

Preface

The *DirXmetahub Migration Guide* describes how to migrate from DirXmetahub V6.0A10 or V6.0B00 to V6.0C00.

DirXmetahub Document Set

The DirXmetahub document set consists of the following manuals:

- *DirXmetahub Administration Guide*. Use this book to obtain a description of DirXmetahub architecture and components and to understand the basic tasks of DirXmetahub administration using DirXmetahub Manager, the DirXmetahub configuration database, and the DirXmetahub server runtime.
- *DirXmetahub Meta Controller Reference*. Use this book to obtain reference information about DirXmetahub server programs, scripts, and files.
- *DirXmetahub Meta Agent Reference*. Use this book to obtain reference information about the DirXmetahub meta agent programs, scripts, and files.
- *DirXmetahub Troubleshooting Guide*. Use this book to track down and solve problems in your DirXmetahub installation.
- *DirXmetahub Migration Guide* (this manual). Use this book to migrate from DirXmetahub V6.0A10 or V6.0B00 to V6.0C00.
- *DirXmetahub Installation Guide*. Use this book to install DirXmetahub.
- *DirXmetahub Release Notes*. Use this book to understand the features and limitations of the current release. This document is shipped with the DirXmetahub installation as the file **Readme.txt**.

Notation Conventions

Boldface type

In command syntax, bold words and characters represent commands or keywords that must be entered exactly as shown.

In examples, bold words and characters represent user input.

Italic type

In command syntax, italic words and characters represent placeholders for information that you must supply.

[]

In command syntax, square braces enclose optional items.

{ }

In command syntax, braces enclose a list from which you must choose one item.

In Tcl syntax, you must actually type in the braces, which will appear in boldface type.

|

In command syntax, the vertical bar separates items in a list of choices.

...

In command syntax, ellipses indicate that the previous item can be repeated.

install_path

The exact name of the root of the directory where DirXmetahub programs and files are installed. The default installation directory is *userID_home_directory/DirXmetahub* on UNIX systems and **C:\Program Files\Siemens\DirXmetahub** on Windows NT/Windows 2000 systems. During installation the installation directory can be specified. In this manual the installation-specific portion of pathnames is represented by the notation *install_path*. This manual uses the Windows NT/2000 style for pathnames.

1 Introduction

This migration guide helps to perform the migration from DirXmetahub V6.0A10 or V6.0B00 to V6.0C00.

It describes

- Chapter 2: The migration concept
- Chapter 3: Performing the automatic migration
- Chapter 4: Performing additional manual migration steps

2 Migration Concept

Many concepts in DirXmetahub have been extended or have been changed especially between V6.0A10 and the higher versions. To use the new features, migration of the database is absolutely necessary.

A concept for upgrade installation and migration of the configuration database for DirXmetahub was developed. It consists of:

- A reference database, which is used to perform a new installation. In this case of course a migration is not necessary. All new features are directly available.
- Upgrade scripts together with the relevant attributes in the database (schema and content version at the root node object of the configuration tree) that allow performing an update from previous versions of the configuration database to the actual one.

An upgrade includes a full update of the default applications of DirXmetahub's configuration database. Thus it contains all new features the new versions provides.

Nevertheless upgrades must be handled carefully to not disturb the copied workflows of the customer. Parts of the copied information must be changed or extended to be able to use the features of the new version.

To be absolutely sure that the customer information is not touched, all centrally used information (e.g. Tcl scripts) are copied to a special tree under Configuration:

Configuration -> Versions -> *version* -> Tcl

Version stands for the previous DirXmetahub version (e.g. 60A10 or 60B00 when an update to 60C00 is to be performed). The relevant links are automatically set to this location. So the customer information is not touched at all.

Nevertheless to use new features, the customer must perform manual steps to use the new information for his workflows (for example he must change links to the above mentioned versions folder to the newly delivered versions of these objects and afterwards adapt its workflow accordingly). These procedures are described below.

The steps of the automatic migration procedure from V6.0A10 or V6.0B00 to 6.0C00 are described in the next sections.

