

knowledge MANAGEMENT

MultiUser Scripts

SIEMENS

Documentation

MultiUser Scripts

We reserve the right to make technical changes to this product.

Copyright

Reproduction, transmission, or use of this document or its contents is not permitted without express written authority. Offenders will be liable for damages. All rights, including rights created by patent grant or registration or a utility model or design, are reserved.

MultiUser Scripts

General Notes

NOTE

The Application Examples are not binding and do not claim to be complete regarding the circuits shown, equipping, and any eventuality. The Application Examples do not represent customer-specific solutions. They are only intended to provide support for typical applications. You are responsible in ensuring that the described products are correctly used. These Application Examples do not relieve you of the responsibility in safely and professionally using, installing, operating and servicing equipment. When using these Application Examples, you recognize that Siemens cannot be made liable for any damage/claims beyond the liability clause described. We reserve the right to make changes to these Application Examples at any time without prior notice. If there are any deviations between the recommendations provided in these Application Examples and other Siemens publications - e.g. Catalogs - then the contents of the other documents have priority.

Warranty, liability, and support

We do not accept any liability for the information contained in this document.

Any claims against us - based on whatever legal reason - resulting from the use of the examples, information, programs, engineering and performance data etc., described in this Application Examples shall be excluded. Such an exclusion shall not apply in the case of mandatory liability, e.g. under the German Product Liability Act ("Produkthaftungsgesetz"), in case of intent, gross negligence, or injury of life, body or health, guarantee for the quality of a product, fraudulent concealment of a deficiency or breach of a condition which goes to the root of the contract ("wesentliche Vertragspflichten"). However, claims arising from a breach of a condition which goes to the root of the contract shall be limited to the foreseeable damage which is intrinsic to the contract, unless caused by intent or gross negligence or based on mandatory liability for injury of life, body or health. The above provisions does not imply a change in the burden of proof to your detriment.

Copyright© 2007 Siemens A&D. It is not permissible to transfer or copy these standard applications or excerpts of them without first having prior authorization from Siemens A&D in writing.

Qualified personnel

In the sense of this documentation qualified personnel are those who are knowledgeable and qualified to mount/install, commission, operate and service/maintain the products which are to be used. He or she must have the appropriate qualifications to carry-out these activities
e.g.:

MultiUser Scripts

- Trained and authorized to energize and de-energize, ground and tag circuits and equipment according to applicable safety standards.
- Trained or instructed according to the latest safety standards in the care and use of the appropriate safety equipment.
- Trained in rendering first aid.

There is no explicit warning information in this documentation. However, reference is made to warning information and instructions in the Operating Instructions for the particular product.

Reference regarding export codes

AL: N

ECCN: N

MultiUser Scripts

Table of contents

Table of contents	5
1 Scripts for the application of MultiUser	6
1.1 Idea and benefit	6
1.2 Functionality	6
1.3 Installation	7
1.4 Start scripts	9
2 Settings	10
2.1 Project settings	10
2.2 MultiUser Script settings	10
2.2.1 List of all users	10
2.2.2 Current user	11
2.2.3 External data filing	12
2.2.4 Warnings	12
3 Use cases	13
3.1 Start MultiUser Script	13
3.2 Export from the project to the external data filing	14
3.3 Import from the external data filing to the project	17
3.4 Analyzing and testing functions	20
4 External data filing	22
4.1 Data filing path	22
4.2 Log files	22
4.3 Analysis files	22
5 Boundary conditions	23
5.1 TOs	23
5.2 Units	24
5.3 Libraries	24
5.4 Watch tables	25
5.5 Scripts	25
6 History	26

MultiUser Scripts

1 Scripts for the application of MultiUser

1.1 Idea and benefit

The missing support by a MultiUser environment within the engineering system of SIMOTION Scout lead to the intention to create this solution.

The goal of this solution, being based on SIMOTION scripting, is to offer you an automatic support for the development of a SIMOTION application within a team.

Focus is set on the exchange of project components (technology objects programs...) between the single team members.

We just want to offer you, as a user, quick and easy possibilities of configuration in combination with a safe utilization and an intuitive user interface.

1.2 Functionality

The scripts for MultiUser offer the possibility to work simultaneously with several developers on one SIMOTION project.

Single project components are assigned to certain users or user functions.

According to the user specification, the assigned components are entered or taken from a common data filing (e.g. network drive) via script. The script executes these processes automatically. The single user is relieved from the details of the handling so that possible faulty operations will be reduced.

All existing components in the common data filing indicate the current state of the project.

