www.wallstreetsystems.com

Wall Street Systems – **Empowering** Treasury Trade and Settlement

Wallstreet Suite **Accounting Module** *System Administration Guide*

Version 7.3.14

First Edition (April 2011)

This edition applies to Wallstreet Suite version 7.3.14 and to all later releases and versions until indicated in new editions or Wall Street Systems communications. Make sure you are using the latest edition for the release level of the Wall Street Systems product.

Contents

Preface

Welcome to the Accounting Module System Administration Guide. This guide describes the system administrative tasks required for the Wallstreet Suite Accounting Module (ACM). The guide also includes a few tips, how to customize the default accounting behavior of the system.

This guide is intended for system administrators, who maintain and administer ACM, and the technical consultants implementing the system. They all should have experience with the following:

- The particular database and operating system platforms being used
- 3-tier architecture concepts
- Network configuration and maintenance
- TRM installation.

How to use this guide

How to rotate pages in Acrobat Reader

Some pages in this book have graphics rotated so that they can fit on the page. To see these normally on your computer monitor, use the **View - Rotate View** menu option in Acrobat Reader, or the equivalent keyboard shortcuts: Shift+Ctrl+Plus keys and Shift+Ctrl+Minus keys.

Methodology

There is no single "correct" way of administrating ACM. This guide proposes one way which can be used as a basis.

Using the script examples

Included in this guide are scripts that you can use during administration. These are examples, and therefore use some values that you must change to suit your administration.

IMPORTANT! UTF-8 encoding

From TRM version 7.1, all SQL scripts use UTF-8 encoding. When copying and pasting script examples, ensure that your text editor or terminal console supports UTF-8. Otherwise, characters that are outside the 7-bit ASCII character range can be inadvertently changed, and the database tools that you use may not warn you that a problem has occurred.

Copying and pasting script examples

You can copy script examples by opening this guide in Adobe Acrobat or Acrobat Reader, then pasting the clipboard contents into a plain text editor. However, if the script starts on one page and ends on another, you should ensure that you do not copy a block of text that goes over a page boundary; if you do this, you will find unwanted data being inserted into the script. To avoid this:

- In Acrobat, Select the Text tool (the mouse pointer becomes an "`I`" bar).
- In Acrobat, select and copy only as far as the end of the page.

- In the plain text editor, paste the text.
- In Acrobat, select and copy from the top of the next page.
- And so on.

Warning: Some code lines in this book may be too long to be displayed as a single line and as such they are displayed on more lines. When you paste such code into an editor, it will be pasted as **more** lines of code. Please remember to remove the unwanted line breaks.

Associated documents

Associated documents can be accessed from the **Help** menu of the Wallstreet Suite’s applications.

Chapter 1 System architecture

This chapter outlines the key concepts of ACM architecture. A technical understanding of the overall ACM architecture is a prerequisite for correct ACM administration and customizaition.

1.1 Overview

- ACM is based on a 3-tier architecture containing client applications, a J2EE application server, and a database server.
- ACM is installed as an add-on module to TRM:
  - ACM uses some core TRM components in all tiers (client, application server, and database).
  - TRM has to be fully installed and operational prior to ACM installation. The versions of TRM and ACM must match.

1.2 Application server

- J2EE based, allows deployment under Apache Tomcat or BEA Weblogic.
- Executes the business logic of the ACM application.
- Uses connection pooling in order to use database resources efficiently. Connections are open under a privileged application server user acmdbo

1.3 Database

- ACM builds and uses its specific database objects (tables, views, stored procedures etc.) in the same database as TRM.
- ACM accesses some TRM database objects (security model, shared static data entities, interfacing tables).

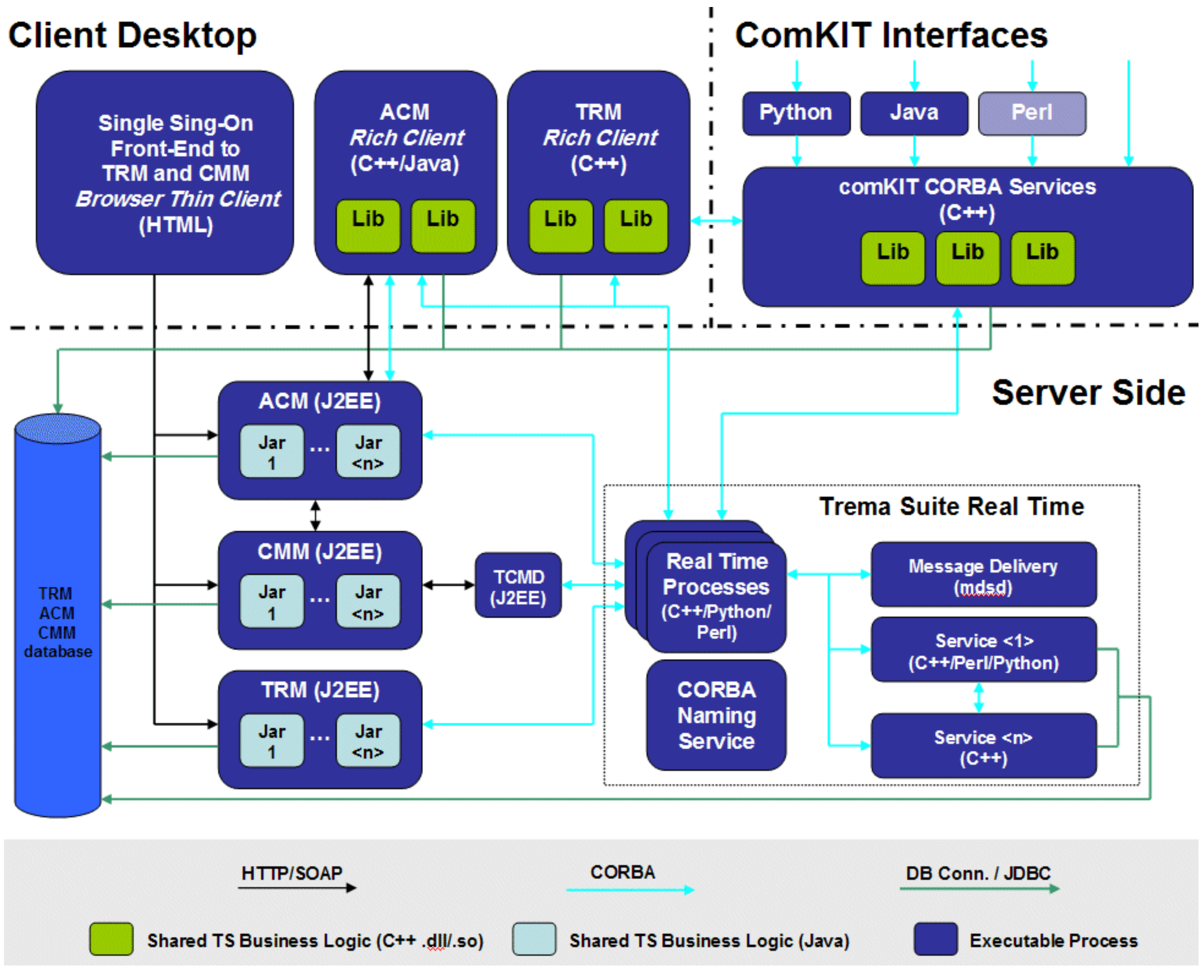
1.4 Shared components

These are components delivered with TRM and used by ACM:

- Static Data Framework – all ACM editors are implemented as SDF editors. The binaries from the TRM installation are executed when any ACM editor is launched.
- Report Generator.
- Naming Service.

- TRM Message Delivery System (mdsd).
- Subset of TRM database objects (tables, stored procedures, views) in the database.
- TRM database build script (extended by ACM-specific features).

The following picture shows the the overall Wallstreet Suite architecture:



The diagram depicts several important aspects, namely in the area of communication between the key components:

- ACM Client
  - ACM editors are **SDF** based and consequently they communicate with the ACM database via the standard TRM components (database connection).
  - ACM reports are based on the TRM Report Generator. ACM reports use the capability to invoke its data source as a **CORBA** service. ACM implements this CORBA server inside its application server. ACM registers with the naming service (shared with TRM) under the context `finance-kit/<FK_IDENT>` as service report_datasource. `<FK_IDENT>` represents the real value,for example `PROD` or `TEST1`: `FK_IDENT` is a mandatory TRM environment variable.
  - ACM Entry Manager communicates with ACM Application Server via the **HTTP** protocol. The ACM Application Server business logic is implemented as servlets running inside the J2EE container.
- ACM Application Server
  - ACM Application Server is registered as a listener to mdsd events in order to be notified about static data changes in TRM. Therefore it is essential that mdsd is operational for applications to behave correctly.

- ACM Application Server communicates with CMM via the **Web Services** interface in order to exchange accounting data. Consequently, on the network level, the HTTP protocol must be enabled.
- ACM Application Server communicates with the TRM / ACM database via pooled **JDBC** connections in order to use DB resources efficiently.

Chapter 2 Hardware requirements

ACM installation requires an installed and fully operational TRM database. The TRM database must be exactly the same version as ACM (e.g. ACM version 7.1.0.7 requires TRM version 7.1.0.7)

ACM requires the same hardware as TRM (see the *WSS Database Setup Guide)*. In addition, ACM modifies and adds the following requirements:

For the ACM database server, a computer equipped with the following:

– TRM database server requirements

– An extra **500 MB** of free disk space available need **per each 100,000 accounting entries** in the system.

Oracle Database Server tablespace sizes:

Tablespace	Used by	Minimum size
USERS	ACM tables	TRM size + 250MB per 100,000 entries
INDX	ACM indexes	TRM size + 250MB per 100,000 entries
WORK	ACM temporary tables	TRM size + 200MB

For the ACM Server:

– TRM Real-time Server requirements

– An extra **500 MB** of free disk space.

For the ACM Client:

– TRM client requirements

– An extra 2**00 MB** of free disk space

– the network must be configured to allow the ACM Client computers communicating with the ACM Application Server via the **HTTP protocol** on the ports specified in the ACM Application Server configuration.

Note: ACM Application Server and ACM database can be installed either on a single machine or on two dedicated machines if higher performance is required.

Chapter 3 Database maintenance

Correct database maintenance is one of the key assumptions of a correct system usage and operation. You can read all about it in the *WSS Database Setup Guide.* That guide covers all three supported databases: Oracle, MSSQL, and Sybase.

3.1 Building the database

ACM is installed into the same database (and into the same database schema for Oracle) as TRM. Once the TRM database is installed, you simply add the ACM objects to those created for TRM.

For building the database objects, ACM uses a wrapper script acmbuild.pl around the TRM build script (build.pl); this acm scripts adds some functionality to the TRM build script (for instance, accounting perspective checking, extended validations, referencing integrity checking, etc.). All that functionality is added into the generated stored procedures used by ACM editors, e.g. they are added into the commit, remove, revert and search procedures.

The command line options for the ACM build are the same as for the TRM build script, i.e. all the build script options (-U, -P, -S, etc.) as described in the appendix of the *WSS Database Setup Guide* are valid for the ACM build too.