2.1 Pre migration steps

1. Central scripts are copied to the location described above.
2. All changes are performed, which have to be done before the structural changes. These are:
 - Use of central INI-Files
 - Deletion of unnecessary INI-Rules
 - Content of INI-Files updated
 - ODBC Workflows use central object and mapping editor
 - New mapping function IStringRange
 - Meta2ADS mapping script corrected
 - Mapping items of LDIF2Meta mapping adapted
 - ModifyDatafile of BA_file2MetaStore_Join_Full_Data Connected Directory corrected
 - DSML Workflows corrected
 - Central Delta Import Control TCL Script references now use the 'use_DN' switch
 - Attribute configuration of MetaStore contains new NT attributes
 - Wrong Wizard Types in Join Workflows corrected
 - Many string references in central TCL Scripts are now enclosed by " to support blanks in these values.

2.2 Structural changes

The main changes are structural changes that make DirXmetahub easier to use and better to handle (this is only done in the upgrade from 6.0A10 to 6.0C00).

The first part removes the channel folder in several steps:

1. **Move all channels** from the central channel folder to the relevant connected directory objects as sub objects.
2. **Move all intermediate connected directories** from the connected directory folder to the relevant job object (that is the job object that produces this intermediate connected directory).
3. **Delete the channel folder** because it is empty now.
4. **Remove all channel mappings** because they are no longer needed. An automatic

channel mapping routine replaces the manual configuration.

The channels are now located under the connected directories and the intermediate connected directories under the jobs objects. Additionally the manual setup of the channel mapping is no longer necessary.

DirXmetahub allows defining individual folder structures in the new version. The next steps adapt the default applications to a suitable structure.

5. **Create default folders** in the connected directories, jobs, schedules and workflow folders
6. **Move the relevant objects** belonging to the default applications into these folders.

Copied objects from now on are located in the top level folder when the **New** connected directory or workflow wizards in the Global view are used. All links are updated with a complex algorithm correctly. In the expert view, the user can create its own folder structures and move the copied objects to these folders in a second manual step (**Move Object**).

If objects are copied in the expert view directly (**Copy Object**) then the copied object remains in the folder of the original object and only internal links are updated (no links are updated that point outside the object). The **Move Object** function can be used to shift it to the correct folder.

Some folders have been moved from the top level structure to the Configuration folder. The next steps describe this procedure:

7. **Move the Systems folder** to the Configuration folder.
8. **Move the Services folder** to the Configuration folder.
9. **Move the DirXmetahub Servers folder** to the Configuration folder.
10. **Move the Messaging Services folder** to the Configuration folder.

The last part of the migration procedure performs some additional changes:

11. **Rename all objects** (display names) to the changed naming rules (see the chapter **Default Application Object Naming Conventions** in the DirXmetahub Manager help).
12. **Adjust all descriptions** and add new ones to the newly created objects (especially folders).

Please note, that of course all links from or to objects that have been moved to other locations are automatically adapted to the new locations. That is the reason that the migration procedure needs a long time to finish.

2.3 Post migration steps

1. All changes are performed, which have to be done after the structural changes. These

are:

- Notification Workflows
- Superior Info of SAPfile2Meta_Join_Full workflow
- MetaStore Attribute configuration update
- Usage of anchor attribute in Activities
- Use of the mapping function `convert_value_import` in default applications
- Removal of objects no longer needed.

The last step copies all default applications to the configuration database to be sure that the latest information is available in the configuration directory.

2. Update the DirXmetahub history file **install_history.txt** in the folder *inst_path*.

3 Automatic Migration

The steps of an upgrade installation for this version of DirXmetahub are:

1. **Save the database** with the native tools of your LDAP directory server product.
2. **Perform the installation** of the new version of DirXmetahub
3. **Save the database again** with the native tools of your LDAP directory server product.
4. **Perform the automatic migration procedure:**
 - Only on Sun platforms: Change to the folder `inst_path/confdb`. Set the correct passwords for the variables `METADIR_PW` in the file `basic.input.tcl`.
 - Change to the folder `inst_path/Gui/tools/migration-60A10to60C00` or `migration-60B00to60C00`.
 - Start `migration.bat`.
 - A graphical user interface displays the migration steps.
 - If errors occur, view the files in folder `inst_path/tools/migration-migration-60A10to60C00/trace` or `migration-60B00to60C00/trace`
 - Only on Sun platforms: Delete the password in the `basic.input.tcl` file.
5. **Save the database again** with the native tools of your LDAP directory server product.
6. **Restart the DirXmetahub Server**, otherwise the server does not work correctly with the moved DirXmetahub Server object.

