


Test of connection to database successful.

10. Click **Test GRANTs**.

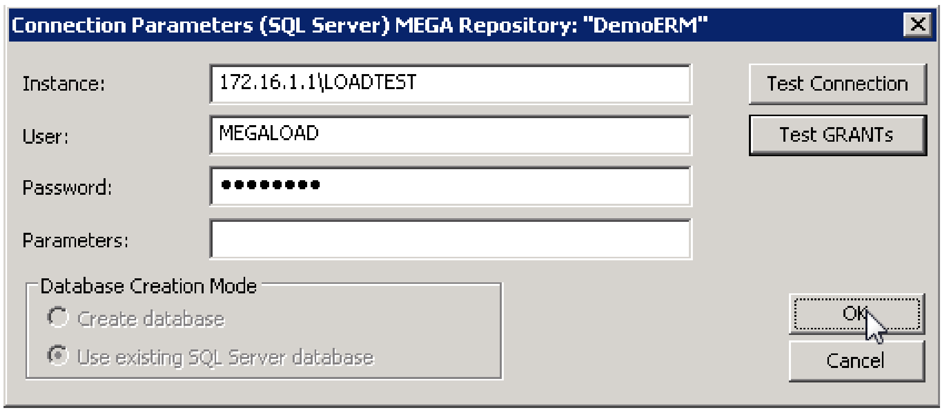

Connection Parameters (SQL Server) MEGA Repository: "DemoERM"
Test Connection
Test GRANTs
OK
Cancel

Instance: 172.16.1.1\LOADTEST
User: MEGALOAD
Password: •••••••••
Parameters:
Database Creation Mode
Create database
[✓ Use existing SQL Server database]

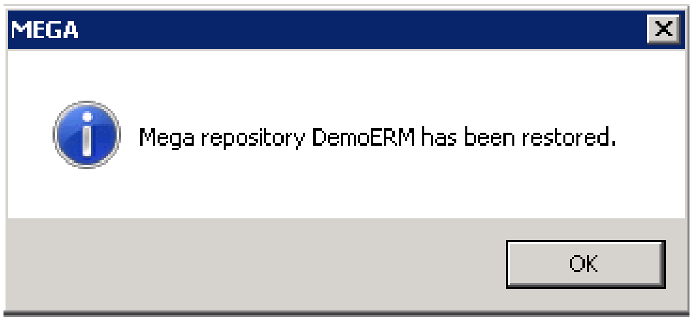


User GRANTs OK

11. Click **OK** to create the repository.

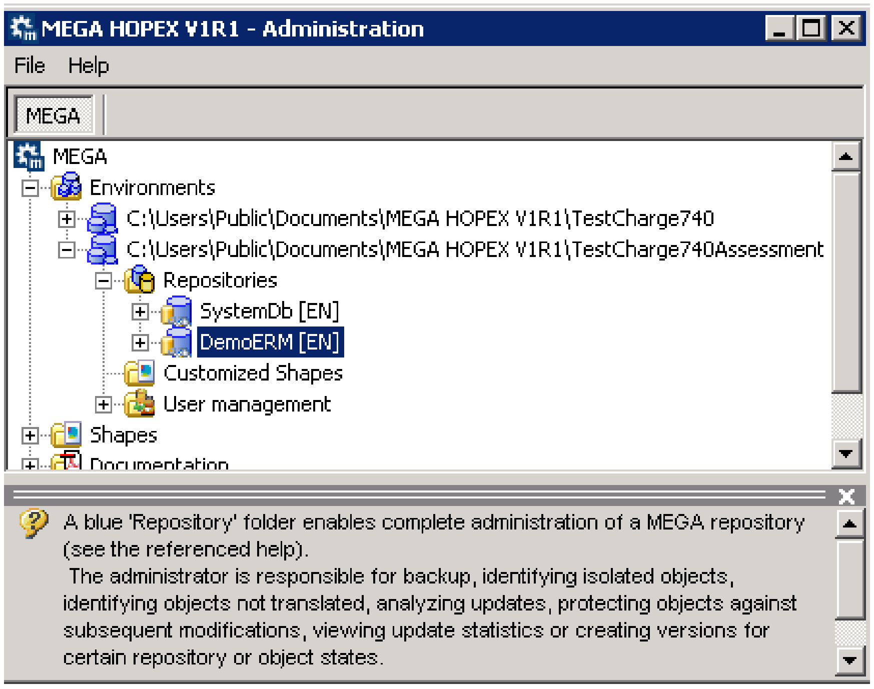

Connection Parameters (SQL Server) MEGA Repository: "DemoERM"
Test Connection
Test GRANTs
OK
Cancel

Instance: 172.16.1.1\LOADTEST
User: MEGALOAD
Password: •••••••••
Parameters:
Database Creation Mode
Create database
[✓ Use existing SQL Server database]


MEGA

Mega repository DemoERM has been restored.

The “DemoERM” repository appears in the **Repositories** list.


MEGA HOPEX V1R1 - Administration
File
Help
MEGA
MEGA
Environments
C:\Users\Public\Documents\MEGA HOPEX V1R1\TestCharge740
C:\Users\Public\Documents\MEGA HOPEX V1R1\TestCharge740Assessment
Repositories
SystemDb [EN]
DemoERM [EN]
Customized Shapes
User management
Shapes
Documentation

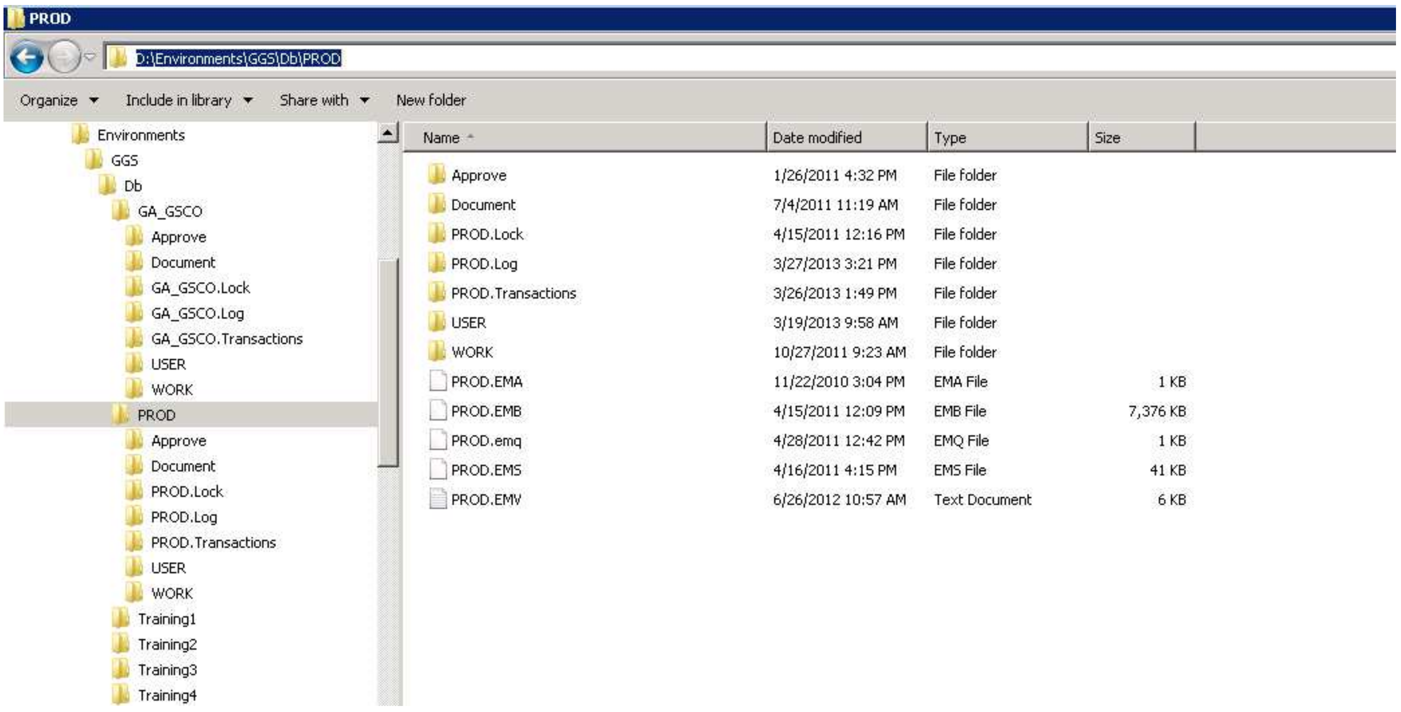
A blue 'Repository' folder enables complete administration of a MEGA repository (see the referenced help).
The administrator is responsible for backup, identifying isolated objects, identifying objects not translated, analyzing updates, protecting objects against subsequent modifications, viewing update statistics or creating versions for certain repository or object states.