3.1.1 Oracle

This section describes the ACM database build process for Oracle. Follow these steps:

1. Check the connection to the Oracle database (replace `<password>` with the *dbo* user's password)

   Unix:

   ```
   sqlplus dbo/<password>@$FK_DB_SERVER
   ```

   Windows:

   ```
   sqlplus dbo/<password>@%FK_DB_SERVER%
   ```

2. When successfully connected, type **`exit`** and press ENTER

   If the connection fails, check that:

   – `tnsnames.ora` contains a proper connection specification.

   – Oracle client is properly installed.

   – The database server you are connecting to is running.

   – There is no network error.

3. Switch to the appropriate directory

   Unix

   ```
   cd $FK_HOME/share/oracle
   ```

   Windows

   ```
   cd /d %FK_HOME%\share\oracle
   ```

4. Build ACM database objects using the Perl script `acmbuild.pl` provided:

Unix:

```
./acmbuild -D $FK_DB_SERVER -U dbo -P <password> -t \* -V \* -p \* -g \* -u -w -v
```

Windows:

```
perl acmbuild.pl -D %FK_DB_SERVER% -U dbo -P <password> -t * -V * -p * -g * -u -w
-v
```

Warning: The -w option enables the database for the Static Data Manager (SDM). You should include the -w option **only** if you are going to install SDM. Please note that it is mandatory to install SDM for ACM when SDM has been installed for TRM.

5. Start an SQLPLUS session

Unix

```
sqlplus dbo/<password>@$FK_DB_SERVER
```

Windows

```
sqlplus dbo/<password>@%FK_DB_SERVER%
```

6. Recompile the invalid dbo's stored objects (run inside the SQLPLUS session):

Unix

```
START $FK_HOME/share/oracle/migration/fast_recomp.sql
```

Windows

```
START %FK_HOME%\share\oracle\migration\fast_recomp.sql
```

7. Ensure that there are no invalid objects for the user dbo (run inside the SQLPLUS session)

```
select object_type, count(object_name) qty_invalid from user_objects where status
= 'INVALID' group by object_type;
```

8. Make sure that the last database build version finished is the same as the version of ACM you installed:

```
select max(SPKGVersion) from ACMVersion where SParameters = 'finished';
```

9. Exit the SQLPLUS session by typing exit and pressing the ENTER key.

Notes

- It is necessary to escape the asterisks (*) with the backslashes for non-Windows platforms. Replace `<password>` with the *dbo* user password in the command line examples.
- Owing to a bug in Oracle database server version 9.2.0.6 and some later versions, views using ANSI joins become invalid when compiled under a user other than the view's owner. If you use a recompilation script run under the SYS user, then ACM views become invalid. Recompile ACM views under the DBO user only. For the bug reference, see https://metalink.oracle.com/metalink/plsql/showdoc?db=Bug&id=3466980
- Despite the fact that ACM Application Server uses the `ORACLE_SID` environment variable to populate the `ACM_JDBC_URL` environment variable, the `sqlplus` command must be run with a database name according to the local `tnsnames.ora` configuration file. Typically `ORACLE_SID` contains the Oracle database SID while `FK_DB_SERVER` environment variable contains a TNS alias from the local `tnsnames.ora`.

3.1.2 Microsoft SQL Server

This section describes the ACM database build process for MSSQL. Follow these steps:

1. Check the connection to the MSSQL database (replace <password> with the *fk* user's password)

   ```
   isql -S %FK_DB_SERVER% -d %FK_DATABASE% -U fk -P <password>
   ```

2. When successfully connected, type exit and press the ENTER key.

   If the connection fails, check that

   – The MSSQL client is properly installed.

   – The database server you are connecting to is properly registered.

   – The database server you are connecting to is running.

   – You are using the MSSQL `isql` tool and not the Sybase `isql` tool. The MSSQL `isql` utility path must be on system `PATH` prior to any path containing the Sybase `isql` utility.

   – There is no network error.

3. Switch to the appropriate directory

   ```
   cd /d %FK_HOME%\share\mssql
   ```

4. Build ACM database objects using the Perl script acmbuild.pl provided:

   ```
   perl acmbuild.pl -S %FK_DB_SERVER% -D %FK_DATABASE% -U dbo -P <password> -t * -V
   * -p * -g * -u -w -v
   ```

Warning: The -w option enables the database for the Static Data Manager (SDM). You should include the -w option **only** if you are going to install SDM. Please note that it is mandatory to install SDM for ACM when SDM has been installed for TRM.

5. Start an **`isql`** session:

   ```
   isql -S %FK_DB_SERVER% -d %FK_DATABASE% -U fk -P <password>
   ```

6. Make sure that the last database build version finished is the same as the version of ACM you installed:

   ```
   select max(SPKGVersion) from ACMVersion where SParameters = 'finished'
   go
   ```

7. Exit the isql session by typing `exit` and pressing the ENTER key.

3.1.3 Sybase

This section describes the ACM database build process for Sybase. Follow these steps:

1. Check the connection to the Sybase database (replace <password> with the *fk* user's password)

   ```
   isql -S %FK_DB_SERVER% -D %FK_DATABASE% -U fk -P <password>
   ```

2. When successfully connected, type exit and press the ENTER key.

   If the connection fails, check that

   – The Sybase client is properly installed.

   – The database server you are connecting to is properly registered.

   – The database server you are connecting to is running.

   – You are using the Sybase `isql` tool and not the MSSQL `isql` tool. The Sybase `isql` utility path must be on system `PATH` prior to any path containing the MSSQL `isql` utility.

   – There is no network error.

3. Switch to the appropriate directory

   ```
   cd /d %FK_HOME%\share\sybase
   ```

4. Build ACM database objects using the Perl script acmbuild.pl provided:

   ```
   perl acmbuild.pl -S %FK_DB_SERVER% -D %FK_DATABASE% -U dbo -P <password> -t \* -V
   \* -p \* -g \* -u -w -v
   ```

Warning: The -w option enables the database for the Static Data Manager (SDM). You should include the -w option **only** if you are going to install SDM. Please note that it is mandatory to install SDM for ACM when SDM has been installed for TRM.

5. Start an `isql` session:

   ```
   isql -S %FK_DB_SERVER% -D %FK_DATABASE% -U fk -P <password>
   ```

6. Make sure that the last database build version finished is the same as the version of ACM you installed:

   ```
   select max(SPKGVersion) from ACMVersion where SParameters = 'finished'
   go
   ```

7. Exit the isql session by typing `exit` and pressing the ENTER key.

3.2 Manual database upgrade

ACM upgrade depends on a version from which you start the upgrade.

- If your version, from which you start the upgrade, is 7.2.2.1 or later, follow **just** the guidelines described in chapter *3.2.2 Upgrading from 7.2.2.1* on page 22.
- If your version, from which you start the upgrade, is older than 7.2.2.1, you have to upgrade to 7.2.2.1 version **first** as described in *3.2.1 Upgrading from version before 7.2.2.1* on page 20 **and then** you must continue the upgrade up to the just installing version according the guidelines in chapter *3.2.2 Upgrading from 7.2.2.1* on page 22.

3.2.1 Upgrading from version before 7.2.2.1

All the upgrade tasks including the database upgrade should be done via the Suite Installer. However, if there is a reason for it, you can perform the database upgrade manually.

ACM provides an automatic upgrade script to upgrade the database structures of ACM.

Before running the upgrade script, make sure that you have evaluated the TRM/ACM environment.

To run the script run the following commands (replace `<db_platform>` with one of the following: `oracle`, `mssql`, `sybase`):

Unix:

```
cd $FK_HOME/share/<db_platform>
```

– **Oracle**:

  ```
  ./acmupgrade.pl -D <database_name> -U <user_name> -P <password> -w
  ```

– **Sybase**:

  ```
  ./acmupgrade.pl -S <server_name> -D <database_name> -U <user_name> -P
  <password> -w
  ```

Windows:

```
cd %FK_HOME%\share\<db_platform>
```

– **Oracle**

  ```
  perl acmupgrade.pl -D <database_name> -U <user_name> -P <password> -w
  ```

– **MSSQL**, **Sybase**

  ```
  perl acmupgrade.pl –S <server_name> -D <database_name> -U <user_name> -P
  <password> -w
  ```

Warning: The -w option enables the database for the Static Data Manager (SDM). You should include the -w option **only** if you are upgrading an installation with SDM installed.

The `acmupgrade.pl` options are as follows:

-S <server_name>	Database server name. This option is used for MSSQL and Sybase only.
-D <database_name>	Database name (MSSQL, Sybase) or Oracle TNS alias set up on the local machine
-U <user_name>	Username of the database objects owner (typically "DBO" for Oracle, "fk" for MSSQL and Sybase)
-P <password>	User password
-w	Create or upgrade the SDM data structures. You should include the -w option **only** if you are upgrading an installation with SDM installed.
-s <source>	Source version of ACM, e.g. "7.1.1.0" (without the double quotes). Optional for sources of version 7.1.1.0 or above. It is recommended to omit the parameter when performing standard upgrade to the latest version. The upgrade script is able to determine the installed ACM version reading the version info from the database.
-d <destination>	Destination version of ACM, e.g. "7.1.2.0" (without the double quotes). Optional, defaulted to the version of the package. It is recommended to omit the parameter when performing standard upgrade to the latest version. **Note:** Destination version is an optional parameter. If it is not specified, the upgrade script executes all the steps up to the version of the package. If upgrading from 7.1.1.0 version or any higher it is recommended to not specify the source version and to let the upgrade script automatically determine the source. **Note:** As of 7.2.2.2 version the only accepted destination version is version 7.2.2.1, even you install a later version (e.g. 7.2.2.3). However, you can still run the upgrade without specifying this parameter.
-Y	This option ensures that all the prompted questions raised during the upgrade script execution are answered automatically Yes. This option can be used, for instance, if you want to perform an unattended upgrade, e.g. as a part of the nightly tasks. Read more below in the text. **Warning:** It is recommended to check the log, whether the automated answer Yes was not set for some unexpected question.

The script detects the actual version of ACM installed, and does not allow an upgrade to start from a version higher than the detected version. If the script cannot detect the version of ACM installed, it reports it as "UNKNOWN" and asks for confirmation.

When the upgrade of the database structures finishes successfully, the following prompt (or similar) is displayed:

```
Do you want to proceed (Do you want to run build of procedures, views, indexes and
grants for ACM version 7.1.2.0? )?
```

Warning: If you run the upgrade script with -Y option then this question will be answered as Yes and you will not have an option to rebuild the objects manually.

If you want the upgrade script to rebuild the procedures, views, indexes and grants simply type `y` and press the ENTER key.

If you want to rebuild the objects manually, type `n` and press the ENTER key: the upgrade script will terminate.

Rebuilding procedures, views, indexes and grants is **mandatory** to complete the ACM upgrade process. If you do not rebuild the mentioned objects, the upgrade cannot be considered finished.

If you do not want the upgrade script to rebuild the objects, you can rebuild it manually using the ACM build script:

Unix:

```
./acmbuild (-S ...) -D ... -U ... -P ... -V \* -i \* -p \* -g \* -u -w -v
```

Windows:

```
perl acmbuild.pl (-S ...) -D ... -U ... -P ... -V * -i * -p * -g * -u -w -v
```

For further information about the acmbuild script, please refer to the section on building the database in section *3.1 Building the database* on page 17.