4 Manual Migration

Copied workflows of the customer can only be partly migrated automatically. The next chapters describe topics that are eventually relevant for a specific customer situation. This is of course highly dependent on the workflows, which have been used at the customer site.

4.1 Central INI-Files

Previous State

In V6.0A10 INI-Files were located under the Job which calls the Agent (e.g. MetaStore2nt_Full_NTAgent Import-INI-File).

Actual State

INIFiles(Templates) are now located in **Configuration -> Agent Types -> <Agenttype>**. All Default Workflows use these INI Files.

How to get to actual state

If you want your old WF to use the central INI-Files do the following:

- In expert view select the Job (e.g. MetaStore2nt_Full_NTAgent)
- Select Input/Output Channel
- Press edit and select the INI-File("..." button) (e.g. **Configuration -> Agent Types -> NT -> Import-INI-File**)
- Save your changes.

Be aware of changes you made in your old INI-File.

4.2 Content of INI-Files updated

Previous State

Some INI-Files used the channels of the related job to get the selected attributes. (e.g.

ODBC Import used its input-channel). But the wizard used another channel to modify the selected attributes. Therefore changes made by the wizard didn't affect the content of the generated INI-File.

Actual State

INIFiles(Templates) are now using the metacp channels to get the selected Attributes.

Also the Wizards use these channels for update of selected attributes.

The following INI-Files have been changed:

- Notes Export-INI-File
- ODBC Import-INI-File
- ODBC Export-INI-File

How to get to actual state

For the above problem patches were available for V6.0A10. If the problem wasn't solved by these patches or manually you can perform the same steps described in **Central INI Files** to solve the problem

- In expert view select the Job (e.g. MetaStore2ODBC_Full_ODBCImport)
- Select Input/Output Channel
- Press edit and select the INI-File("..." button) (e.g. **Configuration -> Agent Types -> ODBC -> Import-INI-File**)
- Save your changes.

Be aware of changes you made in your old INI-File.

4.3 Workflows use central objects and mapping editor

Previous State

Workflows (for example ODBC workflows) used job local TCL-Files located under the Job. Mapping was done via a pure TCL script located under the job.

Actual State

ODBC Workflow ODBC2meta_Full uses central "Delta Import" Profile and Control TCL Script.

Meta2ODBC uses central "Full Export Notify" Profile and Control TCL scripts.

Both workflows use a mapping, which is created via the mapping editor.

How to get to actual state

Tcl Control/Profile Script must be compatible to the mapping. Profile Scripts and mapping uses handles called rh<something>. In old Scripts rh and rh_file was used. The central profile script uses handles called rh_<rolename>. Where rolename is taken from the corresponding Channel.

So if you just use the central profile script your existing mapping script may be incompatible because the profile script uses rh_ldap and rh_file (not rh). Mappings generated by the mapping editor name the handles in the same way as the central profile script. So it's a good idea to use the mapping editor when you use the central scripts. If you just use the central script, you have to change your mapping script so it uses the same handle names.(e.g. change all rh to rh_ldap).

If you want your old WF to use the central TCL-Files do the following:

- In expert view select the metaCP Job (e.g. MetaStore2nt_Full_metaCP)
- Select TCL Scripts
- Press edit and select the Profile and Control("..." button) Scripts (e.g. **Configuration -> TCL -> Profile Scripts -> Full Export**)
- Save your changes.

Be aware of changes you made in your old TCL Profile/Control Scripts.

If you want your old WF to use the mapping editor do the following:

- In expert view select the Job (e.g. MetaStore2nt_Full_metaCP)
- Right mouse click New-mapping script
- Edit your mapping
- Save your changes.
- Select the Job again and edit the TCL script tab
- In Mapping select your new mapping script
- Save your changes

How to convert a pure TCL mapping to a mappingeditor controlled mapping.