Copy the documents from source to target

You have to copy the documents from source to target for each repository.

The Word/RTF documents

1. Connect to the source server.
2. Go to the folder hosting the environment (for example environment “GSCO CTT”).
3. Go to the **Db** folder, and in the sub-folder of each repository you migrated (in this example « D:\Environments\GGS\Db\PROD » contains the data of the repository called “CTFTM_OLD”).



PROD

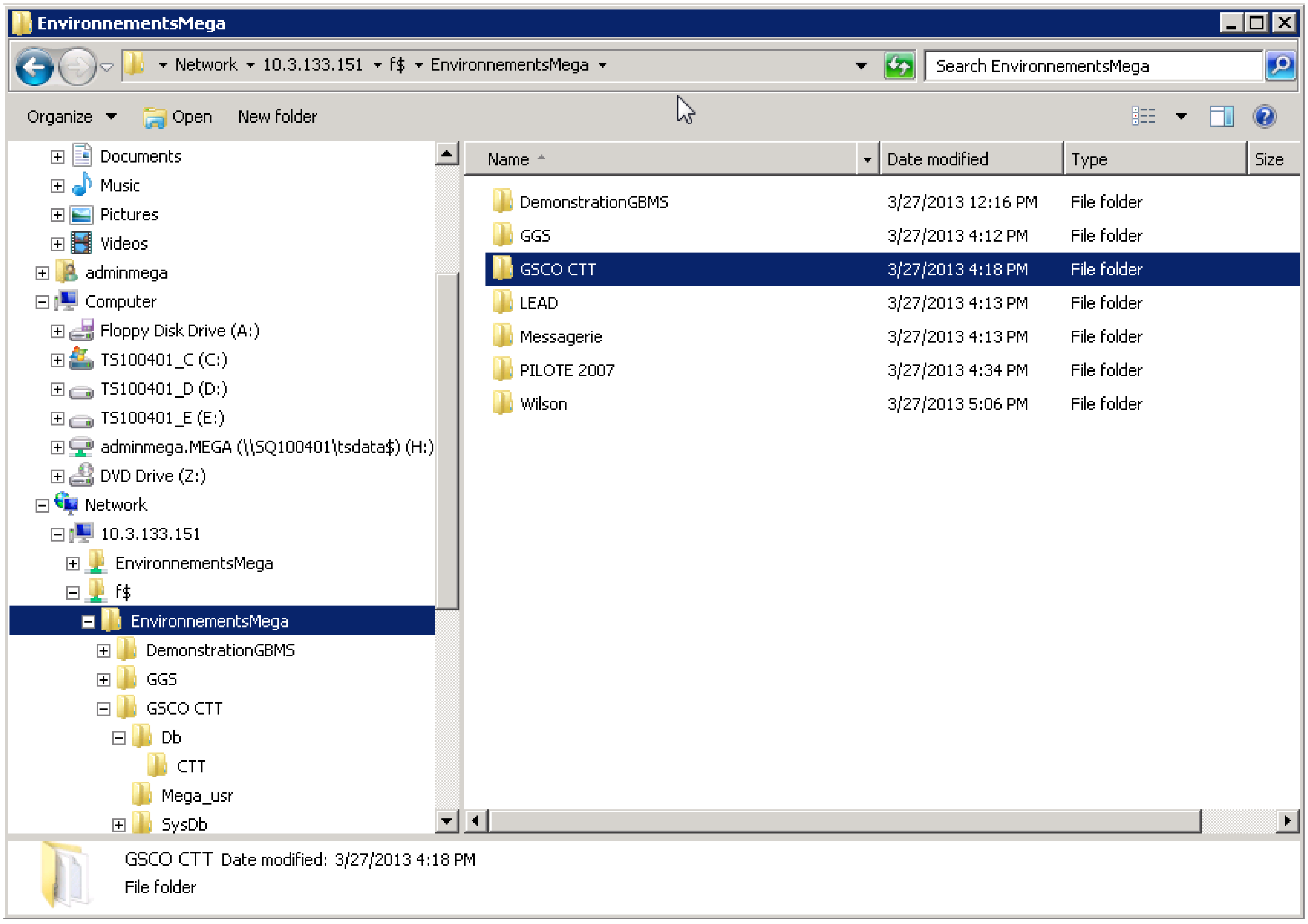
D:\Environments\GGS\Db\PROD

Name	Date modified	Type	Size
Approve	1/26/2011 4:32 PM	File folder	
Document	7/4/2011 11:19 AM	File folder	
PROD.Lock	4/15/2011 12:16 PM	File folder	
PROD.Log	3/27/2013 3:21 PM	File folder	
PROD.Transactions	3/26/2013 1:49 PM	File folder	
USER	3/19/2013 9:58 AM	File folder	
WORK	10/27/2011 9:23 AM	File folder	
PROD.EMA	11/22/2010 3:04 PM	EMA File	1 KB
PROD.EMB	4/15/2011 12:09 PM	EMB File	7,376 KB
PROD.emq	4/28/2011 12:42 PM	EMQ File	1 KB
PROD.EMS	4/16/2011 4:15 PM	EMS File	41 KB
PROD.EMV	6/26/2012 10:57 AM	Text Document	6 KB

4. From the source server, open an explorer on the target server, in the same folder.

   In this example, the environments on the target server are hosted on the server with 10.3.133.151 IP address, and on that server the environments are located on the F drive in the “EnvironnementsMega” folder, so that the syntax is:

   \\10.3.133.151\f$\EnvironnementsMega\



EnvironnementsMega

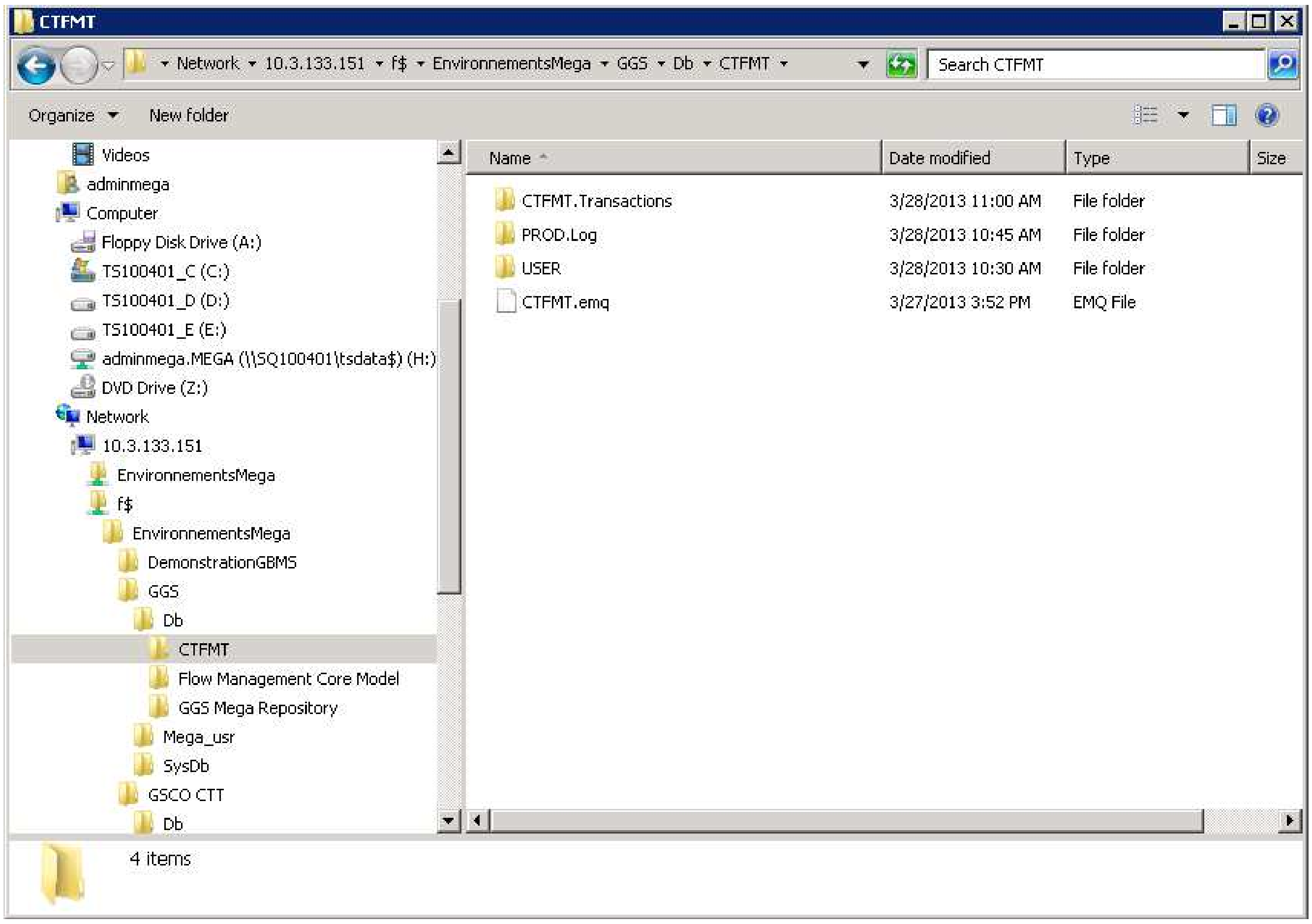
Network ▸ 10.3.133.151 ▸ f$ ▸ EnvironnementsMega

Name	Date modified	Type	Size
DemonstrationGBMS	3/27/2013 12:16 PM	File folder	
GGS	3/27/2013 4:12 PM	File folder	
GSCO CTT	3/27/2013 4:18 PM	File folder	
LEAD	3/27/2013 4:13 PM	File folder	
Messagerie	3/27/2013 4:13 PM	File folder	
PILOTE 2007	3/27/2013 4:34 PM	File folder	
Wilson	3/27/2013 5:06 PM	File folder	

GSCO CTT Date modified: 3/27/2013 4:18 PM
File folder

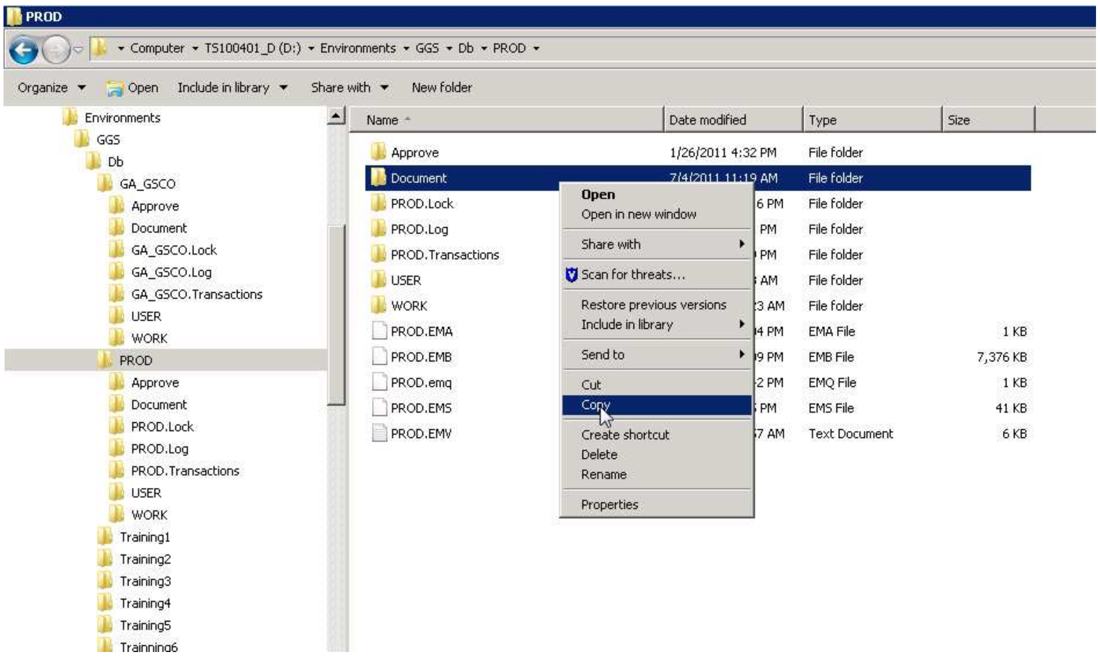
5. Go to the same sub-folder « …\GGS\Db\CTFMT ».

   Note: If you are wondering why it is not the same environment name and the same repository name, that is because in this example, this repository was taken from one environment to another, with a rename.