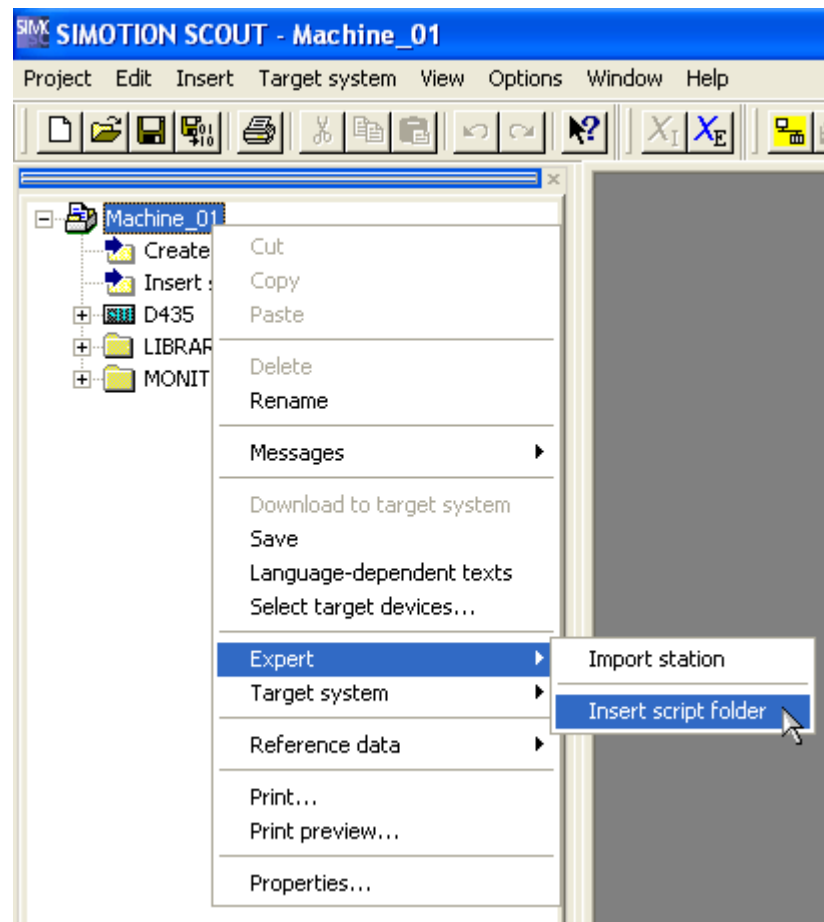
MultiUser Scripts

1.3 Installation

The scripts for MultiUser are provided as ASCII text and can be found, among other things, on the CD „SIMOTION Utilities & Applications“ in the sub-directory **3_SCRIPTSMultiUser** as files **MultiUser.txt**, and **MultiUserConfig.txt**.

On the project level, a script level has to be created. For this, please select the command **Expert → Insert script folder** in the context menu of the project (see figure 1-1 insert a script).

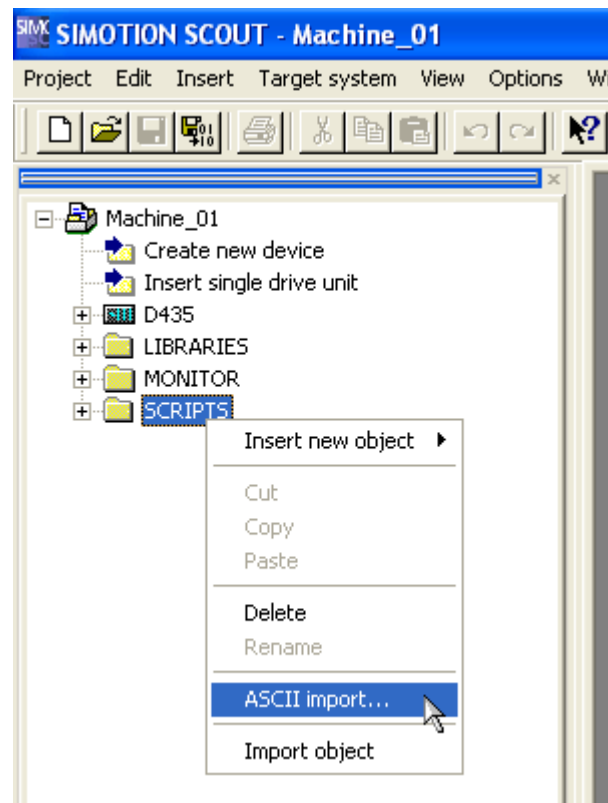
figure 1-1 insert a script folder



MultiUser Scripts

The scripts **MultiUser.txt** and **MultiUserConfig.txt** are imported as ASCII files in the script folder on the project level. For this, you have to select the command **ASCII import** in the context menu of the created script folder (see figure 1-2 import MultiUser scripts).

figure 1-2 import MultiUser scripts



MultiUser Scripts

1.4 Start scripts

There are three ways to start scripts within SIMOTION.