Warning: The -w option enables the database for the Static Data Manager (SDM). You should include the -w option **only** if you are upgrading an installation with SDM installed.

Note: When you run the upgrade for MS SQL Server, make sure that the `PATH` environment variable is set so that it contains the directory where the isql or bcp utility is present (usually under `C:\Program Files\Microsoft SQL Server\80\Tools\Binn`). Optionally you can set the `MSSQL_HOME` environment variable to the MS SQL Server installation directory (`C:\Program Files\Microsoft SQL Server`) – the upgrade script then determines the correct location.

3.2.2 Upgrading from 7.2.2.1

Warning: This upgrade procedure is very different from the previous one. It uses ACM build **-X** option which aligns database tables by adding, altering, renaming, and **dropping** columns so that a table in the database has the same structure as in the perl description of the installing package. So if you have a column that was added by a plain SQL command (for example, a CSD), it will be lost (including its data). Therefore you must first modify the perl description so that it reflects your CSD.

1. Run the following acmbuild command, and if Static Data Management (SDM) is already installed (and **only** if it is already installed), add the option **-w**:

Unix:

```
./acmbuild (-S ...) -D ... -U ... -P ... -X -v -e
```

Windows:

```
perl acmbuild.pl (-S ...) -D ... -U ... -P ... -X -v -e
```

2. Rebuild all procedures, triggers, views, indexes, permissions, default logins, groups, and users:

Unix:

```
./acmbuild (-S ...) -D ... -U ... -P ... -V \* -i \* -p \* -g \* -u -v
```

Windows:

```
perl acmbuild.pl (-S ...) -D ... -U ... -P ... -V * -i * -p * -g * -u -v
```

Note: Option -u creates default logins, groups, and users that do not yet exist in the database. It does not change existing ones.

3.3 Enabling SDM on a non-SDM system

This chapter describes how to enable Static Data Management (SDM) for ACM. Note that SDM is described in the TRM System Administration Guide.

Warning: This section applies only to Wallstreet Suite version 7.2.1.0 or later. Do not attempt this procedure on a system before version 7.2.1.0.

If you are enabling SDM on ACM, then:

- You should complete all ACM upgrade steps before enabling SDM; in other words, you should have an upgraded and fully-functional Wallstreet Suite before following the steps below.
- You should ensure that ACM is configured without SDM; in other words, the steps below must not be applied on ACM with enabled/installed SDM.
- You should ensure that SDM is already enabled on TRM.

The process of enabling SDM on ACM consists from two steps.

1. Upgrade non SDM system to SDM system:

Unix:

```
./acmbuild (-S ...) -D ... -U ... -P ... -X -v -e -w
```

Windows:

```
perl acmbuild.pl (-S ...) -D ... -U ... -P ... -X -v -e -w
```

2. Rebuild all procedures, triggers, views, indexes, permissions, default logins, groups, and users for SDM:

Unix:

```
./acmbuild (-S ...) -D ... -U ... -P ... -V \* -i \* -p \* -g \* -u -v -w
```

Windows:

```
perl acmbuild.pl (-S ...) -D ... -U ... -P ... -V * -i * -p * -g * -u -v -w
```

Note: For all the steps in this chapter the option **-w** is not optional, but **mandatory**.

3.4 Performance tuning

Due to the huge processed and maintained data volumes the accounting processing is prone to the performance issues. For their solving see the instructions in the *WSS Database Setup Guide.* For instance, for the Oracle performance issue, see the chapter about computing statistics.

Chapter 4 ACM server configuration and maintenance

This chapter describes the tasks related the ACM server configuration and maintenance.

4.1 ORB services configuration

Most of the ACM reports is based on a CORBA communication. Sometimes it is necessary to assign a specific IP and port used by the ACM server for the CORBA communication with the Report Generator (FKReport.exe).

ACM Server registers own reference called "report_datasource" into the Naming Service dedicated for the running environment. Report Generator (client) resolves IOR for "finance-kit/<FK_IDENT>/report_datasource" from the Naming Service and uses this references for the query of report parameters and report processing.

From a technical point of view there is Omni Names ORB on client (FKReport); one of the following ORB implementations on ACM Server is used:

- JACORB – since 7220, tomcat, weblogic
- OpenORB – before 7220, tomcat, weblogic
- IBM ORB – all versions, websphere

ACM internally creates two server ORBs, one for the report generator (registered in the naming) and one for MDSD (to get notifications about static data changes). Typically, only the report generator communication will go through firewall as ACM and MDSD are collocated (not separated by the firewall).

4.1.1 Client side

For setting specific ORB parameters on the client side, start fkreport with –ORB* parameter. For seeing the full list of all available parameters open shell and type:

```
>mdsd -ORBhelp
```

If you need to explicitly specify an IP and port for fkreport.exe you can achieve this by using the following command line:

```
>fkreport -ORBendPoint giop:tcp:<host>:<port>
```

For details see the OmniOrb documentation available at http://omniorb.sourceforge.net/docs.html.

4.1.2 Server side

As explained above, there are a few services running inside the ACM Server and using ORB communication. For each such service you can set specific ORB parameters using system variables with a proper prefix.

For MDSD Listener service running inside the ACM Server use parameters with prefix "MDSD_ORB". For Report Service inside ACM server use variables with prefix "RG_ORB".

If you need to explicitly specify an IP and/or port for ACM Report Service (report_datasource) set following environment variable:

```
RG_ORBOAPort=12345.
```

where OAPort is a regular parameter on JacORB. OAPort parameter specifies a port used by the report_datasource service registered by the ACM Server.

For detail list of parameters available in JacORB see JacORB the website http://www.jacorb.org.

Specifically, you may need to read an article *„How can I make the server use a specific port number or IP address (in case of multi-homed hosts)?"* (http://www.jacorb.org/FAQ.html#Help.E)

Warning: Properties with prefixes MDSD_ORB and RG_ORB are taken into account just on 7.2.1.6a and higher 7.2.1.x packages and on 7.2.2.3 and higher 7.2.2.x packages. Older versions of ACM do not support these prefixes for ORB services running inside the ACM Server. Therefore it is no possibility to set these ORB properties selectively for one service only in those old ACM versions. In other words, in old ACM versions that port cannot be effectively assigned by this way (just the first service takes that port and others fails).

If you are looking for parameters on OpenORB which were used before the 7220 version see

http://openorb.sourceforge.net/docs/1.3.0/OpenORB/doc/orb.html#N1036B.

Note: For a permanent change of the environment system variables you have to modify config.peoperties in your site/components/acm/ and reconfigure the environment. For details read the Suite Installation Guide.

4.1.3 Specific configuration use cases

4.1.3.1 BiDirectional setting for ACM Reports

ACM Reports are handled on the client side by FKReport.exe application which communicates with the ACM Server. Sometimes there is a necessity to put a firewall between a client machine and the server. As explained earlier the FKReport.exe application uses OMNI ORB implementation and ACM Server starting with 722x uses JACORB. This chapter is about specifying a single port on the ACM Server for handling requests from the FKReport clients.

The setup described below ensures that the communication between the FKReprt and the ACM server uses a single port on the ACM server side (the one specified in the RG_ORBOAPort property). If you have multiple network interfaces on a machine where the ACM Server is running, you may also need to read *4.1.3.2 ACM Server IP setting on machine with multiple network interfaces* on page 27.

- **Client configuration**

  On client (FKReport) you have to set in the orb.conf file (accessible via the environment variable OMNIORB_CONFIG) a property *clientTransportRule* to allow ***bidir*** and ***offerBiDirectionalGIOP*** as follows:

```
clientTransportRule    = 172.24.64.9          unix,tcp,bidir

                       = *                    ssl,tcp,bidir

offerBiDirectionalGIOP = 1
```

- **Server configuration**

  a. Stop all Weblogic/Tomcat services (ACM, WSS Web Suite) and Weblogic Administration Server (via PMM)

  b. Modify the file /sharedconf/components/appservers/appservers.weblogic.properties (if using Weblogic) or /sharedconf/components/appservers/appservers.tomcat.properties (if using Tomcat) so you have the following two lines (the first line is shown on two rows below) there:

```
ts.appserver.t3.jvmarg.9=-DRG_ORBorg.omg.PortableInterceptor.ORBInitializerCl
ass.bidir_init=org.jacorb.orb.giop.BiDirConnectionInitializer
```

```
ts.appserver.t3.jvmarg.10=-DRG_ORBOAPort=22226
```

You may use different port number (we use 22226); it is the one configured on the firewall.

c. Reconfigure your environment according to an explanation in the WSS Installation Guide, for instance:

```
...SI_Manager>setup
```

```
...SI_Manager>%si% reconfigure -p ts-weblogic -i c:\ws-suite\v7216 -e dev71 -n 1
```

or

```
[...SI_Manager>%si% reconfigure -p ts-tomcat -i c:\ws-suite\v7216 -e dev71 -n 1]
```

d. Start all Weblogic/Tomcat services (ACM, WSS Web Suite) and Weblogic Administration Server (via PMM).

4.1.3.2 ACM Server IP setting on machine with multiple network interfaces

On a multi-homed host (a host having multiple IP addresses configured) you might need to specify one IP which should be used for Corba services published to the naming service by the ACM server.

A service responsible for the ACM reports is called "report_datasource". As mentioned earlier you can configure ORB properties for this service using the property prefix "RG_ORB".

The following procedure specifies report_datasource's IP address published in IOR; a configuration is done on the server side only (not on the client side):

- **Server configuration**

a. Stop all Weblogic/Tomcat services (ACM, WSS Web Suite) and Weblogic Administration Server (via PMM)

b. Modify the file /sharedconf/components/appservers/appservers.weblogic.properties (if using Weblogic) or /sharedconf/components/appservers/appservers.tomcat.properties (if using Tomcast); add next free number for the `ts.appserver.t3.jvmarg`, for instance:

```
ts.appserver.t3.jvmarg.11=-DRG_ORBOAIAddr=123.124.125.126
```

i.e. the previous used `ts.appserver.t3.jvmarg` was `ts.appserver.t3.jvmarg10` since the sequence must unbroken and always increased by 1. The specified IP address (in our example `123.124.125.126`) is one of the server's IP addresses you want to use.

c. Reconfigure your environment according to an explanation in the WSS Installation Guide, for instance:

```
...SI_Manager>setup
```

```
...SI_Manager>%si% reconfigure -p ts-weblogic -i c:\ws-suite\v7216 -e dev71 -n 1
```

or

```
[...SI_Manager>%si% reconfigure -p ts-tomcat -i c:\ws-suite\v7216 -e dev71 -n 1]
```

d. Start all Weblogic/Tomcat services (ACM, WSS Web Suite) and Weblogic Administration Server (via PMM).