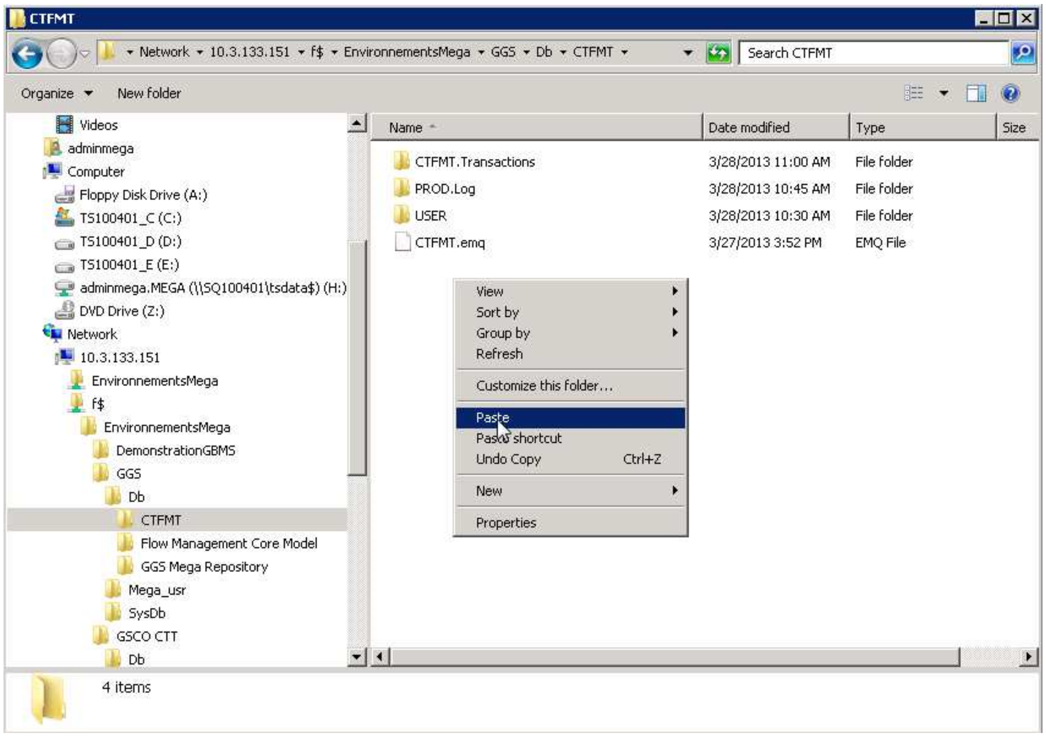
Name	Date modified	Type	Size
CTFMT.Transactions	3/28/2013 11:00 AM	File folder	
PROD.Log	3/28/2013 10:45 AM	File folder	
USER	3/28/2013 10:30 AM	File folder	
CTFMT.emq	3/27/2013 3:52 PM	EMQ File	

6. Right-click the **Document** folder from the source and select **Copy**.



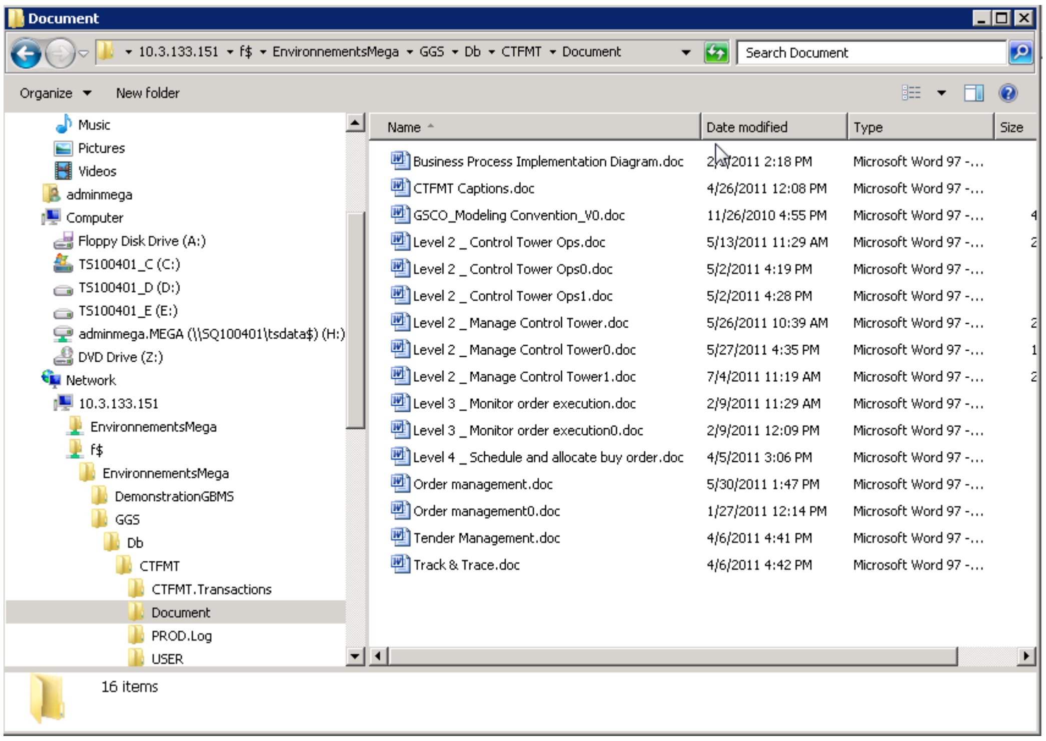
Name	Date modified	Type	Size
Approve	1/26/2011 4:32 PM	File folder	
Document	7/4/2011 11:19 AM	File folder	
PROD.Lock	6 PM	File folder	
PROD.Log	PM	File folder	
PROD.Transactions	PM	File folder	
USER	AM	File folder	
WORK	3 AM	File folder	
PROD.EMA	4 PM	EMA File	1 KB
PROD.EMB	9 PM	EMB File	7,376 KB
PROD.emq	2 PM	EMQ File	1 KB
PROD.EMS	PM	EMS File	41 KB
PROD.EMV	7 AM	Text Document	6 KB

7. Paste it on the target (future Production) :



Name	Date modified	Type	Size
CTFMT.Transactions	3/28/2013 11:00 AM	File folder	
PROD.Log	3/28/2013 10:45 AM	File folder	
USER	3/28/2013 10:30 AM	File folder	
CTFMT.emq	3/27/2013 3:52 PM	EMQ File	

8. Check that the documents are all in the target folder.



Name	Date modified	Type
Business Process Implementation Diagram.doc	2/4/2011 2:18 PM	Microsoft Word 97 -...
CTFMT Captions.doc	4/26/2011 12:08 PM	Microsoft Word 97 -...
GSCO_Modeling Convention_V0.doc	11/26/2010 4:55 PM	Microsoft Word 97 -...
Level 2 _ Control Tower Ops.doc	5/13/2011 11:29 AM	Microsoft Word 97 -...
Level 2 _ Control Tower Ops0.doc	5/2/2011 4:19 PM	Microsoft Word 97 -...
Level 2 _ Control Tower Ops1.doc	5/2/2011 4:28 PM	Microsoft Word 97 -...
Level 2 _ Manage Control Tower.doc	5/26/2011 10:39 AM	Microsoft Word 97 -...
Level 2 _ Manage Control Tower0.doc	5/27/2011 4:35 PM	Microsoft Word 97 -...
Level 2 _ Manage Control Tower1.doc	7/4/2011 11:19 AM	Microsoft Word 97 -...
Level 3 _ Monitor order execution.doc	2/9/2011 11:29 AM	Microsoft Word 97 -...
Level 3 _ Monitor order execution0.doc	2/9/2011 12:09 PM	Microsoft Word 97 -...
Level 4 _ Schedule and allocate buy order.doc	4/5/2011 3:06 PM	Microsoft Word 97 -...
Order management.doc	5/30/2011 1:47 PM	Microsoft Word 97 -...
Order management0.doc	1/27/2011 12:14 PM	Microsoft Word 97 -...
Tender Management.doc	4/6/2011 4:41 PM	Microsoft Word 97 -...
Track & Trace.doc	4/6/2011 4:42 PM	Microsoft Word 97 -...