1. Select the script in the project navigator and select **Accept and execute** in the context menu.
2. Open the script with double click in the project navigator (or with the command **open**) and select the command **Accept and execute** in the function bar, while the opened script focused (see figure 1-3 start a script from the function bar).

figure 1-3 start a script from the function bar



3. Write an own script and call other scripts in the project with `scripts(<scriptName>).Execute` auf.
If a calling script and a script to be called are not in the same script file, it is necessary to navigate there in the call.
`PROJ.Devices(<DeviceName>).scripts(<scriptName>).Execute`

MultiUser Scripts

2 Settings

2.1 Project settings

Each user works on a personally assigned PG. At the beginning, there is a unified project state on each PG.

Each user stores his identification (e.g. last name) in the property **Author** of the objects he is responsible for (see figure 2-2).

The property **Version** remains empty and will be used by the MultiUser Scripts for an internal determination of the version.

The property **Comment** can be used as desired, e.g. for copyright information or specific version identifications.

The scripts are installed on the PGs according to chapter 1.3 and set according to chapter 2.2.

2.2 MultiUser Script settings

All settings being relevant for the user are stored in the script MultiUserConfig.

NOTE Hinweis zeigt auf einen möglichen Vorteil hin. Hat Tipp-Charakter

NOTE When placing strings in the script MultiUserConfig, it is necessary to consider capitalization of the words.

2.2.1 List of all users

In the array `g_asAllUsers()` a list of all users is stored (see figure 2-1 list of all users). This setting is universally valid and is recommended to be done before the import to the single projects so that all users can dispose of the same user list.

figure 2-1 list of all users

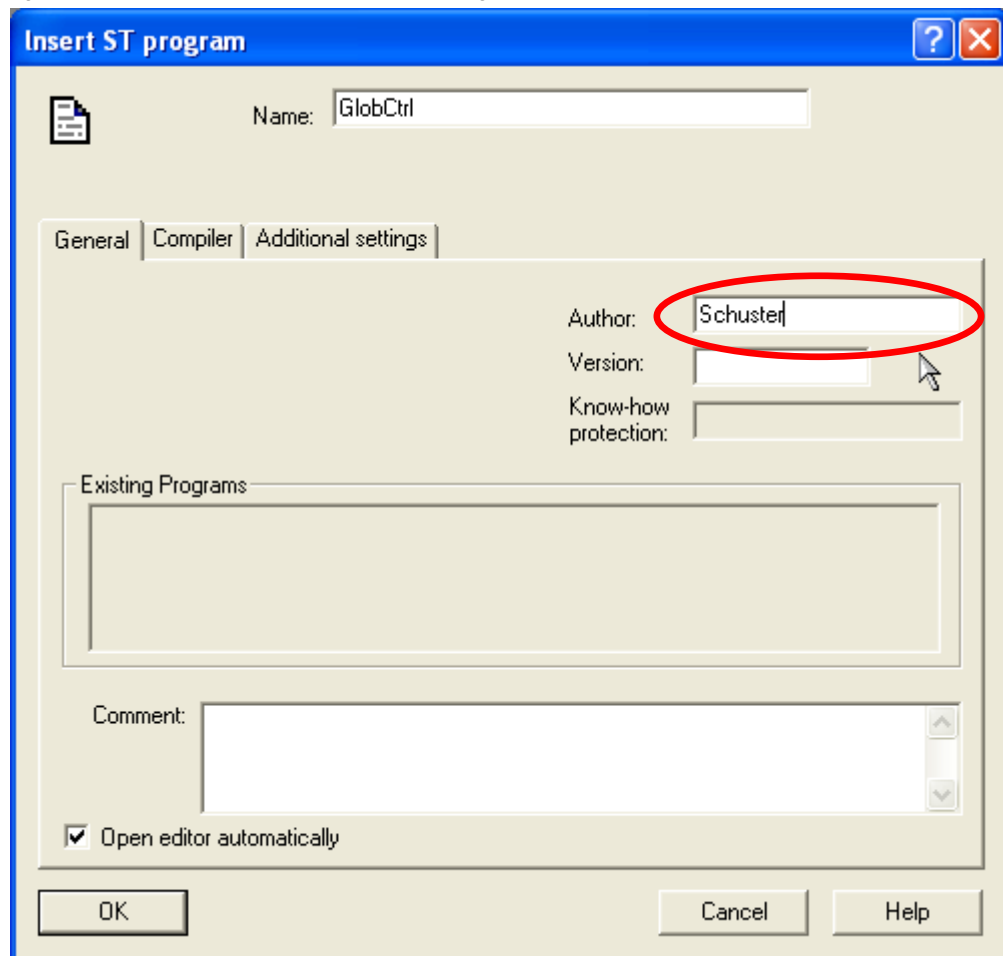
```
' List of all possible users / authors
Private g_asAllUsers
  g_asAllUsers = Array ( _
    "Bauer", _
    "Becker", _
    "Esser", _
    "Hansen", _
    "Jäger", _
    "Klein", _
    "Meier", _
```

MultiUser Scripts

```
"Müller", -
"Schmitz", -
"Schuster", -
"Wagner")
```

The names have to be identical to those being stored in the property **Author** of the single objects (see sample figure 2-2 properties author of a new ST program).

figure 2-2 properties author of a new ST program



2.2.2 Current user

The name of the current user is stored in the variable `g_sUser`. An element from the list of all users is assigned to the variable (figure 2-3 setting of the current user).

figure 2-3 setting of the current user

```
' Name of current user
Private g_sUser
  g_sUser = g_asAllUsers(0)
```

MultiUser Scripts

As an alternative, it is also possible to indicate a string that corresponds to an element of the user list.