Let's have a look to the old Meta2ODBC Mapping Script:

```
# All Rights Reserved
#

proc convert_value { value } {
# =====
```

```

# escapes the 'dirxcp' special characters in an attribute value
#
# Parameters:
#     value          attribute value that should be escaped
#
# Return Value:
#     conv_value     escaped attribute value
#
set conv_value ""
foreach elem $value {
    append conv_value [meta unescapechar $elem -ldap]
}

return $conv_value
}

proc perform_mapping args {
# =====
# performs the mapping between the attribute values of the SEARCH result
# ("rh_<?job@InputChannel-DN@RoleName/>(…)" data fields) and the data fields that
# should be printed
# in the output file ("rh(…)" of the MS-Exchange agent
#
# Parameters:
#     -
#

global debug_trace
global rh
global rh_<?job@InputChannel-DN@RoleName/>

#
# dump the X.500 attribute values ("rh_<?job@InputChannel-DN@RoleName/>(…)"
fields) either on screen
# or into the tracefile
#
if {$debug_trace == 1} then {
    puts "\nX500-ATTRS"
    foreach var [array names rh_<?job@InputChannel-DN@RoleName/>] {
        if {$rh_<?job@InputChannel-DN@RoleName/>($var) != ""} then {
            puts "rh_<?job@InputChannel-DN@RoleName/>($var) =
$rh_<?job@InputChannel-DN@RoleName/>($var)"
        }
    }
}
}

```

```

} else {
  if {$debug_trace == 2} then {
    meta writetrace -text "\n# X500-ATTRS"
    foreach var [array names rh_ <?job@InputChannel-DN@RoleName/>] {
      if {$rh_ <?job@InputChannel-DN@RoleName/>($var) != ""} then {
        meta writetrace -text "\t<?job@InputChannel-DN@RoleName/>($var) =
$rh_ <?job@InputChannel-DN@RoleName/>($var)"
      }
    }
  }
}

if { $rh_ <?job@InputChannel-DN@RoleName/>(dxmDelus) == "TRUE" } then {
  set rh(ODDEL) "delete"
}

set rh(PNR) [convert_value $rh_ <?job@InputChannel-
DN@RoleName/>(employeeNumber)]

set rh(RMB) [convert_value $rh_ <?job@InputChannel-DN@RoleName/>(mail)]
if { $rh(RMB) == "" } then {
  set rh(RMB) " "
}

# set rh(FAX) [convert_value $rh_ <?job@InputChannel-
DN@RoleName/>(collectiveFacsimileTelephoneNumber)]

# set rh(PHONE) [convert_value $rh_ <?job@InputChannel-
DN@RoleName/>(collectiveTelephoneNumber)]

set rh(SN) [convert_value $rh_ <?job@InputChannel-DN@RoleName/>(sn)]

set rh(DEP) [convert_value $rh_ <?job@InputChannel-DN@RoleName/>(ou)]

#
# dump the generated MS-Exchange attribute values ("rh(...) " fields) either
# on screen or into the tracefile
#
if {$debug_trace == 1} then {
  puts "\nMAPPED-INFO"
  foreach var [array names rh] {
    if {$rh($var) != ""} then {
      puts "rh($var) = $rh($var)"
    }
  }
}

```

```

    }
  } else {
    if {$debug_trace == 2} then {
      meta writetrace -text "\n# MAPPED-INFO"
      foreach var [array names rh] {
        if {$rh($var) != ""} then {
          meta writetrace -text "\trh($var) = $rh($var)"
        }
      }
    }
  }
}

```

There are two procedures **convert_value** and **perform_mapping**. **Convert_value** is a “mapping function”. This mapping_function must be defined in the folder **Configuration - > TCL -> Mapping Functions** in the expert view.

Note: Due to a generalized concept all mapping functions have been renamed. For compatibility reasons the old and new functions are still available. Use `!StringUnescape` for `convert_value`.

`Convert_value` and about 50 more are mapping_functions that come with DirXmetahub by default. If you want to define own mapping functions please define it using the GUI feature **New -> Mapping Function**. See **Defining own mapping functions** in the DirXmetahub Manager help.

`Perform_mapping` is the mapping itself. At the beginning and at the end of this procedure there is tracing stuff. If you use the mapping editor, tracing is generated automatically.

All statements like

```
set rh(SN) [convert_value $rh_<?job@InputChannel-DN@RoleName/>(sn)]
```

can be converted to a mappingItem very easy. See the Mapping items in the following screenshot.