4.2 Memory settings

ACM Server is running as a java process on a selected operating system. Sun's JVM is used normally. There are some restrictions put by a hardware and the operating system on a maximum size of a memory which can be used by each process. If you use 64bit operating system and you use

–d64 parameter for a java process, you can set a heap memory size according your needs. However, on 32-bit data model you are limited by various factors. For details see the following links:

1. Sun's JVM Tuning

   http://java.sun.com/performance/reference/whitepapers/tuning.html

   (See part called *"Heap Sizing" for explanation of maximum heap size*)

2. Memory Limits for Windows Releases

   http://msdn.microsoft.com/en-us/library/aa366778.aspx#physical_memory_limits_windows_xp

3. Why can´t I get a larger heap with the 32-bit JVM

   http://java.sun.com/docs/hotspot/HotSpotFAQ.html#gc_heap_32bit

Each Java process can be configured using the –X options on a command line:

- **-Xms<size>**

  Sets initial Java heap size

- **-Xmx<size>**

  Sets maximum Java heap size

- **-Xss<size>**

  Sets java thread stack size

ACM server is provided with the following default settings in appservers.tomcat.properties (or appservers.weblogic.properties):

```
ts.appserver.t3.jvmarg.1=-Xmx1300m

ts.appserver.t3.jvmarg.2=-Xms128m
```

You can change –Xmx to a value according your needs and limits taking into account your operating system or hardware restrictions. Once you change the mentioned properties file you have to follow instructions for reconfiguring the environment (at least ts-tomcat or ts-weblogic) as described in the WSS installation Guide.

Note: Wall Street Systems experienced that for the 32-bit java a maximum memory heap available for the ACM Server is around a value -Xmx1440m (on condition that you have enough physical resources on that machine).
If your limit is set nearly to value 1400m and you get OutOfMemory exceptions in logs regularly, you should consider the 64-bit option.

4.3 Logging

4.3.1 Server log file size

The server log files size is 10 times 100 MB for the standard logging and 3 times 100 MB for the ERP export logging. A total size of the ACM logs is 1.3 GB.

4.3.2 Server log file appender

The ACM server contains a rolling file Log4J appander collecting the runtime performance statistics. They statistics are written into a CSV file located in ACM server's log directory. Once the file reaches its limit of 500 MB, it is renamed, appended by an index and new file is created. The system creates up to 10 files containing historical performance date. After reaching the limit of 10 files, the oldest one is deleted and a new one is created. When dedicating a space for the ACM server, you must reserve 5 GB for these files. The statistics can be used by Wall Street Systems engineers for investigating potential performance issues.

4.4 Connection Pool setup

To be completed.

4.5 Time zones setup

Wallstreet Suite is designed to run in one time zone. All parts of the system should use the same time zone to avoid unexpected conversions of a date. Default installation forces GMT setting to server as well as client. Such configuration is suitable for all installations regardless of a real time zone in which the system runs. It is strongly recommended to not touch the default setting (GMT) since the customization of the time zone may cause serious troubles (especially in accounting).

An example of possible troubles related to time zone setting is the communication between WebSuite and ACM or usage of the Accounting Manager. If you get into a situation where, for instance, Accounting Manager has GMT and ACM Server is defined in MST, the date of 2/1/08 0:00 will be interpreted on ACM Server as 1/1/08 17:00 and all subsequent actions will be done using 1/1/08 17:00.

Warning: ACM Server version 7.2.2.12 and higher will not start if the user.timezone property is not set to GMT!

Chapter 5 ACM client configuration and maintenance

This chapter describes the tasks related the ACM client configuration and maintenance.

5.1 Verifying ACM menu

To check that the ACM menu was successfully added to the TRM Client menu during the ACM package installation:

1. Start the TRM Application Manager.
2. Check that the **Accounting Module** submenu appears at the root level of the Application Manager menu.

The menu structure of the first and the second level is as follows:

- Accounting Module
- Set-Up Management
- Advanced Set-Up Management
- Process Management
- Accounting Reports
- Set-Up Reports
- Verification Reports
- ERP Interface.

5.2 Setting Accounting Manager directory

Accounting Manager needs Write access to some directories to store its configuration and create temp files. By default, it creates these files in the home directory of the logged-in user in its subdirectory Application Data/Accounting Manager. This directory is also used for writing logs.

If necessary, the directory can be changed by changing the file `%ACM_HOME%/share/java/acm/client/AccountingManager.ini`. Change line `-configuration @user.home/`Application Data/Accounting Manager `to -configuration <your dir>`.

You also need to change the configuration of log4j (file log4j.xml in the same directory) to output the log somewhere else.

If you are not successful with starting Accounting Manager, you can enable console output. To enable it, add to AccountingManager.ini as first line option `-consoleLog`.

5.3 SSL setup

There are two security levels when working with SSL. The level depends on the server setup. The description of all properties needed for the set up can be found in *Appendix D ACM environment variables* on page 163. For more details about SSL see *Appendix E Secure Socket Layer (SSL)* on page 167.

There is a demo setup which can be found in the `%ACM_HOME%/share/java/acm/client. This setup is` ready for a usage with Tomcat. See the *README.txt* file for other details.

Warning: Do not use the demo certificates in any production environment!

5.3.1 SSL without client authentication

The simpler way of SSL communication is based only on server certificates. To make this work, set ACM_PROTOCOL and truststore location in ACM_TRUSTSTORE.

5.3.2 SSL with client authentication

This more advanced setup provides a higher security. In order to make this functional, the client needs the keystore having the key-pair and the server needs client's public certificate in its truststore.

ACM_PROTOCOL and ACM_TRUSTSTORE are essential, for the client authentication additional properties ACM_KEYSTORE and ACM_KEYSTORE_PASS are to be set on the client side to make things work.

5.3.3 Example

To run Accounting over SSL, first it is needed to setup protocol.

```
SET ACM_PROTOCOL=https
```

Ensure the port is a https port, 8443 for example.

Let's assume we get the *server.crt* certificate that contains server public certificate. Move to the Accounting Manager directory (`%ACM_HOME%/share/java/acm/client`). Then create a truststore using the keytool (best to have it on the path):

```
>keytool -keystore truststore.jks -import -alias server -trustcacerts -file
server.crt
Enter keystore password:  server
Owner: CN=ekit, OU=esolution, O=trema, L=sophia-antipolis, ST=cote d'azur, C=fr
Issuer: CN=ekit, OU=esolution, O=trema, L=sophia-antipolis, ST=cote d'azur, C=fr
Serial number: 0
Valid from: Mon Jan 28 14:57:04 CET 2002 until: Tue Jan 28 14:57:04 CET 2003
Certificate fingerprints:
MD5:  4B:A9:7D:5C:3C:75:AA:A6:A8:2A:EE:DC:64:ED:0B:80
SHA1: 4B:18:C2:7C:85:EC:EB:1F:3F:4C:E4:E3:9B:8D:5B:E7:F7:DE:AB:C0
Trust this certificate? [no]:  yes
Certificate was added to keystore
```

Then set the property

```
SET ACM_TRUSTSTORE=truststore.jks
```

That's it. Now the Accounting Manager will communicate with the Accounting Server over SSL.

For more advanced setup (the server has the client authentication enabled) we go further. Let's create a key-pair for the Accounting Manager.

```
>keytool -keystore AccountingManager.jks -genkey -alias AccountingManager
Enter keystore password:  AccountingManager
```

```
What is your first and last name?
[Unknown]:  Accounting Manager
What is the name of your organizational unit?
[Unknown]:
What is the name of your organization?
[Unknown]:
What is the name of your City or Locality?
[Unknown]:
What is the name of your State or Province?
[Unknown]:
What is the two-letter country code for this unit?
[Unknown]:
Is CN=Accounting Manager, OU=Unknown, O=Unknown, L=Unknown, ST=Unknown, C=Unknown
correct?
[no]:  yes
Enter key password for <AccountingManager>
(RETURN if same as keystore password):
```

Hit <ENTER>. It is essential for Accounting Manager to have the same password for keystore itself and the key-pair.

Now it's time to sign the key-pair. The easiest way is to create so called self-signed certificate. There is no security problem in it.

```
>keytool -keystore AccountingManager.jks -selfcert -alias AccountingManager
Enter keystore password:  AccountingManager
```

Good habit is to check the certificates once they are created to ensure that everything is right:

```
>keytool -keystore AccountingManager.jks -list
Enter keystore password:  AccountingManager
Keystore type: jks
Keystore provider: SUN
Your keystore contains 1 entry
accountingmanager, 13.2.2007, keyEntry,
Certificate fingerprint (MD5): 7A:B2:53:D0:CC:89:4C:6B:F9:55:0F:65:CD:30:94:5B
```

Now it's time to setup the last system properties like this:

```
SET ACM_KEYSTORE=AccountingManager.jks
SET ACM_KEYSTORE_PASS=AccountingManager
```

5.4 ORB configuration

In some cases it is necessary to define the ORB parameters on the client side. For details see *4.1 ORB services configuration* on page 25.

Chapter 6 Using accounting administrator tools

6.1 Accounting dynamic data verification report

6.1.1 Introduction

Consistency of dynamic accounting data is a critical assumption for correct working of ACM. *Accounting Dynamic Data Verification Report* is a tool executing a few basic verification checks performed on accounting dynamic data like vouchers, entries and account balances. It is recommended to run the report as the last step of the following three actions:

1. Upgrade from GLM to ACM.
2. Upgrade from TRM accounting to ACM.
3. Upgrade (patch installation) of ACM.

The report performs various checks divided into a few logical groups.

6.1.1.1 Ledger independent checks

The handling (assignment) of system IDs is handled via a specific table ACMId. It table keeps a next available ID for any entity used by ACM, which contains column ID. This check verifies the next system ID, what will be assigned to the entity, is higher than a maximum system ID already assigned (used).

6.1.1.2 Perspectives checks

It checks if all entities on ACMEntry and ACMVoucher really belong to a ledger for which the voucher belongs. The following entities are checked:

- on ACMEntry: account, cost center, project, mapping rule, mapping rule action, secondary account.
- on ACMVoucher: voucher type, period

6.1.1.3 Stored balances checks

It checks if stored balances (table ACMBalance) correspond with posted vouchers and entries. Verification is done on the most detail granularity (ledger, account, period, cost center, project, document currency). Following checks are available:

- Movement of stored balances equals to sum of amounts on the corresponding entries.
- Opening balance of stored balances equals to sum of amounts on the corresponding entries.
- Whether for each FINAL voucher with set balance checkpoint exists the corresponding stored balance row (balance for combination of the key characteristics: ledger, account, period, cost center, project, document currency). Amounts are not checked.

The first two checks are driven from the stored balances, i.e. for each stored balance (the combination of ledger, account, period, cost center, project, document currency) it checks the stored value (balance) by summing the posted entries in Final vouchers. The third check starts from the posted vouchers, i.e. for each posted voucher it finds the appropriate row in the table of balances.

Warning: When performing this check it is necessary to ensure that the balances and vouchers are not changed at that moment, i.e. no new voucher should be created and no voucher state shift should be performed.

6.1.1.4 Voucher balances checks

It checks if vouchers in any state after OPEN match the following conditions:

- Voucher is balanced per event date.
- Each voucher has at least two entries.
- Voucher is balanced per event date and document currency if the voucher property ACM-VOUCHER-BALANCE-PER-DOCUMENT-CURRENCY on voucher type is set.

Note: Since the voucher type property ACM-VOUCHER-BALANCE-PER-DOCUMENT-CURRENCY can be defined later, it may happen an issue is reported for vouchers which were created before defining that property.