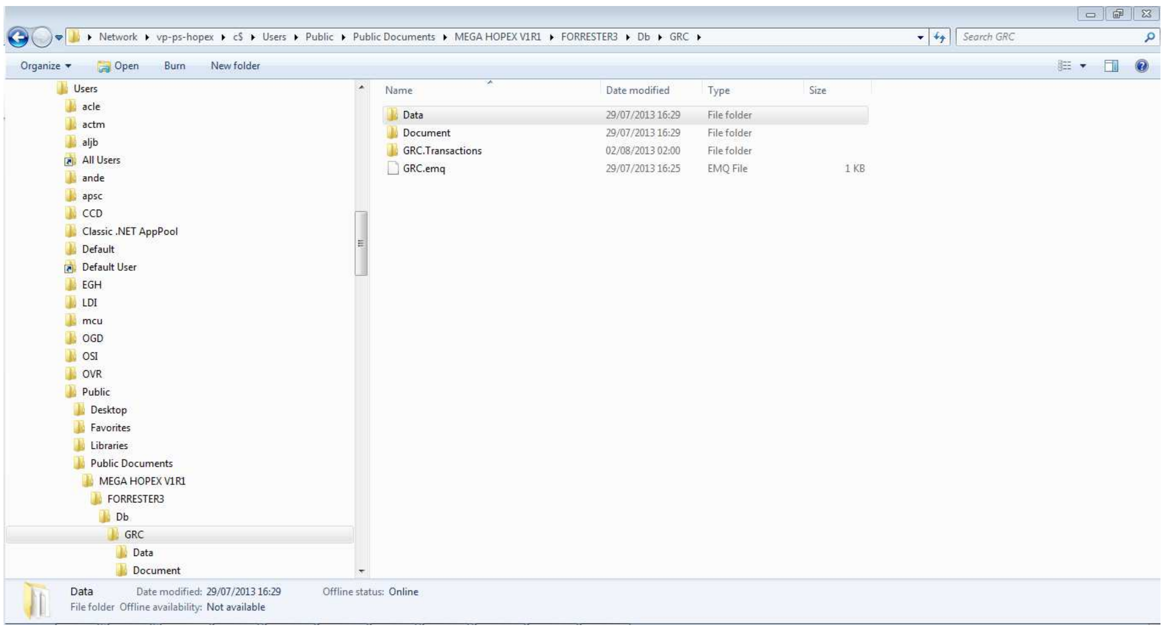
The internal documents (.DAT files)

Some documents can also be stored partially inside the SQL Server databases (as references) and partially in .DAT files (generated in the folder of each repository of the migrated environment).

By restoring the SQL Server databases, you already retrieved the references. Now you have to copy a folder from source to target.

1. From the source environment, expand the **Db** folder and the sub-folder of the database.

In this example, on server vp-ps-hopex, we are looking at environment py aRESTER3 from source to target.the migrated environment).ry name, that is because in tX V1R1\FORRESTER3\Db\GRCxa


Network ▸ vp-ps-hopex ▸ c$ ▸ Users ▸ Public ▸ Public Documents ▸ MEGA HOPEX V1R1 ▸ FORRESTER3 ▸ Db ▸ GRC ▸

Search GRC

Organize ▾ Open Burn New folder

Name	Date modified	Type	Size
Data	29/07/2013 16:29	File folder	
Document	29/07/2013 16:29	File folder	
GRC.Transactions	02/08/2013 02:00	File folder	
GRC.emq	29/07/2013 16:25	EMQ File	1 KB

```
Users
  acle
  actm
  aljb
  All Users
  ande
  apsc
  CCD
  Classic .NET AppPool
  Default
  Default User
  EGH
  LDI
  mcu
  OGD
  OSI
  OVR
  Public
    Desktop
    Favorites
    Libraries
    Public Documents
      MEGA HOPEX V1R1
        FORRESTER3
          Db
            GRC
              Data
              Document
```

Data Date modified: 29/07/2013 16:29 Offline status: Online
File folder Offline availability: Not available

2. Right-click the **Data** folder and select **Copy**.
3. Paste it at the same level on the target folder.

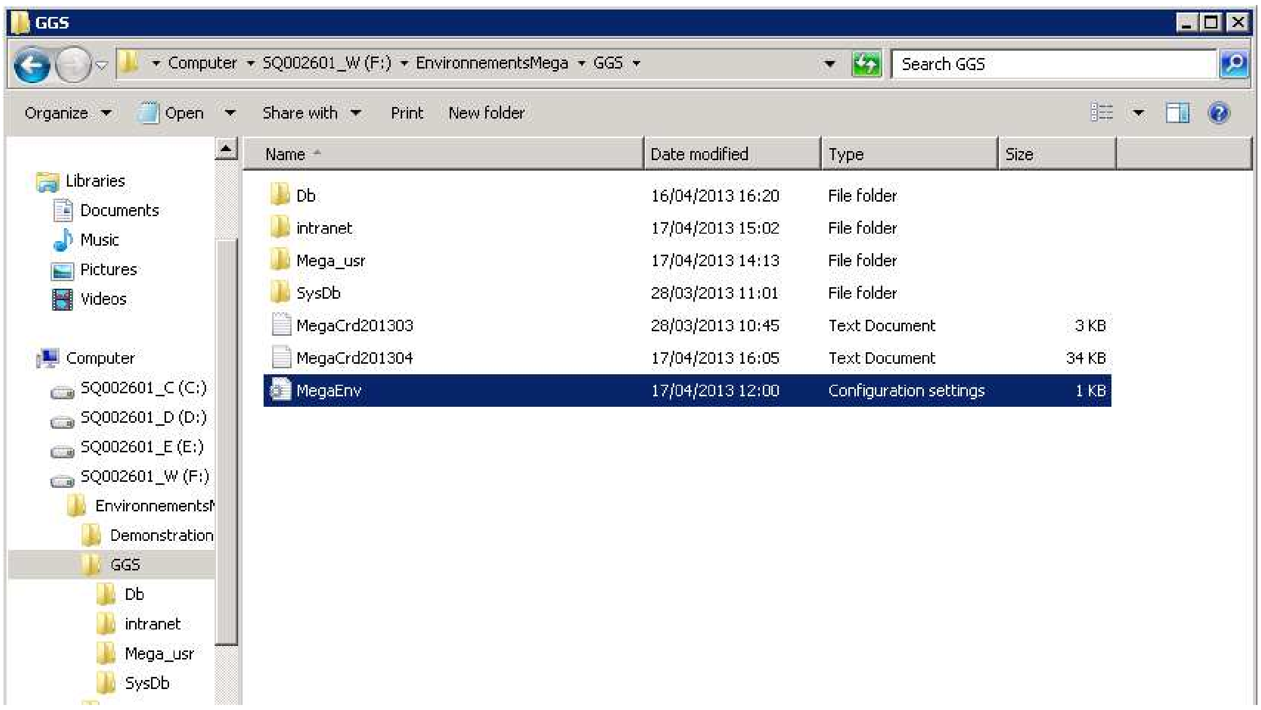
Get the parameters of the environment

For each environment, you need to retrieve certain types of configuration included in:

- the **MegaEnv.ini** file

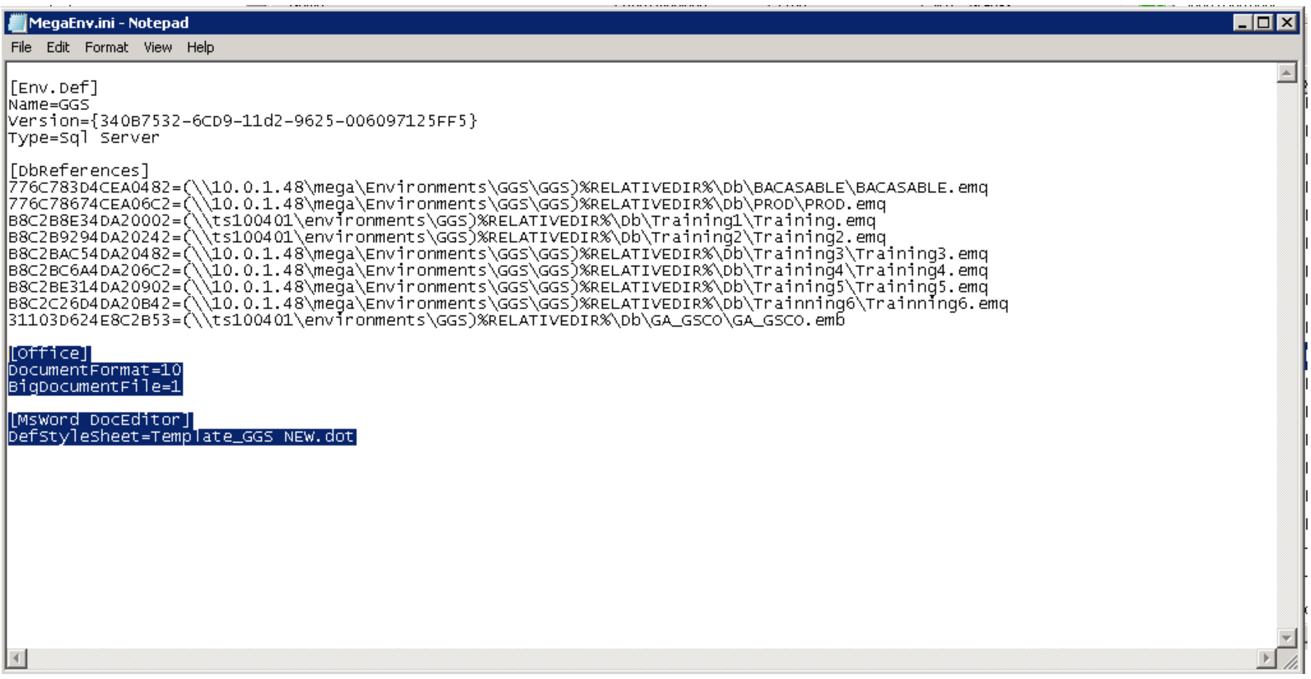
MegaEnv.ini file

The MegaEnv.ini file is located at the root level of each environment (in this example, in the GGS environment folder) on the source server.



Name	Date modified	Type	Size
Db	16/04/2013 16:20	File folder	
intranet	17/04/2013 15:02	File folder	
Mega_usr	17/04/2013 14:13	File folder	
SysDb	28/03/2013 11:01	File folder	
MegaCrd201303	28/03/2013 10:45	Text Document	3 KB
MegaCrd201304	17/04/2013 16:05	Text Document	34 KB
MegaEnv	17/04/2013 12:00	Configuration settings	1 KB

1. Open the **MegaEnv.ini** file on the source server.

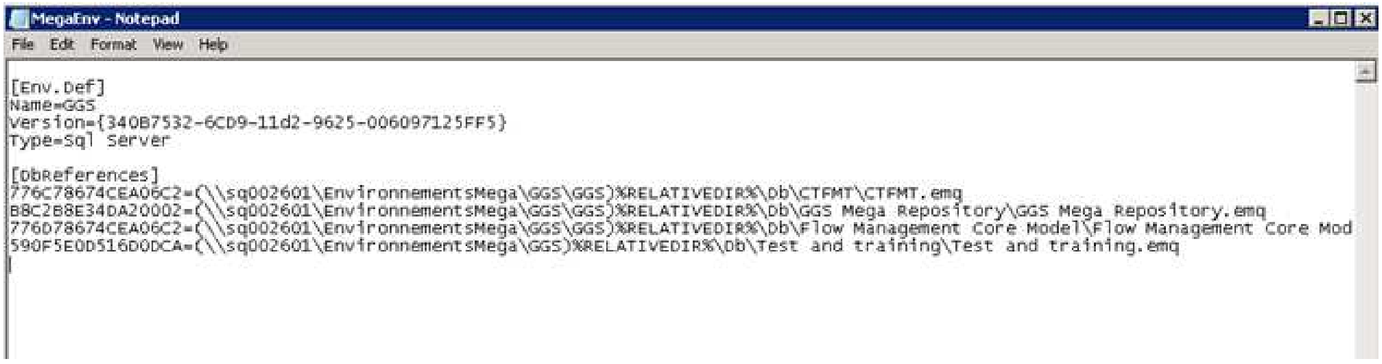


```
[Env.Def]
Name=GGS
Version={340B7532-6CD9-11d2-9625-006097125FF5}
Type=Sql Server

[DbReferences]
776C783D4CEA0482=(\\10.0.1.48\mega\Environments\GGS\GGS)%RELATIVEDIR%\Db\BACASABLE\BACASABLE.emq
776C78674CEA06C2=(\\10.0.1.48\mega\Environments\GGS\GGS)%RELATIVEDIR%\Db\PROD\PROD.emq
B8C2B8E34DA20002=(\\ts100401\environments\GGS)%RELATIVEDIR%\Db\Training1\Training.emq
B8C2B9294DA20242=(\\ts100401\environments\GGS)%RELATIVEDIR%\Db\Training2\Training2.emq
B8C2BAC54DA20482=(\\10.0.1.48\mega\Environments\GGS\GGS)%RELATIVEDIR%\Db\Training3\Training3.emq
B8C2BC6A4DA206C2=(\\10.0.1.48\mega\Environments\GGS\GGS)%RELATIVEDIR%\Db\Training4\Training4.emq
B8C2BE314DA20902=(\\10.0.1.48\mega\Environments\GGS\GGS)%RELATIVEDIR%\Db\Training5\Training5.emq
B8C2C26D4DA20B42=(\\10.0.1.48\mega\Environments\GGS\GGS)%RELATIVEDIR%\Db\Trainning6\Trainning6.emq
31103D624E8C2B53=(\\ts100401\environments\GGS)%RELATIVEDIR%\Db\GA_GSCO\GA_GSCO.emb

[Office]
DocumentFormat=10
BigDocumentFile=1

[MsWord DocEditor]
DefStyleSheet=Template_GGS NEW.dot
```

2. Below the [DbReferences] section, you find the environment-specific parameters, like the type of document (DocumentFormat=10 meaning that they were converted from .DOC to .RTF) or the templates used by default.