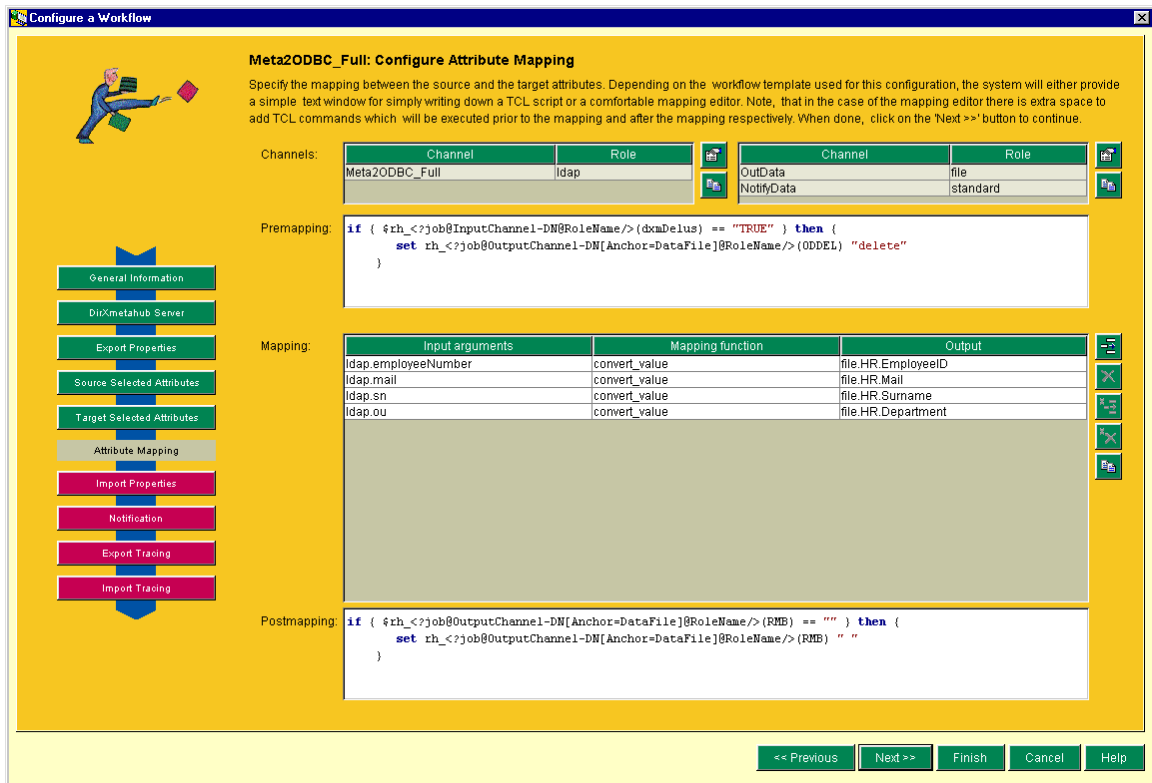
```

if { $rh_<?job@InputChannel-DN@RoleName/>(dxmDelus) == "TRUE" } then {
  set rh(ODDEL) "delete"
}
set rh(RMB) [convert_value $rh_<?job@InputChannel-DN@RoleName/>(mail)]
if { $rh(RMB) == "" } then {
  set rh(RMB) " "
}

```

These statements can't be formulated directly as mapping items. We put the first if-

statement to the premapping . The second if-statement goes to the postmapping, because RMB has to be mapped first and afterwards it can be checked if its an empty string.



All mapping functions (here only `convert_value`) are generated into the mapping. The Mapping generated by the mapping editor is:

```
# Mapping script "Mapping Script"
# Generated by DirXmetahub-Manager at <?date/> <?time/>

proc debug.out args {
    # =====
    # Performs the debug print out.
    #
    # Parameters:
    #     args    attribute array to be printed
    #
    # No return parameters.
```

```

#

global debug_trace

switch -exact -- $debug_trace {
  "1" {
    foreach part $args {
      puts "$part"
    }
  }
  "2" {
    set cmd "meta writetrace -text \\""
    set text ""
    foreach part $args {
      append text " $part"
    }
    append cmd "$text \\""
    regsub -all {\$} $cmd {\|\$} cmd
    set status [ catch { eval $cmd } result ]
  }
  default {
  }
}
}

#
# Copyright (c) 2000-2001 SIEMENS AG
# All Rights Reserved
#

proc convert_value { value } {
# =====
# escapes the 'dirxcp' special characters in an attribute value
#
# Parameters:
#   value           attribute value that should be escaped
#
# Return Value:
#   conv_value     escaped attribute value
#
  set conv_value ""
  foreach elem $value {
    lappend conv_value [meta unescapechar $elem -character ";{}"]
  }
}