This setting is user-specific and is recommended to be done after the import to the single projects.

2.2.3 External data filing

Indications of drive and path of the external data filing are stored in the variable `g_sPath` (figure 2-4 setting of the external data filing).

figure 2-4 setting of the external data filing

```
' Location of repository path
Private g_sPath
  g_sPath = "X:\Repository"
```

This setting is universally valid and is recommended to be done before the import to the single projects so that all users can dispose of the same data filing.

2.2.4 Warnings

Via the variable `g_boWarnOnSameVersion`, you can switch on or off the warning message during the import in case of a unit version to be imported being identical to the existing unit in the project.

Via the variable `g_`, you can switch on or off a warning message to be acknowledged during the import in case that an object to be imported does not exist in the project

figure 2-5 setting of warnings

```
' Display warning, if same version during import recognized
Private g_boWarnOnSameVersion
  g_boWarnOnSameVersion = False
' Display warning, if new object during import recognized
Private g_boWarnOnNewObject
  g_boWarnOnNewObject = False
```

MultiUser Scripts

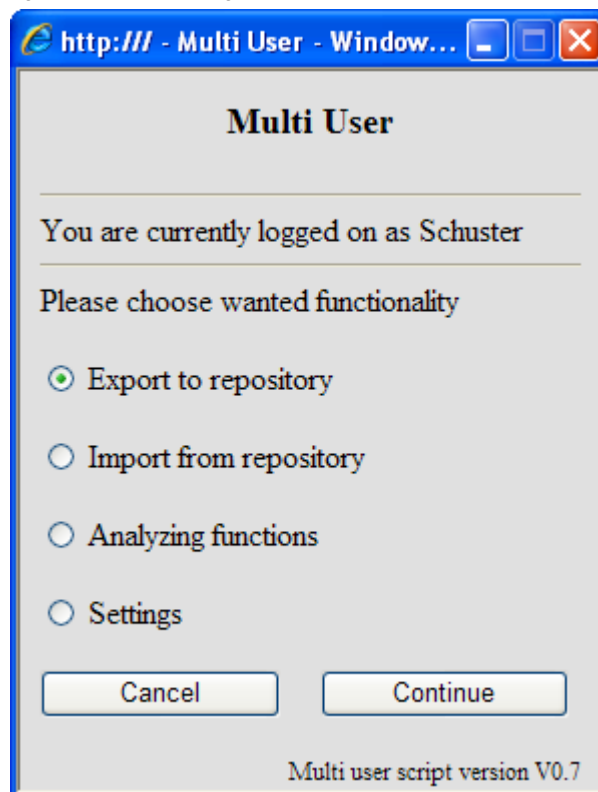
3 Use cases

Before using the MultiUser scripts, please consider the correct settings according to chapter 2.

3.1 Start MultiUser Script

The MultiUser functionality is started via the script MultiUser. The following dialog box appears.

figure 3-1 start dialog



Choose **Export to repository** to export the supported object types to the external data filing.

Choose **Import from repository** to import the supported object types from the external data filing.

Choose **Analyzing functions** to call various analyzing and test functions such as listing of all objects of a user, listing of all object with unknown or without authors...