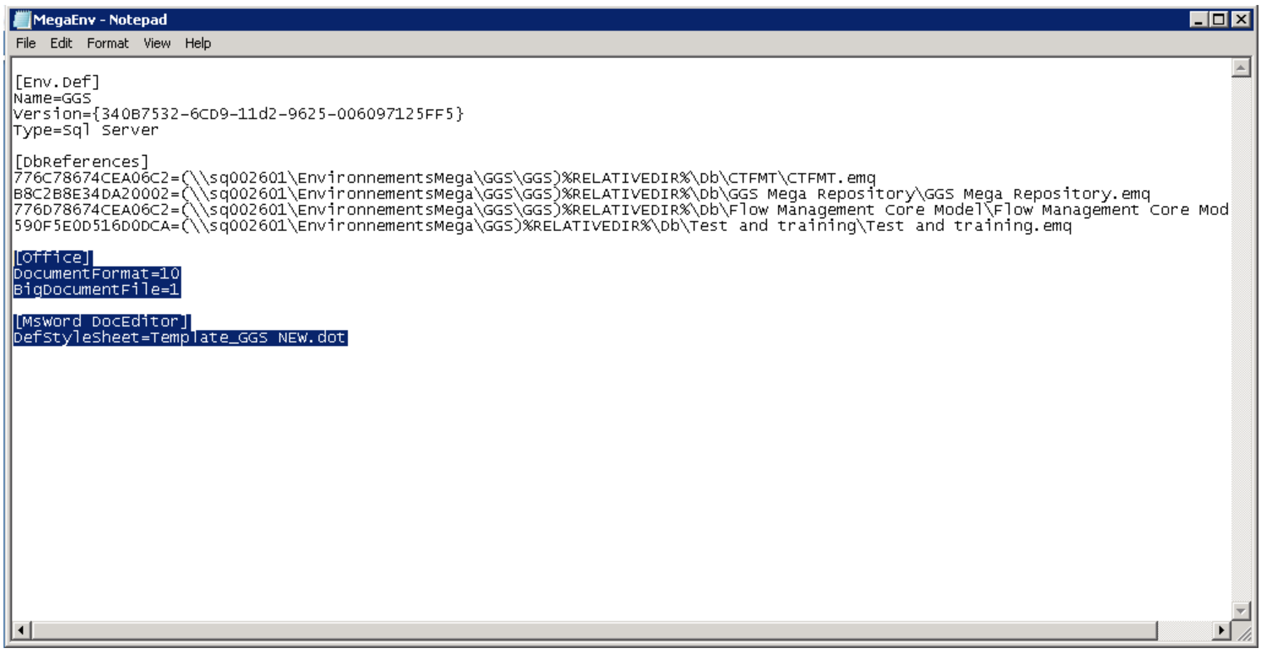
Copy this specific section.

3. Open the **MegaEnv.ini** of the target server.



```
[Env.Def]
Name=GGS
Version={340B7532-6CD9-11d2-9625-006097125FF5}
Type=Sql Server

[DbReferences]
776C78674CEA06C2=(\\sq002601\EnvironnementsMega\GGS\GGS)%RELATIVEDIR%\Db\CTFMT\CTFMT.emq
B8C2B8E34DA20002=(\\sq002601\EnvironnementsMega\GGS\GGS)%RELATIVEDIR%\Db\GGS Mega Repository\GGS Mega Repository.emq
776D78674CEA06C2=(\\sq002601\EnvironnementsMega\GGS\GGS)%RELATIVEDIR%\Db\Flow Management Core Model\Flow Management Core Mod
590F5E0D516D0DCA=(\\sq002601\EnvironnementsMega\GGS)%RELATIVEDIR%\Db\Test and training\Test and training.emq
```

4. Paste this section at the same level in the file and save the document.



```
[Env.Def]
Name=GGS
Version={340B7532-6CD9-11d2-9625-006097125FF5}
Type=Sql Server

[DbReferences]
776C78674CEA06C2=(\\sq002601\EnvironnementsMega\GGS\GGS)%RELATIVEDIR%\Db\CTFMT\CTFMT.emq
B8C2B8E34DA20002=(\\sq002601\EnvironnementsMega\GGS\GGS)%RELATIVEDIR%\Db\GGS Mega Repository\GGS Mega Repository.emq
776D78674CEA06C2=(\\sq002601\EnvironnementsMega\GGS\GGS)%RELATIVEDIR%\Db\Flow Management Core Model\Flow Management Core Mod
590F5E0D516D0DCA=(\\sq002601\EnvironnementsMega\GGS)%RELATIVEDIR%\Db\Test and training\Test and training.emq

[Office]
DocumentFormat=10
BigDocumentFile=1

[MsWord DocEditor]
DefStyleSheet=Template_GGS NEW.dot
```

AUTOMATIC TRANSLATION ADMINISTRATION

The automatic translation proposed by **HOPEX** is based on Microsoft Translator Text API.

There are two ways of using automatic translation of the data created by the users:

- interactively managed by the users, object by object.
- per batch processing by an Administrator.

This section is dedicated to the administration work of the batch processing.

PREREQUISITES FOR THE USE OF AUTOMATIC TRANSLATION

For the commands linked to the automatic translation of data to be available, you must have acquired a **Microsoft Translator Text API** license.

☛ *For further details about this license, see* https://azure.microsoft.com/en-us/services/cognitive-services/translator-text-api/.

To use this service, your IT architecture must allow the HOPEX servers to access the URL https://api.cognitive.microsoft.com/.

The API key of your **Microsoft Translator Text API** license must be declared and, if your API key is region-dependent, the region to which this key belongs must also be declared.