```

```

    return $conv_value
}

proc perform_mapping {args} {

    global debug_trace
    global rh_ldap
    global rh_file
    global rh_standard
    if { $rh_<?job@InputChannel-DN@RoleName/>(dxmDelus) == "TRUE" } then {
        set rh_<?job@OutputChannel-DN[Anchor=DataFile]@RoleName/>(ODDEL)
"delete"
    }

    if {$debug_trace != 0} then {
        debug.out "\n# REC-INFO"
        foreach var [array names rh_ldap] {
            if {$rh_ldap($var) != ""} then {
                debug.out "rh_ldap($var) = $rh_ldap($var)"
            }
        }
    }

    set rh_file(PNR) [convert_value $rh_ldap(employeeNumber)]
    set rh_file(RMB) [convert_value $rh_ldap(mail)]
    set rh_file(SN) [convert_value $rh_ldap(sn)]
    set rh_file(DEP) [convert_value $rh_ldap(ou)]

    if { $rh_<?job@OutputChannel-DN[Anchor=DataFile]@RoleName/>(RMB) == "" } then
    {
        set rh_<?job@OutputChannel-DN[Anchor=DataFile]@RoleName/>(RMB) " "
    }

    if {$debug_trace != 0} then {
        debug.out "\n# MAPPED-INFO"
        foreach var [array names rh_file] {
            if {$rh_file($var) != ""} then {
                debug.out "rh_file($var) = $rh_file($var)"
            }
        }
    }
}

```

```
}  
}
```

Alternatively to the pre/postmapping you can define own mapping_functions that implement the control logic. This is especially helpful if you need a function several times.

4.4 Meta2ADS mapping script corrected

Previous State

Erroneous Post Mapping. Problem occurred when `dxmADsResetUserPassword) = "TRUE"`.

Actual State

Mapping Script under Meta2ADS_Full_MetaCP Job is updated.

4.5 Mapping items of LDIF2Meta mapping

Previous State

In V6.0A10 mapping was incorrect (List of mapping items contained non existing items).

Actual State

List of mappingitems was corrected.

4.6 ModifyDatafile of BA_file2MetaStore_Join_Full_Data Connected Directory

Previous State

The ModifyDatafiles had two cn values DataFile and ModifyDatafile .This caused some problem viewing this object in the GUI.

Actual State

The cn of the modifyDatafile is now the correct value ModifyDatafile.

How to get to actual state

If you want correct this in your old workflows, which use copies of this Connected directory please do the following:

- Eliminate the cn Datafile via DirXmanage (if you use DirX as directory server)
- or
- Use the following LDIF-change file and import it with the native directory tool.

```
#####
#
# delete cn Datafile from ModifyDatafile for file2meta join wf
#
dn: cn=ModifyDatafile,cn=BA_file2MetaStore_Join_Full_Data,dxmC=Connected
Directories,dxmC=DirXmetahub
changetype: modify
delete: cn
cn: Datafile
-
```

4.7 DSML Workflows

Previous State

In some cases the workflows failed

Actual State

The following changes have been made

- Source statement in profile.tcl corrected
- Variable server_address is used instead of dsa_address
- Central common.tcl is used
- delete unnecessary second hardcoded superiorinfo in TCL control script
- correct superiorinfo ou->organizationalUnit

How to get to actual state

Default application workflows are up to date. To update copies of the default DSML workflows :

- Use the same TCL Scripts as the default applications. If you have changed the Scripts you have to merge the changes
- Correct the superiorinfo in tab Join/Import properties of the channel that points to the metaDirectory. Change ou to organizationalUnit

4.8 Doubled Tcl scripts in DSML Workflow

Previous State

Not relevant.

Actual State

You copy the Meta2DSML_Full workflow to your private scenario. The workflow does not run. The error message is 'Doubled display name'. The reason is a doubled profile and control script in the original object.

How to get to actual state

- Open the expert view
- Navigate to the job Meta2DSML_Full_Metacp
- Open this node. You will see two profile and control scripts
- Check which control and profile script is used by this job (follow the link from the Tcl Scripts tab in the job object)
- Delete the other two scripts that are not referenced.

4.9 Central Delta Import Control TCL Script references use_DN

Previous State

Use_DN was hardcoded in the TCL Script.

Actual State

Use_DN is now taken as reference from the appropriate channel object.