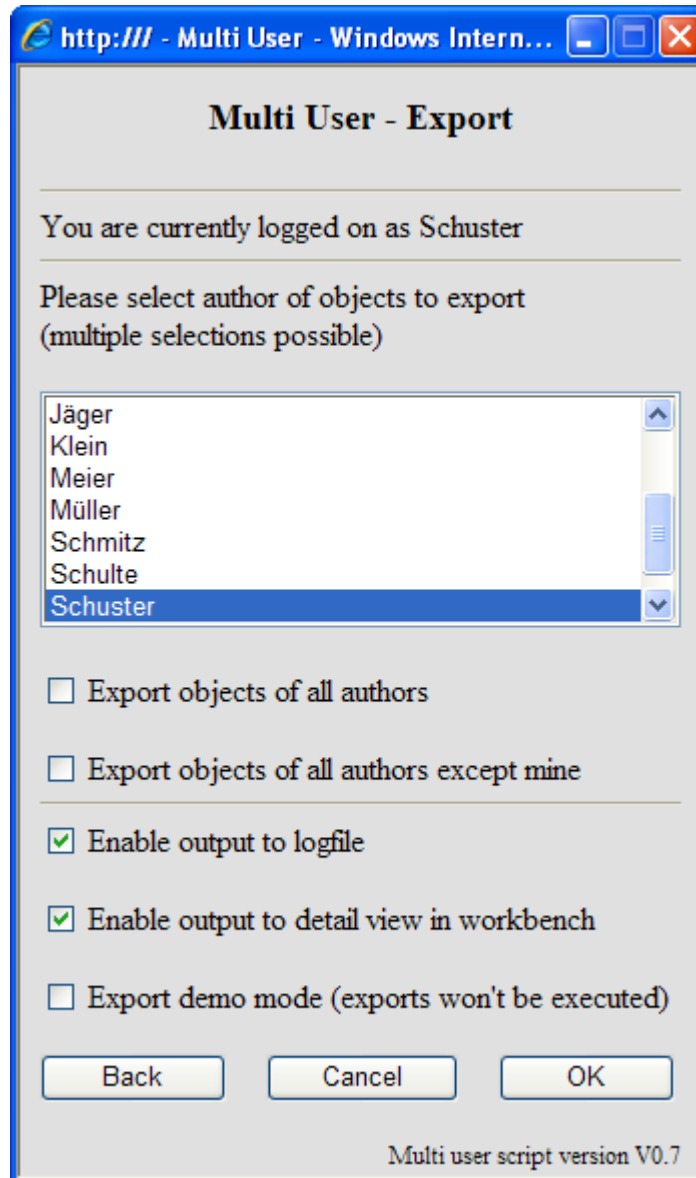
The function **Settings** is not supported in the current version.

MultiUser Scripts

3.2 Export from the project to the external data filing

After selecting the export function, you can make special settings for the export. With pre-setting, those objects, which are assigned to the current user, are exported.

figure 3-2 export dialog box



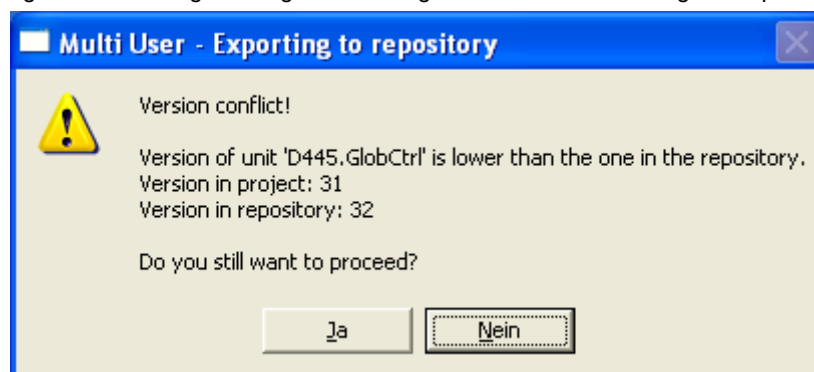
MultiUser Scripts

- Via the list field, you can select one or more authors whose objects are to be exported.
- **Export objects of all authors** exports all objects of all authors. The list field is deactivated with the selection of this option.
- **Export objects of all authors except mine** exports all objects of all authors except that of the current user. The list field is deactivated with the selection of this option.
- **Enable output to logfile** writes the executed exports to a special log file.
- **Enable output to detail view in workbench** writes the executed exports to the detail view in the Scout workbench.
- **Export demo mode** activates a demonstration mode where all actions are checked and recorded (logged) without executing a real export. By this, you can check in advance the executed actions.

With **OK**, the export is started and the following tests and actions are executed.

1. Via all SIMOTION devices existing in the project, all supported objects are checked if the author entered in the object corresponds to one of the selected ones.
2. If the author entered in the object is identical to one of the selected ones, the system checks if the object is already stored in the external data filing.
3. If the object is not stored in the external data filing, it will be exported. Before the export, the version identification is incremented. If the property is not occupied, a default value of 1 is set. There will be a simple version numbering with 1, 2, 3... in the range of numbers of an integer value of 32.
4. If the object already exists in the external data filing, the author and the version identification is read from there and then compared with the information stored in the project. If the author is different or if there is a higher version identification in the data filing, a warning message will appear and has to be acknowledged.