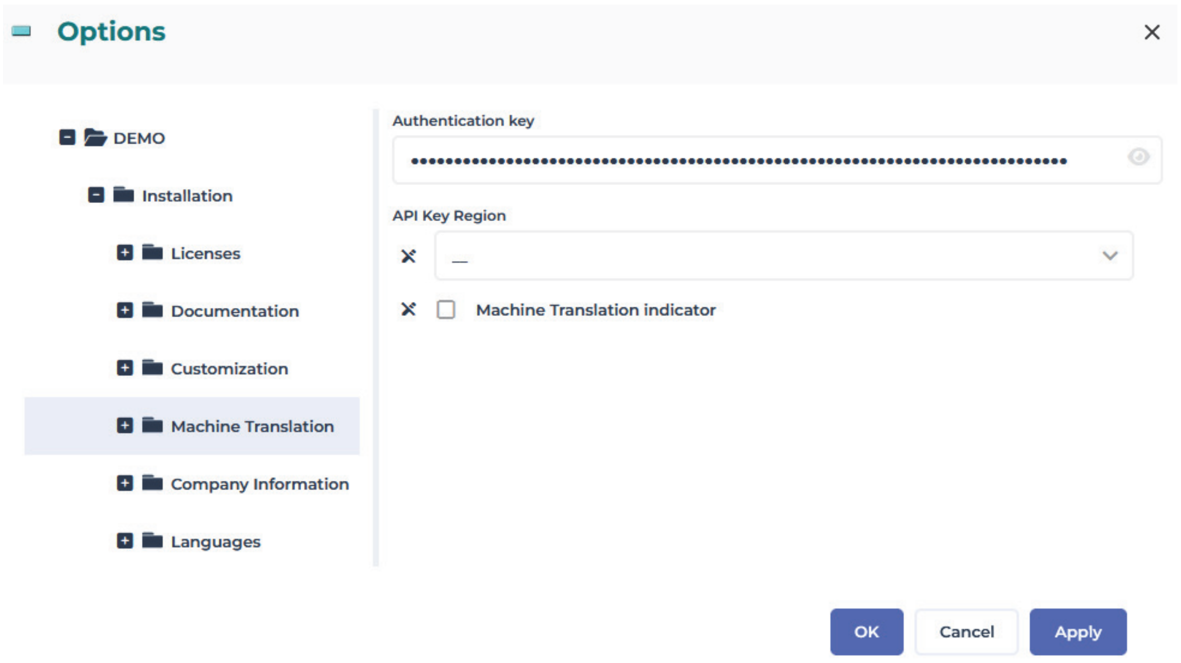
An option enables an administrator to enter the **Authentication key** number and its region, enabling activation of the automatic translation APIs.

To activate this option:

1. Connect to **HOPEX Administration** desktop.

   ☛ *See Accessing Web Administration Desktop.*

2. Click the **Environment Options** navigation menu.
3. In the **Options** tree, select **Installation > Machine Translation**.
4. In **Authentication key** field paste your API key.



5. If your API key is region-dependent, select the **API Key Region**.

6. If you want to indicate that the translation was done automatically, select the **Machine Translation indicator** option.
   A note is displayed at the end of each translated text.

CONFIGURING THE INCREMENTAL AUTOMATIC TRANSLATION BATCH

To activate automatic translation batches, the Administrator must create a trigger in the administration tool which implements the incremental automatic translation macro. When the trigger has been created, you must then plan the automatic update of the translations. See Initiating automatic batch translation.

Before processing incremental automatic translations from a source language to a target language, these translations must be initialized via automatic batch processing. See Configuring the incremental automatic batch translation.

Initiating automatic batch translation

To create an automatic translation initialization trigger from a source language to a target language, the administrator must perform the following tasks:

1. Opening the triggers definition window.
2. Creating the automatic translation initialization trigger.
3. Creating a task specific to automatic translation initalization.

Opening the triggers definition window

To open the triggers definition window:

1. Open the **HOPEX Administration Windows Front-End** module.

   ☛ *For further information about logging in to **HOPEX Administration Windows Front-End**, see the "Access **HOPEX Administration**" chapter in the **HOPEX Administration - Supervisor** guide.*

2. Open the environment that interests you.
3. Expand the repository folder for which you wish to translate the data.
4. Right-click the **Scheduler** folder and select **Manage Triggers**.
   The triggers definition window is opened.

Creating the automatic translation initialization trigger

To create an automatic translation initialization trigger:

1. Open the triggers definition window.
2. Select the **Triggers Definitions** tab.
3. Click the **New** button to create a **System Trigger Definition** specific to automatic translation.
4. In the creation wizard of the **System Trigger Definition**, create **System Job Definition**.
5. In the creation wizard of the **System Job Definition**, from the **Implementation** field, link the “Machine Translation Scheduler [All] Macro” and click **OK**.
6. In the creation wizard of the **System Trigger Definition**, click **Next**.
7. Specify the time and date of the batch launch.

8. Set the **Recurrence Type** at “One time”.
9. Click **Finish**.

Creating a task specific to automatic translation initalization

To create an initializing automatic translation user trigger from a source language to a target language:

1. Open the triggers definition window.
2. Select the **User Triggers** tab.
3. Click the **New** button to create a task specific to automatic translation initialization.
4. In the creation wizard, select the **System Job Definition** for automatic translation initialization.

   ☛ *For more details, see Creating the automatic translation initialization trigger.*

5. Specify the trigger name.
6. Enter the text of the **Job Context** in the following form: "[source language],[target language]," (a comma must be added at the end).

   ```
   For example, "it,fr," signifies that the source language is
   Italian and the target language is French.
   ```

   ☛ *If a language is not specified, it will be replaced by the data language.*

   ☛ *For more information about language abbreviations, see Table of abbreviations associated with languages.*

7. Click **Finish**.
   The batch will be activated at the time stipulated in the **System Job Definition** to which it is linked.

Configuring the incremental automatic batch translation

The Administrator must therefore carry out the following tasks:

1. Activate initial automatic translation, see Initiating automatic batch translation.
2. Opening the triggers definition window.
3. Creating the incremental automatic translation trigger.
4. Creating a task for incremental automatic translation.

Creating the incremental automatic translation trigger

To create an automatic translation initialization trigger:

1. Open the triggers definition window.
2. Select the **Triggers Definitions** tab.
3. Click the **New** button to create a **System Trigger Definition** specific to automatic translation.
4. In the creation wizard of the **System Trigger Definition**, create **System Job Definition**.
5. In the creation wizard of the **System Job Definition**, from the **Implementation** field, link the "Machine Translation Scheduler [Incremental] Macro" and click **OK**.

6. In the creation wizard of the **System Trigger Definition**, click **Next**.
7. Specify the time and date of the batch launch.
8. Set the **Recurrence Type** for the period best corresponding to your activity.
9. Click **Finish**.

Creating a task for incremental automatic translation

The procedure for creating a **user trigger** for incremental automatic translation from a source language to a target language is identical to the procedure for a **user trigger** for automatic translation initialization (see Creating a task specific to automatic translation initalization).

However, in the user trigger creation wizard, select the **System Job Definition** for incremental automatic translation.

☛ *For more details, see Creating the incremental automatic translation trigger.*

The batch will be activated at the time stipulated in the **System Job Definition** to which it is linked.

Table of abbreviations associated with languages

Language	**Abbreviation**
Arabic	ar
Bosnian	bs
Bulgarian	bg
Chinese (Simplified)	zh-Hans
Chinese (Traditional)	zh-Hant
Croatian	hr
Czech	cs
Danish	da
German	de
Dutch	nl
English	en
Spanish	es
Finnish	fi
French	fr
Greek	el

Language	Abbreviation
Hebrew	he
Hungarian	hu
Icelandic	is
Indonesian	id
Italian	it
Japanese	ja
Korean	ko
Malay	ms
Norwegian	nb
Polish	pl
Portuguese	pt-br
Romanian	ro
Russian	ru
Serbian	sr-Latn
Slovak	sk
Slovenian	sl
Swedish	sv
Thai	th
Turkish	tr
Vietnamese	vi