6.1.1.5 Account properties checks

It checks if account settings are reflected on the posted entries. The following account settings are checked:

- Counterparty code required, i.e. if each entry posted on the account keeps information about a counterparty code.
- Cost center required, i.e. if each entry posted on the account keeps information about a cost center.
- Project required, i.e. if each entry posted on the account keeps information about a project.
- Transaction maturity date required, i.e. if each entry posted on the account keeps information about a transaction maturity date.
- Document currency, i.e. if each entry posted on the account has document currency equal to the specified account document currency.
- Non-posting, i.e. if no entry is posted on the account which is marked as non-posting.

Note: All the mentioned account settings can be change in course of time, so it may happen an issue is reported for entries which were created at the moment when the setup was consistent.

6.1.1.6 Posting checks

It performs the following general checks:

- All entries in state POSTED are assigned to some voucher.
- All vouchers in state FINAL or PREBOOKED have period.
- No accounting entry is assigned to a parent account.
- No voucher is posted to a parent period.
- Entries mapped according the mapping rules with actions, where account breakdown is not used, have accounts which exist in the appropriate actions as account or secondary account. The check is performed only for entries with filled action id.

6.1.1.7 All checks

It performs all the checks listed above at once.

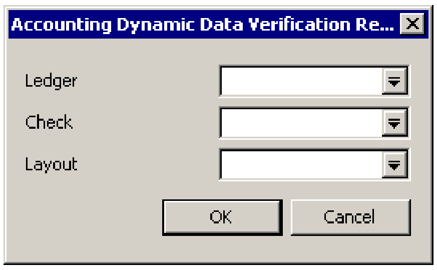
6.1.2 Starting the report

The report is not accessible from the standard Application Manager menu. If you wan to run it, you must run the Report Generator, File -> Open, go one folder level up, select the folder *types* and from there open *acm-dynamic-data-verification.frd* file.

6.1.3 Starting parameters

The report is executed per ledgers. However, there is no ledger permission check. For the report execution you must have just the object permission *ACMDynamicDataVerification*. This permission is by default granted for the user group ACM-ADMIN only.

The report is started with the following starting parameters:



Starting Parameter	Parameter Description	Default Value
Ledger	The ledger you want to verify. It is not a mandatory starting parameter. If none specified, it performs the checks for all the ledgers in the application.	-, i.e. it performs the checks for all the ledgers. **Note:** The execution for all ledgers will last longer than just for one ledger.
Check	The type of check: • All • Account Properties • Ledger Independent • Perspective • Postings • Stored Balances • Voucher Balances	All, i.e. it performs all the checks.
Layout	How the report output is displayed, i.e. what columns are selected, how they are grouped, etc. The report is distributed with one pre-configured layout.	Accounting Dynamic Data Verification

6.1.4 Report layouts

The total list of report output columns including their description is in the table below:

Column name	Column description
Check ID	Each verification check has its ID. The ID can be used as a link to the documentation for a detail description, e.g. P5. If there is no issue reported, the Item ID is P.
Entity System ID	System ID of an entity for which an issue is reported. If there is no issue reported, this field contains the ID of the ledger.

Column name	Column description
Entity Type	Type of an entity for which an issue is reported. It can be voucher, entry or balance.
Tip	What you should do for correcting the reported error.
Ledger	For what ledger the check was performed.
Message	Description of a check result.
Severity	Severity of the message: Info or Error.

It is recommended to group the report output by column Ledger and Check. It allows you to see for what ledger and check the result is reported.

The report is distributed with one default layout. It can be used as a starting point for creating own report layouts.

Layout name	Layout usage
Accounting Dynamic Data Verification	The default layout. Can be used for most of the cases.

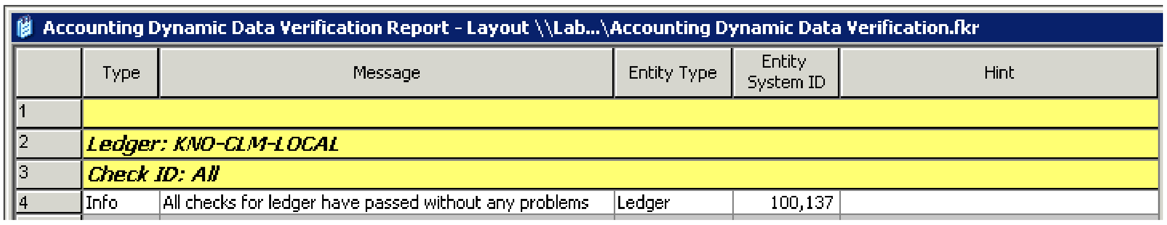
Note: Depending on the amount of dynamic data the report execution may last a few minutes.

6.1.5 Report drill-downs

The report does not contain any drill-down.

6.1.6 Report output example

If all the checks are passed without any problem, the following report output appears:



Accounting Dynamic Data Verification Report - Layout \\Lab...\Accounting Dynamic Data Verification.fkr

	Type	Message	Entity Type	Entity System ID	Hint
1					
2	*Ledger: KNO-CLM-LOCAL*				
3	*Check ID: All*				
4	Info	All checks for ledger have passed without any problems	Ledger	100,137	

6.2 Accounting System Administration Task activity

6.2.1 Introduction

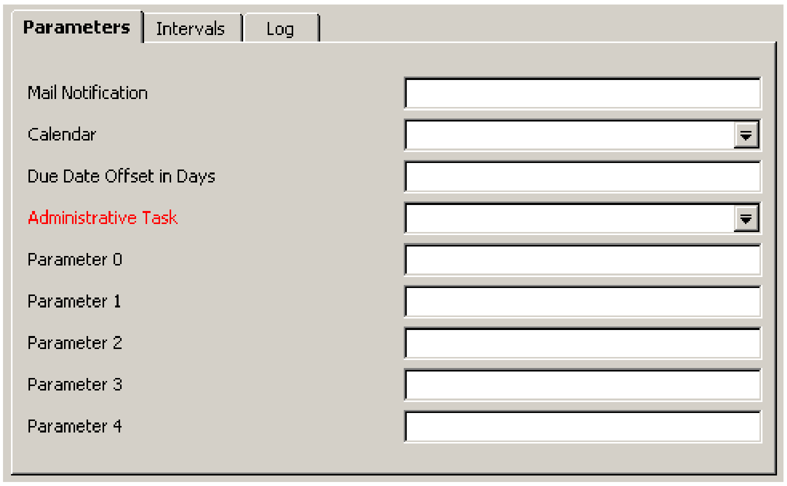
Accounting System Administration Task activity is a special ACM activity, which can be used by the system administrators only. There are several tasks which can be performed by this activity. The activity allows:

- Activating a workflow
- Saving ERP Document to file
- Setting accounting system log level
- Setting the system properties

All the tasks are described in the next sub-chapters.

6.2.2 Starting parameters

The activity starting parameters are following:



Mail Notification	
Calendar	
Due Date Offset in Days	
Administrative Task	
Parameter 0	
Parameter 1	
Parameter 2	
Parameter 3	
Parameter 4	

The meaning of the activity-specific starting parameters is following:

Starting Parameter	Parameter Description
Due Date	Due date is used just for starting the activity.
Administrative Task	There are several administration task which can be performed by this activity: • Activate workflow • Save ERP Document to file • Set Accounting System Log Level • Set System Property You must select one of these. For a detailed description of each task, see below.
Parameter 0 .. 1	The activity has five possible input parameters. Each administration task requires a specific input parameter.

Note: If Parameter 0 is empty, the task fails and generates an exception which describes the expected starting parameters.

The Accounting System Administration Task activity is designed for special administrator tasks so special permission is required. To start the Accounting System Administration Task activity, you (or your user group) must have the object permission "AdministratorActivity". This permission is granted via the Security Center (see the *TRM System Administration Guide* for details).

The following sections describe the particular administrative tasks.

6.2.3 Activating workflow

There is a workflow processor inside the accounting system which performs initial activation of the processes after its start and enables you to stop them. A workflow file is stored as an xml file inside the acm.war file for a complete startup and complete stop of the system. Additionally, there are workflows which enable you to stop or start the individual daemons running inside ACM.

For doing that select the administration task **Activate workflow** and use parameter 0 as follows:

Parameter 0 .... Workflow Name

6.2.3.1 ACM-ERP-SAP-RETRY-STOP workflow

In some situations, your target ERP system (e.g. SAP) might not be accessible for a few hours or days and you have many documents waiting in the queue to be posted to that SAP system. There is SAP Retry Daemon running inside ACM. It holds all the ERP documents which could not be posted/validated by the SAP system due to the inaccessibility. The SAP Retry daemon keeps trying to connect to the target SAP system at specified intervals.

During this time all documents are kept in memory and might exceed the permitted number of documents in the system at same time. When the connection is re-established, all waiting documents are re-processed. If you activate ACM-ERP-SAP-RETRY-STOP workflow, all waiting documents in SAP Retry Daemon are set to error state and memory is released. When the connection to the SAP system is back, you can start ERP Restart Document with given parameters to re-process documents.

6.2.3.2 Other workflows

There are other workflows which might be initiated by the Activate workflow task. All these workflows are used only in specific situations, and they can be processed just on a request by Wall Street Systems Support Center during a troubleshooting phase.

6.2.4 Saving ERP document to file

Select the task **Save ERP Document to file** if you need to save your ERP document to file. This is not for a regular file export. This activity does not make any changes to the document state; it simply saves the specified document to a file you specified without keeping a track of it. There is no check, where you save the document, or whether this file somehow hits the automatic mechanism of a regular file export. If you save an ERP document to the same path where you normally export the documents automatically you could have the same document there twice.

For this administration task you use the activity parameters as follows:

Parameter 0 .... Document System ID (number which you can see in ERP reports) .

Parameter 1 .... path and filename for new document to be created. This path and file must be accessible by the ACM server process.

6.2.5 Setting accounting system log level

Wall Street Systems Support Center could ask you to increase the log level. ACM writes messages into a log file using the Log4J library. This product takes a default setting from a property file.

If you switch the log level from INFO to DEBUG, the system runs in the DEBUG mode, but this change is not saved into the Log4J property file, so the log level is lost when you stop the instance. As soon as you identify the problem for which you increased the log level, switch it back to the INFO level (the default level).

Warning: There are performance costs imposed by higher log levels.

One way to increase the log level is to stop the ACM server, to change the Log4j property file and to start the ACM server again. The administration task **Set Accounting System Log Level** offers the similar functionality in runtime without requiring to stop and start the server.

For this administration task you use only the activity parameter **Parameter 0** as follows:

Parameter 0 .... required log level.

As a value you can set one of the following values from the table below:

Log level	Log level description
trace	The most detail logging; can significantly decrease the system performance. The amount of information is typically uselessly big.

Log level	Log level description
debug	A typical logging level for a development or an issue investigation.
info	The default log level; only errors, warnings and information messages are logged.
warn	Only errors and warnings are logged.
error	The most economic logging; only the errors are logged.

6.2.6 Setting system properties

ACM server operates in a Java runtime environment where system properties specified by the environment file are loaded and accessible by the ACM processes. Some parts of ACM can change their behavior in runtime based on the changes to these system properties in runtime. This task is helpful for troubleshooting.

If you want to change some system property in runtime, select the administration task **Set System Property** and use the parameters 0 and 1 as follows.