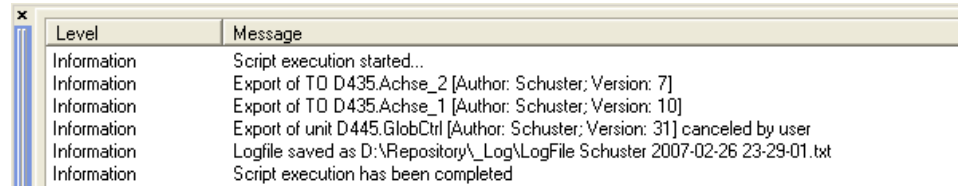
figure 3-3 warning message concerning a version conflict during the export



MultiUser Scripts

With pre-setting, the executed exports are documented in a log file and in the detail view.

figure 3-4 display in the detail view during the export



The screenshot shows a log window with a table of messages. The window has a title bar with a close button (X) and a scroll bar on the left. The table has two columns: 'Level' and 'Message'.

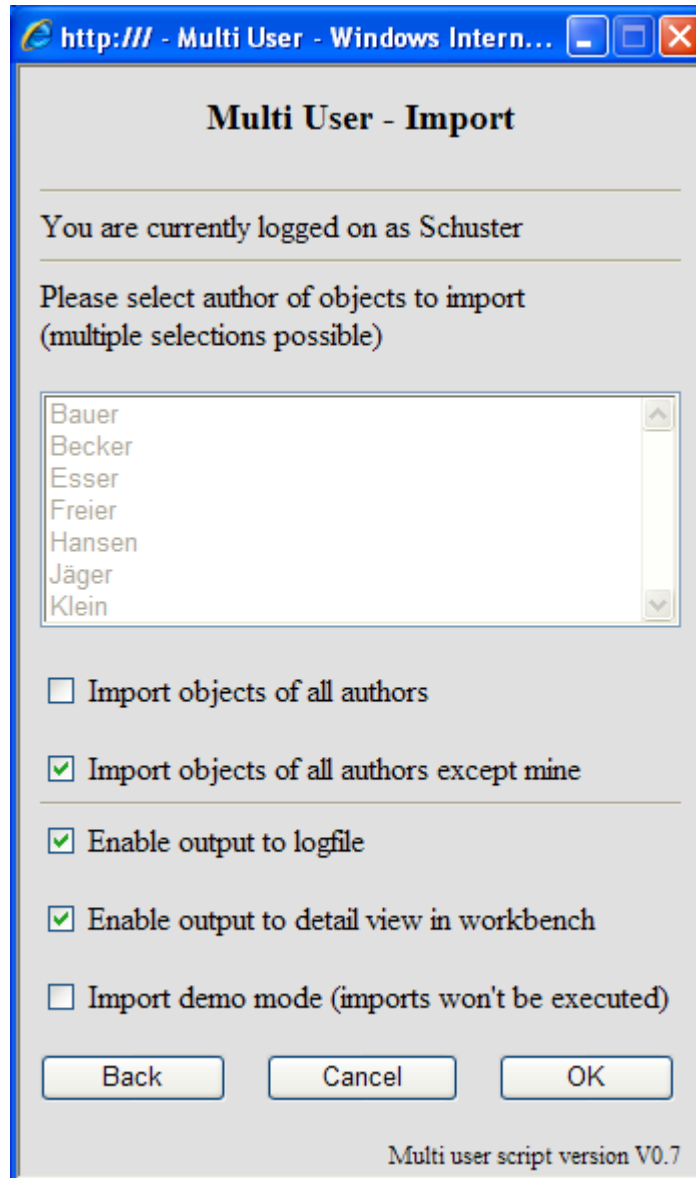
Level	Message
Information	Script execution started..
Information	Export of TO D435.Achse_2 [Author: Schuster; Version: 7]
Information	Export of TO D435.Achse_1 [Author: Schuster; Version: 10]
Information	Export of unit D445.GlobCtrl [Author: Schuster; Version: 31] canceled by user
Information	Logfile saved as D:\Repository_Log\LogFile_Schuster 2007-02-26 23-29-01.txt
Information	Script execution has been completed

MultiUser Scripts

3.3 Import from the external data filing to the project

After selecting the import function, you can make specific settings for the import. With pre-setting, all those objects are imported that are not assigned to the current user.

figure 3-5 import dialog box



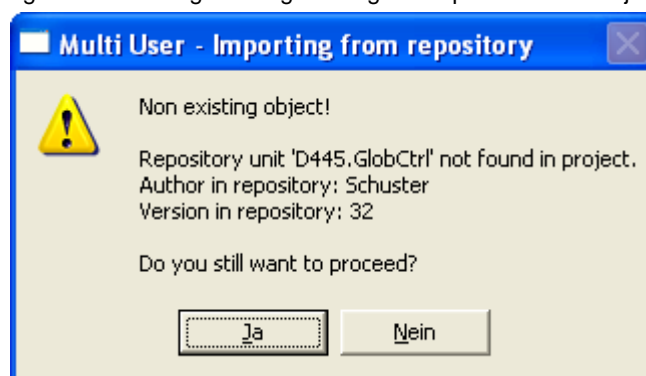
MultiUser Scripts

- Via the list field, you can select one or more authors whose objects are to be exported.
- **Import objects of all authors** imports all objects of all authors. The list field is deactivated with the selection of this option.
- **Import objects of all authors except mine** imports all objects of all authors except that of the current user. The list field is deactivated with the selection of this option.
- **Enable output to logfile** writes the executed imports to a special log file.
- **Enable output to detail view in workbench** writes the executed exports to the detail view in the Scout workbench.
- **Import demo mode** activates a demonstration mode where all actions are checked and recorded (logged) without executing a real import. By this, you can check in advance the executed actions.