How to get to actual state

If you want your old WF to use this central TCL-File do the following:

- In global view configure your Workflow
- Goto Join/Import properties
- Set use_DN to the appropriate value (in A10 FALSE was hardcoded)
- Save your changes with finish

4.10 Attribute configuration updated

Previous State

Attribute configuration contained only the attributes supported in 60A10. Not all names

were equal to the abbreviation. The handling of the different names was therefore complicated.

Actual State

NT Agent supports more NT attributes as in V6.0A10. The attribute configuration of MetaStore and NT was updated. The ADS attribute configuration contains additional attributes.

The names of all connected directories have been set to the to the abbreviations. Some abbreviations have been set to the LDAP names (name and abbreviation has changed). So you use the same names at any point of the graphical user interface.

How to get to actual state

If you have copies of these connected directories and you want to use the actual attribute configuration of any of the updated connected directories, please perform the following steps:

- Select in expert view **Connected Directories -> Default -> *connected_directory_name***
- Select Attribute Configuration
- Export CFG File
- Finish or cancel
- In global view adapt your copy of the connected directory:
- Select Attribute Configuration
- Import CFG File select the former created file
- Save your changes with Finish

4.11 Wrong Wizard Types in Join Workflows

Previous State

For BA_notes2MetaStore_Join_Full and BA_exchange2MetaStore_Join_Full wrong wizard types were stored in the workflow objects.

Actual State

The wizard types are now correct.

How to get to actual state

- In expert view select the Workflow (e.g. exchange2MetaStore_Join_Full)
- Edit

- Change the attribute Wizard (e.g. to EXCHANGE-LDAP)
- Save your changes.

4.12 String references in central TCL Scripts are enclosed by “

Previous State

Some String references weren't enclosed in “. So if your String contains blank you had to set the “ manually .

Actual State

String references like user_name, user_pwd, dsa_address, base_obj are now included in “. So if your using characters like blank the script takes the correct string.

How to get to actual state

If you want your old WF to use the central TCL-Files and you have stringreferences manually includes in “ you have to delete these “.

4.13 Notification Workflows

Previous State

The notification feature was not available in previous versions.

Actual State

The default workflows **Meta2NT_Full**, **Meta2ODBC_Full** and **Meta2ADS** are now using the central “Full Export Notify” Scripts that support a special application of the notification feature. See **Understanding Notifications** in the DirXmetahub help for more information.

Note that in the default configuration of these workflows the notification feature is not activated. You have to activate it via the wizard.

You can use the notification agent in a lot of other situations wherever you use the meta controller with the Tcl language. See **Understanding Notifications** in the DirXmetahub help for more information.

There is a new Folder **Configuration -> Notifications** where you can place central notification configuration files.

How to get to actual state

See **Understanding Notifications** in the DirXmetahub help for more information.

4.14 Superior Info of SAPfile2Meta_Join_Full workflow

Previous State

Due to some missing blanks the superior info was incorrect. If you run this workflow as the first one of all join workflows the attempt to create the people OU failed.

Actual State

Superior info is now correct.

How to get to actual state

If you copied the SAP workflow, check whether the automatic migration corrected your copy. If not, perform the change manually.

4.15 Anchor attribute in Activities

Previous State

Anchor attribute wasn't set.

Actual State

All activities of Workflows that are configured via the ADS-LDAP, NDS-LDAP and EXCHANGE-LDAP wizard have now been changed to use the dxmAnchor attribute. Above wizards use this attribute to determine which objects have to be changed.

How to get to actual state

Migration does this automatically.

4.16 Anchor attribute in Bind Profiles

Previous State

It was not possible to change the display name of a bind profile because set references in the relevant Tcl or INI files did not work any longer.

Actual State

Bind profiles now contain an Anchor attribute with the value **Admin** that is set for all default applications. The references in the Tcl and INI files have been updated accordingly. Previously copied connected directories including the bind profiles do not contain this attribute value.

How to get to actual state

If you encounter a reference resolution error during a workflow run that contains the text **[Anchor=Admin]**, add the value **Anchor** into the anchor attribute of this bind profile.

4.17 Use of convert_value_import in default applications

Previous State

Some workflows use convert_value for import to MetaStore. This leads to erroneous behavior if the imported data contains special characters. In V6.0A10 INI-Files were located under the Job which calls the Agent (e.g. MetaStore2nt_Full_NTAgent Import-INI-File) .

Actual State

All default application Workflows that import to MetaStore now use convert_value_import mapping function instead of convert_value.