Parameter 0 .... System Property Name

Parameter 1 .... New system property value

Warning: Do not use this task unless you are asked by the Wall Street Systems Support Center to do so.

Chapter 7 Setting up security

This chapter describes the security setup in ACM.

7.1 Storing the ACM Server password

A password for the ACM Server can be stored in the shared memory in the same way as for the TRM applications. See *WSS Database Setup Guide* for more information.

For configuring the ACM Server in this way (i.e. based on reading the password from the shared memory), do the following:

1. In the shared memory, store the password of the user name under which you want to run the ACM Server.
2. Set the environment variable `ACM_JDBC_USER` to the user from step 1 and remove the variable `ACM_JDBC_PASS` from the environment.

If you set the `ACM_JDBC_PASS` variable, the ACM Server does not search for the password in the shared memory.

Important: It is not mandatory to read the password from the shared memory.

7.2 Managing permissions

Permission management supported by ACM allows to secure an access to different objects and functions (features) of ACM. Per user or user group you can define:

- What data you can access; note that you can define not only type of data (e.g. accounting period set), but within one data type you can define an access per particular rows.
- What you can do with accessed data (e.g. only read).
- For what ledger(s) (accounting books) you can perform what activity.
- For what ledger(s) you can work with voucher state.
- For what ledger(s) you can run what report.

All that is handled via the following principals:

1. **Domain and object permissions**

   ACM static data access is controlled by domain permissions and by ACM object permissions which allow to define an access to the individual ACM static data editors.

2. **Users and user groups**

   Object permissions are defined per users or user groups. You define the user groups if you can split the users to standardized groups. Instead of defining particular object permissions for each new user, you can just assign a user to some existing user group. Any user can belong to zero, one or more user groups. Note that each user or user group inherits all permissions from all the user groups to which belongs.

3. **Ledger permissions**

   ACM Ledger permissions define per ledger what type of activity, report, and/or voucher state is allowed for what user or user group. The ledger permissions are defined via the Ledger editor, page Ledger Permission. This approach is very close to the TRM portfolio permissions definition via the Allowed Users tab in the Portfolio editor.

As you can see, the very the same principals are used by TRM.

Warning: Each person working with ACM should use his/her own system user name. It is strongly recommended not to use an administrator user name or to share a user name between the multiple users (mainly due to the mixing accounting perspectives).

The description of each permission management follows.

7.2.1 Static data permissions

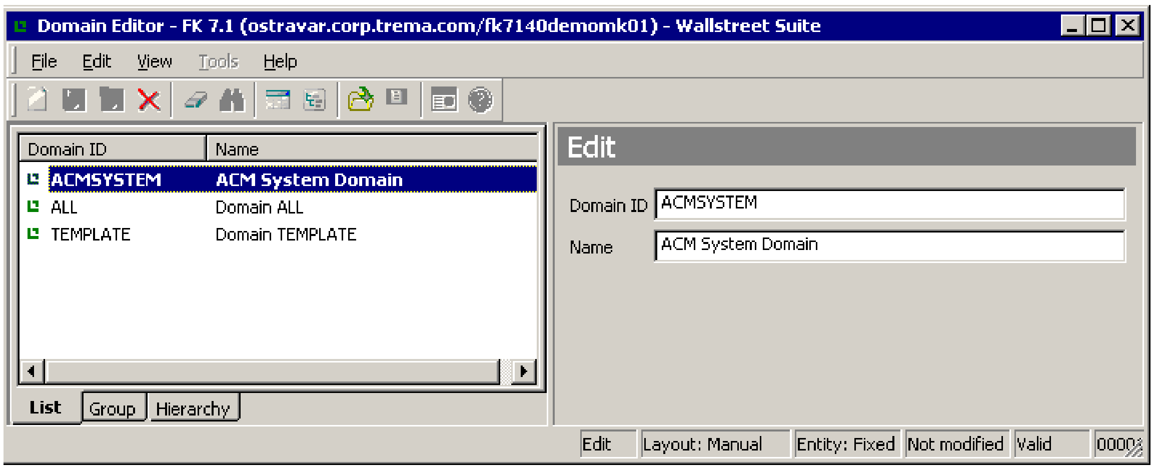
Static data access is controlled by **domain** permissions and by **object** permissions. These two types of permissions are used in combination, i.e. for manipulating with some static data entity you must be granted for both:

- the correct domain permissions for the domain to which the entity belongs and
- correct object permissions for the entity.

In other words: while domains control which set of rows in a database a user can manipulate (for example, change data in a domain called Europe), object permissions allow a user to perform certain general tasks (for example, change a chart).

7.2.1.1 Domain permissions

The domain permissions control the access to the data located in a single organizational unit – **the domain**. Domain permissions specify whether a user can read, create, modify, or delete entities in the domain. Domains are defined via the Domain Editor (Application Manager -> WallstreetSuite -> Transaction and Risk Mgmt -> Administration Tools).


Domain Editor - FK 7.1 (ostravar.corp.trema.com/fk7140demomk01) - Wallstreet Suite

Domain ID	Name
ACMSYSTEM	ACM System Domain
ALL	Domain ALL
TEMPLATE	Domain TEMPLATE

Edit

Domain ID: ACMSYSTEM

Name: ACM System Domain

Edit | Layout: Manual | Entity: Fixed | Not modified | Valid | 0000

Note: Each accounting processing step allows a certain customization/configuration. To simplify the ACM configuration, the setup of some steps is preconfigured. Whereas some pre-configuration is done directly (e.g. Event to entry mapping where no owner is defined), some configuration is a chart dependent. For this reason, each ACM installation contains the ACMSYSTEM chart. All such data are defined in the domain ACMSYSTEM. For details see the *ACM User Guide*.

Warning: Do not ever delete the domain ACMSYSTEM!

Typically, a domain is defined for the set of individual entity instances (for example, several specific charts and several specific ledgers), and one dedicated user is responsible for the definition and maintenance of all entity instances which have the same access level in that domain. Domains are not related to behavior or functionality; they are used to categorize static data that should be maintained together. Consequently, domains should be defined at the beginning of the system setup according to business requirements. The individual entity instances are then created inside the predefined domains according to the domain permissions.

Key ACM entities are domain-dependent. They are:

- Ledger
- Ledger Group
- Period Set
- Chart
- ACM Rule.

Period Set and Chart are the entities to which some other ACM entities belong. The entities dependent on either Period Set (only Accounting Periods) or Chart (Account Types, Accounts, Mapping Rules, Voucher Types, Cost Centers, Projects, Grouping Rules, etc.) are domain-free, and are maintained from the perspective of the key entity only.

Domain permissions are inherited by these key entity-dependent entities via the key entities. For example, if Period Set "1" belongs to a domain "A" and a user has permission for this domain, the user can access the Accounting Periods defined for the Period Set "1" (depending on the object permissions).

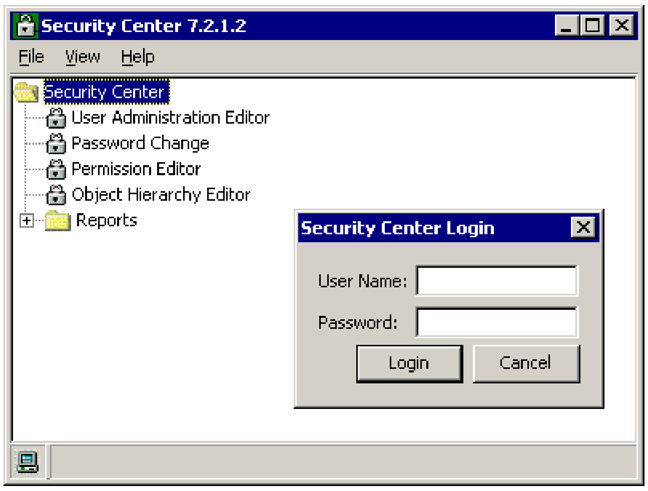
Each entity from the list above is created in (assigned into) a **primary (main) domain**. Beside the primary domain, all of the entities above can be assigned zero, one or more **secondary domains**.

Secondary domains provide a way of making the static data entity accessible to users that typically maintain a domain other than the entity's primary domain. The secondary domains provide only Read access to the static data entities (regardless of the type of permission to the secondary domain the user has).

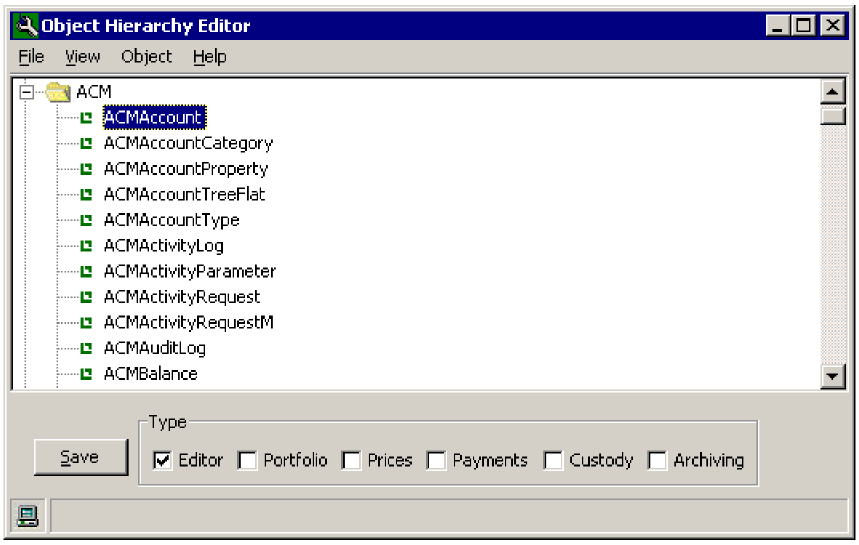
7.2.1.2 Object permissions

Object permissions control the access to the actions that a user can perform on the static data. Even the objects are defined on the level of the particular database tables, they finally represent an access to the individual ACM static data editors.