With **OK**, the import is started and the following tests and actions are executed.

1. In the external data filing all supported objects are checked if the entered author corresponds to one of the selected ones.
2. If the author who is entered in the external data filing is identical to one of the selected ones, the system checks if the object already exists in the local project.
3. If the object does not exist in the local project, a warning message is indicated that has to be acknowledged. This warning message can be switched off via script setting (see chapter 2.2.4). In this case, the import is executed automatically.

figure 3-6 warning message during the import of a new object



4. If the object already exists in the local project, the author and the version identification is read from there and then compared with the information stored in the external data filing. If the author in the project differs from that one in the external data filing, a warning message will appear and has to be acknowledged.

MultiUser Scripts

If there is the same or a higher version identification in the project, there will be also a warning message to be acknowledged. Via script setting (see chapter 2.2.4), this warning message can be switched off in case that the versions are the same. In this case, the import is executed automatically.

With pre-setting, the executed imports are documented in a log file and in the detail view.

figure 3-7 display in the detail view during the import

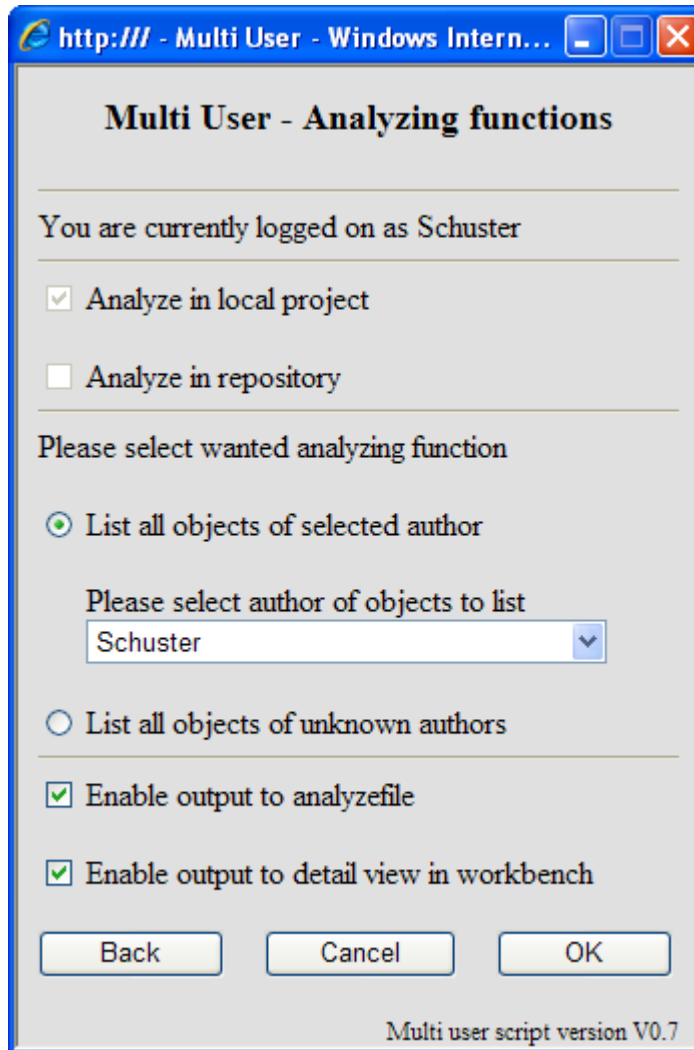
Level	Message
Information	Script execution started...
Information	Import of TO D435.Achse_1 [Author: Schuster; Version: 9]
Information	Import of TO D435.Achse_2 [Author: Schuster; Version: 6]
Information	Import of unit D445.GlobCtrl [Author: Schuster; Version: 32]
Information	Logfile saved as D:\Repository_Log\LogFile Schuster 2007-02-27 14-20-11.txt
Information	Script execution has been completed

MultiUser Scripts

3.4 Analyzing and testing functions

After selecting the analyzing and testing functions, the desired function is selected. With pre-setting, all those objects in the projects, that are assigned to the current user, are listed.

figure 3-8 dialog box for analyzing functions



MultiUser Scripts

- **Analyze in local project** executes the analysis in the local project (on the PG). This option is activated as a standard in the current version and cannot be deactivated.
- **Analyze in repository** executes the analysis in the external data filing. This option is deactivated as a standard in the current version and cannot be activated.
- **List all objects of selected author** lists all objects of an author. Via the drop down list, the author is selected whose objects are to be listed.
- **List all objects of unknown authors** lists all objects of which the authors have not been found in the user list.
- **Enable output to analyzefile** writes the results of the analysis to a special analyze file.
- **Enable output to detail view in workbench** writes the results of the analysis to a detail view in the Scout workbench.