Note: Due to a generalized concept all mapping functions have been renamed. For compatibility reasons the old and new functions are still available. Use IStringUnescape for convert_value and IStringEscape for convert_value_import. All default applications have been adapted to use the new style mapping functions.

How to get to actual state

If you want your copied import workflows that use the convert_value function to use the convert_value_import mapping function do the following:

- Open the wizard for your workflow in the global view or select the Job in the expert view (e.g. MetaStore2nt_Full_NTAgent) and then select the Mapping Script sub object. In the latter case select Edit.
- Select all lines that contain the convert_value function and change it to the convert_value_import function.
- End the wizard or save your changes (if you worked in expert view).

DirXmetahub V 6.0

Migration Guide
Edition March 2002



Table of Contents

Preface	1
DirXmetahub Document Set.....	1
Notation Conventions	1
1 Introduction.....	3
2 Migration Concept	5
2.1 Pre migration steps.....	6
2.2 Structural changes.....	6
2.3 Post migration steps	7
3 Automatic Migration	9
4 Manual Migration	11
4.1 Central INI-Files.....	11
4.2 Content of INI-Files updated.....	11
4.3 Workflows use central objects and mapping editor	12
4.4 Meta2ADS mapping script corrected.....	20
4.5 Mapping items of LDIF2Meta mapping.....	20
4.6 ModifyDatafile of BA_file2MetaStore_Join_Full_Data Connected Directory	20
4.7 DSML Workflows	21
4.8 Doubled Tcl scripts in DSML Workflow	22
4.9 Central Delta Import Control TCL Script references use_DN.....	22
4.10 Attribute configuration updated.....	22
4.11 Wrong Wizard Types in Join Workflows.....	23
4.12 String references in central TCL Scripts are enclosed by “	24
4.13 Notification Workflows	24
4.14 Superior Info of SAPfile2Meta_Join_Full workflow.....	25
4.15 Anchor attribute in Activities	25
4.16 Anchor attribute in Bind Profiles	25
4.17 Use of convert_value_import in default applications	26
Table of Contents.....	i

Infoline:

Tel: +49 (89) 636-48878

Fax: +49 (89) 636-47168

E-Mail: directory@icn.siemens.de

Support:

<http://www.siemens.com/directory>

**Comments
Suggestions
Corrections
Courses**

The User Documentation
Department would like to know
your opinion on this manual.
Your feedback helps us to
optimize our documentation to
suit your individual needs.

All product names quoted are trademarks or registered trademarks of the manufacturers concerned.

© Copyright 1997-2002 Siemens AG

Distribution and reproduction not permitted without the consent of Siemens.

How to use the softbook

1 Structure

The softbook contains several chapters, plus possibly glossary, index, title sheet, table of contents, copyright and notes on how to use the softbook. The order in which they are listed here is the order in which they appear, i.e. the notes are at the end.

Changes and additions made after the copy deadline are documented as "current information" and can be found after the keywords in the softbook.

2 Printing the softbook

For best results print to a PostScript printer.

You can print either the contents of the entire softbook (maximal 255 pages at a time) or particular pages. You can find help for printing problems in the readme file located in the program folder of Acrobat reader.

If you want to compile your own printout of the book, it is advisable to place the title sheet and the table of contents, neither of which have any page numbers, in front of the first chapter.

3 Navigating

You can navigate through the softbook e.g. by paging up and down one page at a time or by retracing your steps through the document (previous view).

In the following some navigational structures are described.

a Bookmarks

The bookmarks palette in the navigation pane contains the structure of the softbook in the form of a visual table of contents. The texts and numbering appearing on the bookmarks correspond to the chapter headings. When you open the softbook, initially only the first-level headings are displayed.

The bookmarks are used to jump direct to the individual chapters and sections of the softbook.

b Index

The page numbers quoted after the keywords are in general linked to the corresponding pages. A click on the sensitive area will take you to the page containing the keyword.

4 Full-Text Index

This online documentation set also provides a full-text index generated by Acrobat Catalog®. To use this index you need *Adobe Acrobat Reader plus Search*.

The full-text index includes all online manuals of DirX or DirXmetahub. The corresponding index is attached automatically when an online manual is opened. All word options (Case sensitive, Sounds Like, and Word Stemming) were enabled when the index was built. There were no numbers or stopwords excluded from the index.