The objects are placed in an **object hierarchy**. The object hierarchy serves as a way of organizing the objects into a user-friendly view. The objects as well as the object permissions are defined via the **Security Center**. Note that it is quite often necessary to change a login name to an user with administration permissions (File/Login):



The objects are defined via the **Object Hierarchy Editor** which looks as follows:



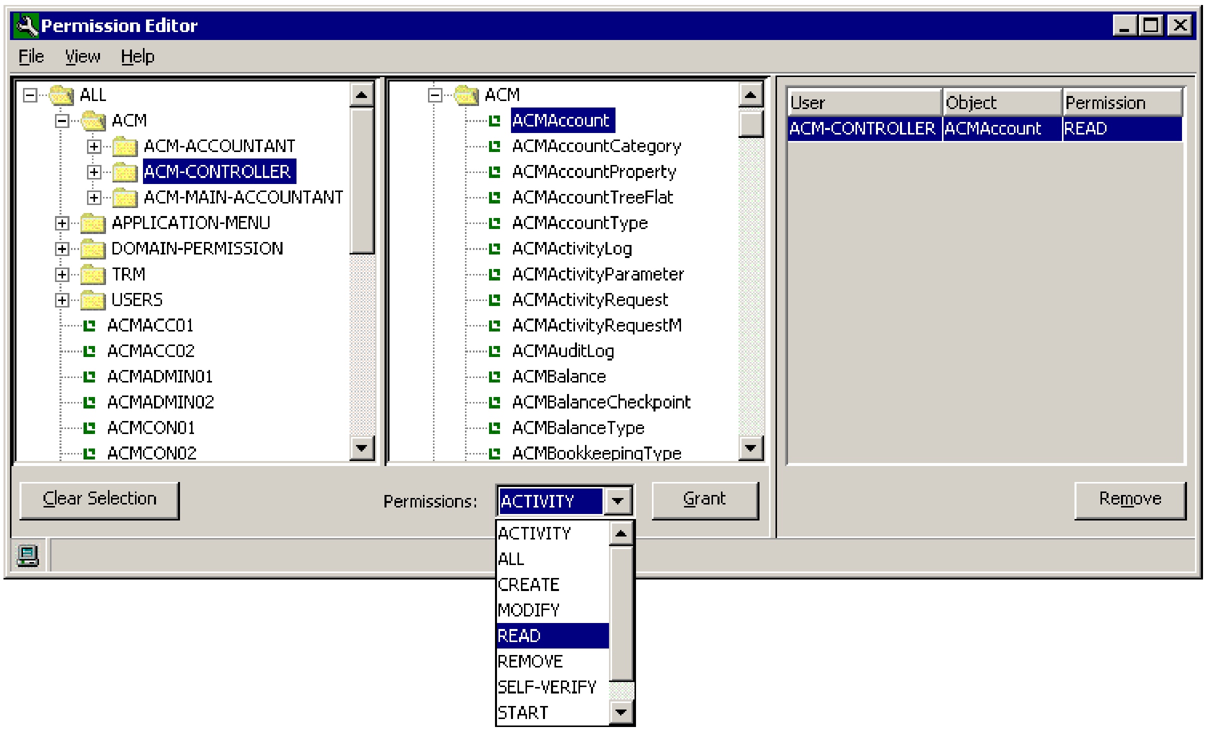
Type: [✓ Editor] Portfolio Prices Payments Custody Archiving

All the ACM (accounting) objects are defined under the ACM parent object.

Note: Besides the ACM parent object you can see there also the parent object Accounting Management. It is a residue object from the TRM accounting available in versions before the 7.x version (accounting before ACM). You should not define anything there. The object is kept in the system just for some backwards compatibility and upgrade needs.

For each user/user group (for their definition see the next chapter *7.2.2 Accounting user groups* on page 48) and object you may define a specific **object permission** (READ, MODIFY, ALL, etc.).

The object permissions are defined via the Security Center, **Permission Editor**. This editor links together the defined users/user groups (the editor lefthand sub-window) and the security objects (the editor middle sub-window). For each combination of a user/user group and object you can define a particular set of permissions (the editor righthand sub-window). See the example below for the user group ACM-CONTROLER and object ACMAccount.



User	Object	Permission
ACM-CONTROLLER	ACMAccount	READ

Ther object permissions are of the following types:

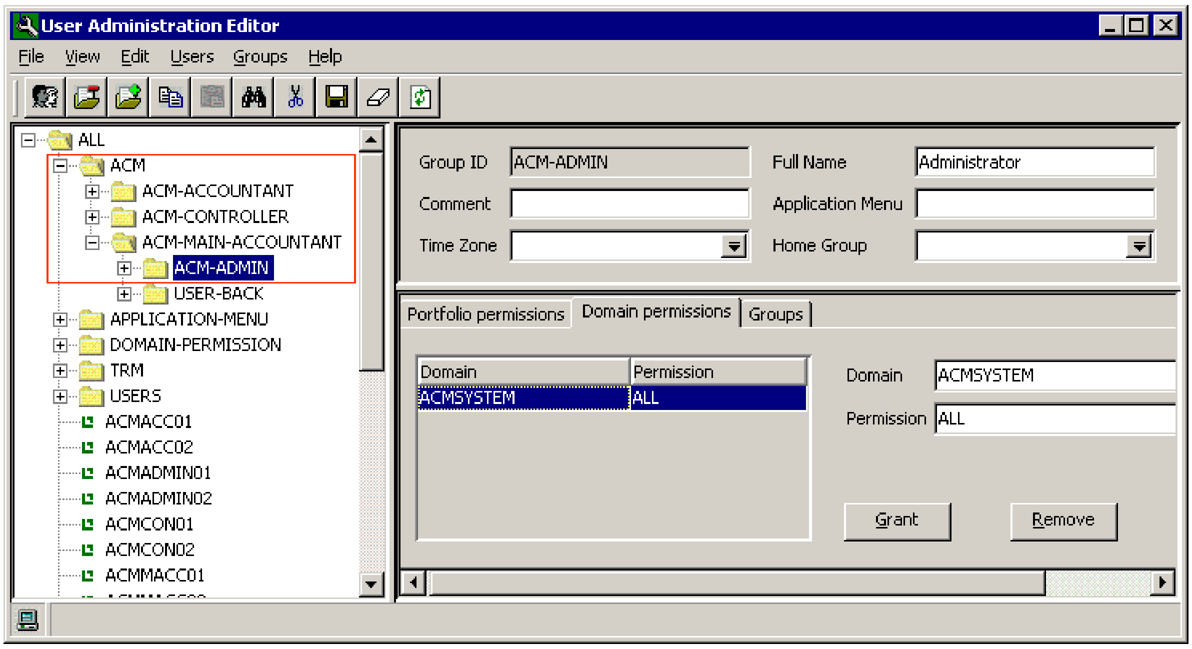
- ALL
- READ
- CREATE
- REMOVE
- MODIFY

Object permissions specify if a user can read, create, modify, or delete e.g. charts, accounts, ledgers, and other static data objects.

A complete list of the ACM objects including the link of the individual objects to the ACM editors (column *Description)* is listed in *Appendix B Object permissions* on page 115.

7.2.2 Accounting user groups

User groups as well as the particular system users are defined via the Security Center, User Administration Editor. See its example below:



```
ALL
  ACM
    ACM-ACCOUNTANT
    ACM-CONTROLLER
    ACM-MAIN-ACCOUNTANT
      ACM-ADMIN
      USER-BACK
  APPLICATION-MENU
  DOMAIN-PERMISSION
  TRM
  USERS
  ACMACC01
  ACMACC02
  ACMADMIN01
  ACMADMIN02
  ACMCON01
  ACMCON02
  ACMMACC01
```

Group ID	ACM-ADMIN	Full Name	Administrator
Comment		Application Menu	
Time Zone		Home Group	

Domain	Permission
ACMSYSTEM	ALL

Domain	ACMSYSTEM
Permission	ALL

For each user and user group you can define:

- Portfolio permissions (what type of access to what portfolio).
- Domain permissions (what type of access to what domain).
- To what group a particular user or user group belongs.

Note: Each user or user group inherits all permissions from all the user groups to which belongs. All users in the same user group have the same object permissions.

ACM is pre-configured with four user groups that can be used without any additional setup, i.e. it is just up to you to what user group you assign a particular system user. The available accounting groups are:

1. ACM-ADMIN

This is the most powerful user group having permission ALL for all the accounting objects including those defined in domain ACM-SYSTEM and including the possibility to run the accounting administrator tools (for details see *Chapter 6 Using accounting administrator tools* on page 35).

2. ACM-MAIN-ACCOUNTANT

This is the second most powerful accounting user group. It has the same permissions as ACM-ADMIN with the following exceptions:

a. Only READ permission for domain ACM-SYSTEM.

b. No possibility to run the accounting administrator tools.

c. Only READ permission for defining event to entry mapping via the Event to Entry Mapping Editor.

d. Only READ permission for defining accounting parameters via the Accounting Parameter.

3. ACM-ACCOUNTANT

This user group has **the same** restrictions as ACM-MAIN-ACCOUNTANT **plus** the following ones:

a. Only READ permission for objects having M at the end of their name (e.g. ACMLedgerM) used for assigning multiple domains on some entities (e.g. ledger). Note that these so called secondary domains add a reading permission for the domains defined in the "Domains" tab of the editors, e.g. in the Ledger editor. The READ permission on the M table allows only to see domains defined in that tab, but no editing.

b. Only READ permission for objects needed for the definition of ERP static data (ACMERPExport Def, ACMERPSystemDef, ACMERPTarget).

c. Only READ permission for defining ledger permissions via the Ledger Editor (object ACMLedgerPermission).

d. Only READ permission for defining accounting statement report templates (objects ACMStmtReportRow, ACMStmatReportTmpl).

4. ACM-CONTROLER
Has just READ permission for all the accounting objects. If you belong to this group you can open all accounting editors, can view all data in domains, for which you have at least the READ permission, but you cannot do anything else, i.e. you cannot create new data nor modify or remove the existing ones. You have only READ permission for domain ACM-SYSTEM and you cannot run the accounting administrator tools.

A detail list of all the ACM permission objects versus all the ACM user groups can be seen in the *Appendix B Object permissions* on page 115.

Warning: The assignment of the particular accounting activities and reports to users and/or user groups for given ledger must be configured during an implementation project via the Ledger Editor (Ledger Permission page). For details see the next chapter.

7.2.3 Ledger permissions

Ledger permissions allow the definition of fine-grained permissions for operations which are executed for a ledger. There are three basic types of such operations:

1. Activities, i.e. a possibility to manipulate with the key ACM dynamic data: entries and vouchers
2. Reports, i.e. a possibility to view the dynamic data (entries, vouchers, account balances) and ledger specific data (e.g. period state).
3. Voucher states.

Note: If an activity or report is started for a ledger group, it checks a permission for each ledger within the group separately. Permissions must be defined for every ledger: if a permission for even one ledger is missing, the whole process fails.

7.2.3.1 Ledger permissions for activities

One set of ledger permissions is linked to activities. There is a specific ledger permission linked to each ACM activity type, which is starting by a ledger or a ledger group. The ledger permission names related to the activities start with the word "Activity".

At the time when an activity is executed by a user, the system checks whether the user has the particular permission for the given ledger and activity type and only then the activity is performed.

There is a specific permission - **ALL_ACTIVITIES**. If a user is granted ALL_ACTIVITIES permission for a particular ledger, any activity may be performed on that ledger by the user.

7.2.3.2 Ledger permissions for reports

The second set of ledger permissions is linked to reports. There is a specific ledger permission linked to each ACM report which start with a ledger or a ledger group. The ledger permission names related to the reports start with the word "Report".

When a report is executed by a user, the system checks whether the user has the particular permission for the given ledger and report, and only then the report is executed.

Note: To fetch and see vouchers and entries in any state in Accounting Manager, the user must have the ACM-JOURNAL-REPORT permission granted.

There is a specific permission - **ALL_REPORTS**. If a user is granted ALL_REPORTS permission for a particular ledger, any report may be performed on that ledger by the user.

7.2.3.3 Ledger permissions for voucher states

The third set of ledger permissions is linked to voucher states. There is a specific ledger permission linked to each voucher state. They all have the name started by "Voucher state", e.g. *Voucher state Verify.*

When a voucher is shifted to a certain voucher state, the application first checks if the user is granted the proper permission for the given voucher state, i.e. a destination voucher state is checked, not the source one.