OK starts the selected analysis.

figure 3-9 display when searching for unknown authors

Level	Message
Information	Script execution started..
Information	Found TO D435.Achse_2 [Author: Schuster; Version: 4]
Information	Found TO D435.Achse_1 [Author: Schuster; Version: 7]
Information	Found unit D445.GlobCtrl [Author: Schuster; Version: 32]
Information	Analyzefile saved as D:\Repository_Analyze\AnalyzeFile Schuster 2007-02-28 10-01-26.txt
Information	Script execution has been completed

MultiUser Scripts

4 External data filing

4.1 Data filing path

Exports that are executed via MultiUser scripts are stored in directories underneath the external data filing as XML exports. The filing is the same as the structure of the directory

<DataFilingPath>\[<DeviceName>\]<Objecttype>\<ObjectName>.xml

If the data filing path does not exist, it will be created automatically with the first export.

CAUTION Manipulations of files or data in the external data filing (e.g. delete or rename files via Windows Explorer) endanger the consistence of the data.

It may be that the MultiUser scripts does not work correctly or even not at all after a manipulation.

4.2 Log files

Log files that are created via MultiUser scripts are stored in a special directory underneath the external data filing as text file. Each action being logged in a file, creates a separate log file with user name including date and time stamp in the file name.

The filing of a log file is as follows

<DataFilingPath>_Log\<LogfileName>.txt

If the directory for log files does not exist, it will be created automatically.

4.3 Analysis files

Analysis files that are created via MultiUser scripts are stored in a special directory underneath the external data filing as text file. Each analysis being logged in a file, creates a separate analysis file with user name including date and time stamp in the file name

The filing of a log file is as follows

<DataFilingPath>_Analyze\<AnalysisFileName>.txt

If the directory for analysis files does not exist, it will be created automatically.

5 Boundary conditions

The current version supports a simultaneous working on separate TOs, units, libraries, watch tables and scripts.

NOTICE The scripts for MultiUser does not provide any functionality concerning the version administration.

Due to language-dependent components in the external data filing, all users have to work with the same language setting in SIMOTION Scout.

Furthermore, a change of the language is not possible if the scripts for MultiUser are used at the same time.

5.1 TOs

The TOs of all devices existing in the project are supported by the following TO types:

- Speed axis
- Positioning axis
- Synchronous axis
- Cam disk
- External encoder
- Sum block
- Stationary gear
- Formula
- Controller
- Sensor
- Temperature controller

The subordinated objects of a TO are automatically assigned to the basis TO, e.g. tracers, cams, cam tracks and scripts are assigned to the according axis. There is no interpretation of the properties **Author** and **Version** of the subordinated objects.

NOTICE During the import of an axis object, the synchronous relations are reset.

NOTE TOs on the project level (i.e. without connection to a device) are not supported in the current version.

5.2 Units

Units in the program folder of all devices existing in the project are supported. It does not matter which language was taken for the creation of the Unit (ST, MCC, LAD/CSF).

NOTE The current version does not support units in DCC.

NOTE The current version does not support units on project level (i.e. without connection to a device).

NOTE The current version does not support single units in libraries. These ones are handled via the library.

NOTE The merging of a unit that has been changed by several users at the same time is not supported.

5.3 Libraries

All libraries of the project are supported independent of their content.

The objects contained in the library are automatically assigned to the library, e.g. units. There is no interpretation of the properties **Author** and **Version** of the subordinated objects.

NOTE The current version does not support libraries in DCC.

NOTE The merging of a library unit that has been changed by several users at the same time is not supported.

MultiUser Scripts

5.4 Watch tables

All watch tables of the project are supported independent of their content.

5.5 Scripts

The scripts in the script folder of the project as well as the scripts on the device level are supported.

NOTE Scripts in the script folder of a TOs are included in the support for subordinated objects of a TO.

MultiUser Scripts

6 History

The version of this document is the same as the version of the MultiUser scripts.

Table 6-1 History

Version	Date	Alteration
V0.1	January 24, 2007	Vorabstand
V0.2	-none-	No documentation created
V0.3	-none-	No documentation created
V0.4	February 20, 2007	Supplementations to the documentation and functional extensions (Dialogs, Author selection, Demo mode, log files...)
V0.5	February 21, 2007	Functional extensions (Analysis functions)
V0.6	-none-	No documentation created
V0.7	February 28, 2007	Functional extensions (TOs, libraries, watch tables, scripts)