Note: If you want to modify the vouchers, you must have the ledger permission ACM-STATE-OPEN.

There is a specific permission - **ALL_STATES**. If a user is granted ALL_STATES permission for a particular ledger, any voucher may be shifted to any voucher state on that ledger by the user.

Note: There are two voucher states which are not used by the system by default. They are *Voucher state Verify-2* and *Voucher state Verify-3*. It is not necessary to define the permissions for them.

Chapter 8 Customizing accounting framework

This chapter describes possible customizations in the following areas of the accounting framework:

- TRM accounting inputs and events generation
- CLM accounting inputs and events generation
- Accounting processing
- Accounting reports
- Accounting Manager
- Hedge Manager

8.1 TRM accounting inputs and events generation

Accounting events in TRM are generated from accounting inputs. This applies for daily (RUNNING and OFF-BALANCE) as well as for closing accounting events (CTB and DAILY-DELTA).

Accounting input is elementary information from which an accounting event might be generated. Accounting input exists for every transaction/cashflow date which might be important for the accounting of the particular transaction/cashflow. Accounting inputs are processed by input handlers (the part of the AccountingEventRules), which result in accounting events (none, one or more from one input).

Note: Accounting inputs as an entity exist in TRM only. CMM does not use accounting inputs, i.e. CMM does not need accounting inputs for the accounting events generation.

Accounting event is a standardized structure of accounting data generated by TRM containing all necessary information for processing in ACM. Accounting events in TRM are generated from Accounting Inputs. Accounting events are generated by running the appropriate TRM CTB/Daily/Daily Detla Events Generation activities.

Accounting events are taken over by ACM and they are transformed to accounting entries. Accounting entry is an elementary accounting information in ACM, which is the subject of accounting processing in ACM.

This chapter describes the customizations of TRM accounting inputs and events generation.

8.1.1 Customizing inputs generation

Accounting inputs generation logic is handled via the queues framework (aka message bus), i.e. a message from the transaction flow is sent into a queue, and an independently running process takes the message and creates Accounting Inputs asynchronously.

Transaction States are defined in `transaction_state.sql`. By running it, it creates records into TransactionState table.

Transaction Operations/Agents are defined in the `flow.py` file. By running it, it creates records into TransactionOp, TransactionOpAgent, AgentProcedure and AgentMessaging tables.

In addition, there is also a `commit.py` file contributing to TransactionOp and TransactionOpAgent tables for Committing (Applying) transactions.

Note: Configuration in the TransactionOpAgent table specifies what agent procedure (setup_key) is called for (i) what transaction/cashflow (rule_id, not_rule_id), (ii) what transaction action (op_id) and (iii) at what transaction state (minimum_state_id, maximum_state_id). Accounting inputs are generated for the transactions/cashflows matching the rule(s) with a setup_key linked to the agent procedure (table AgentProcedure) DoTAccountingInput, which is the name of a stored procedure generating the inputs.

The Suite standard set-up in the above files ensures that

- **Accounting inputs for relevant transactions are created at state COMMITTED** (a hidden state) when transaction is matching TFLO-ACT_BOOKING rule (delivered with the standard set-up). At this point transaction details are sent as a message to accounting-input.update queue, from which the accounting inputs serviced provider reads it and creates Accounting inputs.
- Reverse accounting inputs are created in the following cases
  - Transaction is matching TFLO-ACT_BOOKING and moves back in the flow (on relevant Reject button) cross COMMITTED (a hidden state). At this point transaction details are sent as a message to accounting-input.update queue, from which the accounting inputs serviced provider reads it and creates reversing accounting inputs.
  - At any Cancel operation (Cancel button) DoTCancelAccountingInput is called directly. Note that in this case queues are not used and that stored procedure is called in the beginning of process, since Cancel deletes transactions cashflows and important information is lost before asynchronous Accounting input processing would process the transaction.
- Some actions (as BVC, additional fees, etc.) which are touching transactions without moving in the flow after COMMITTED state are updating Accounting inputs as necessary. This is done via sending transaction information into accounting-input.finetune queue, from which accounting inputs serviced provider reads it and reverses/creates accounting inputs accordingly. Those actions have to be listed in commit.py file, and all supported actions are in the delivered file. 'execute_bvc' action can be taken as an example to see how they are handled (mask 1 is set, and it is used then as a condition for sending message to queue).
- Accounting inputs serviced provider, which handles all accounting inputs generation, uses new REFRESH-TRANSACTION operation defined by flow to ensure processing of fresh information regardless of the asynchronous processing (it re-fetches transaction information before their actual processing).

In addition, there is a mechanism ensuring when a cashflow of a transaction is changed at specific listed columns and there is an input already existing, the input is marked and then reprocessed. The marking is done via an update database trigger on the Cashflow table.

In addition to a hidden COMMITTED state which determines when Accounting Inputs are created and reversed, there is also a hidden BOOKABLE state, which determines when Accounting Events are created for the transaction (and when also FX conversion is ran).

Note: From the accounting point of view the **7.3 standard set-up** behaves differently than the framework in 7.1 and 7.2: it **reverses accounting inputs whenever the transaction is moved back "over the line" of the COMMITTED state.** This means that when transaction is in any of the states before that (such as RE-OPEN, etc.), reversing Accounting Inputs will be generated (and processed when accounting is run, typically at the end of the day).

Every Suite version comes with a functioning pre-configured transaction flow. If the transaction state agents are customized, it is necessary to ensure that relevant Accounting inputs are still created by calling the Agents at right places to ensure that no new bookkeeping nor reverse is missed. When you customize the flow, it is also important to ensure that the definition files corresponding to the generated flow are safely kept and identified so you do not break the configuration by re-running a different version of the file.

Warning: **Any customization must not be done directly into the database tables**, otherwise the definition files and database content (used for actual processing) would not be in sync and the tables could not be re-build anymore!

Off-balance inputs (type = 2) generation should never require any user customization. The same applies for Closing inputs (type = 3), which are automatically created by the TRM CTB Events Generation and/or TRM Daily Delta Events Generation activities. They should never require any user customization either.

8.1.2 Customizing events generation

Accounting events are generated according to the rules defined in the AccountingEventRule table of TRM database. The standard TRM rule logic applies: if Rule matches and Not Rule does not match, then the procedure specified in the rule is executed and an accounting event of the given type is created.

The accounting event generation process is run for a specified date. Each accounting input is matched with the accounting event rules in order of priority. If Rule matches and Not Rule does not match, the stored procedure specified in event_proc field is executed and accounting events are created. All matching rules are executed, so one accounting input can generate a number of accounting events.

The standard Wallstreet Suite installation contains the pre-configured accounting event rules (table AccountingEventRule). They refer to some default Rules and Not Rules.

Warning: **Even the configuration is in the database and as such could be customized, it is strongly recommended to not perform them and to keep the default setup since these areas are very sensitive and any customization may complicate a support from Wall Street Systems.**

TRM accounting events are of three basic categories corresponding to three input types:

- **RUNNING**
- **OFF-BALANCE**
- **CLOSING**

Key information like event dates, amounts and FX rates are put to the events by the events generation process itself. Additional information are enriched by stored procedures. The information are populated into so called DIM fields. For a default structure of those additional information see *A.1 Accounting Event* on page 107.

TRM includes four procedures for filling the DIM fields.The procedures can be customized by a software engineer during the implementation project at a customer site so they provide additional information which are not provided by default.

Warning: By default, the procedures provide all useful and necessary accounting information, so normally there is no need to customize them. They should be customized only in situations when you need some extra TRM transaction parameters and/or branch codes (first 10 parameters and first 5 branch codes are provided by default only), or when some unusual TRM information is needed.

The procedures should be customized per portfolio owner. You can either customize the default procedures or you can create your own. If you create your own procedures, you must specify their names for the appropriate portfolio owner in the **Accounting** table.

Note: When using the default procedures/setup there is no need to specify the procedures in the Accounting table; they are used by default although they are not specified there.

The default procedures responsible for the creation of TRM events are structured as follows:

Procedure Name	Procedure Description	Customized procedure to be put into Accounting table column
GetGeneralEventDims	Fills event information general for all 3 basic bookkeeping types: Running, CTB, Off-Balance	general_dim_proc
GetRunningEventDims	Fills event information relevant for Running bookkeeping type only	running_dim_proc
GetClosingEventDims	Fills event information relevant for Running bookkeeping type only	closing_dim_proc
GetOffBalanceEventDims	Fills event information relevant for Running bookkeeping type only	off_balance_dim_proc

8.2 CLM accounting inputs and events generation

Accounting of CLM payment advices and payment allocations is based on payment allocation events (aka entity events). An event is a standardized structure of accounting data generated for CLM transactions containing all necessary information for processing in ACM. The events are generated from payment allocation inputs (aka entity inputs). Input is an elementary information from which the events might be generated.

The events are taken over by ACM and they are transformed to accounting entries. Accounting entry is an elementary accounting information in ACM, which is the subject of accounting processing in ACM.

This chapter describes possible customizations of inputs and events generation for CLM payment advices and payment allocations.

8.2.1 Customizing inputs generation

Events for CLM payment advices and payment allocations are generated from inputs.

Accounting inputs are processed by input handlers. The processing of one input results into none, one or more events.

The best way for viewing and checking the accounting inputs in TRM is the usage of the Accounting Payment Allocation Input Event Report. For details about the report see the *ACM User Guide*.

Accounting inputs for CLM transactions (payments and payment allocations) are created without a link to any result mode since they are the same for all of them.

Warning: No customization is allowed for generation of inputs for payment advices and allocations!

8.2.2 Customizing events generation

Events for CLM payment advices and payment allocations are of two categories corresponding to two input types:

- **Payment**
- **Payment Allocation**

They are created by running the TRM Payment Allocation Events Generation activity. The only part of the process, which can be customized, is the structure of the events.

Key information like event dates, amounts and FX rates are put to the events by the events generation process itself. Additional information are enriched by stored procedures. The information are populated into so called DIM fields. For a default structure of those additional information see *A.2 Entity Event* on page 111.

DIM fields are populated by two stored procedures.The procedures can be customized by a software engineer during the implementation project at a customer site so they provide additional information which are not provided by default.

Warning: The procedures provide by default all useful and necessary accounting information, so normally there is no need to customize them. They should be customized only in situations when you need some unusual TRM information is needed.

The procedures should be customized per portfolio owner. You can either customize the default procedures or you can create your own. If you create your own procedures, you must specify their names for the appropriate portfolio owner in table **AccountingEntity**.

Note: When using the default procedures/setup there is no need to specify the procedures in the AccountingEntity table; they are used by default although they are not specified there.

The default procedures responsible for the creation of events for CLM payment advices and allocations are structured as follows:

Procedure Name	Procedure Description	Customized procedure to be put into AccountingEvent table column
GetPaymentEventDims	Fills event information for payment advices and payment side of allocations.	dim_proc for a kind Payment
GetAllocCashflowEventDims	Fills event information cashflow side of allocations.	dim_proc for a kind PaymentAlloc

8.3 Accounting processing

Most of the accounting processing is implemented in Java and runs inside an application container such as Tomcat. Therefore the customizations of that processing are quite limited and are reduced to just a few options.

8.3.1 Customizing entries and vouchers

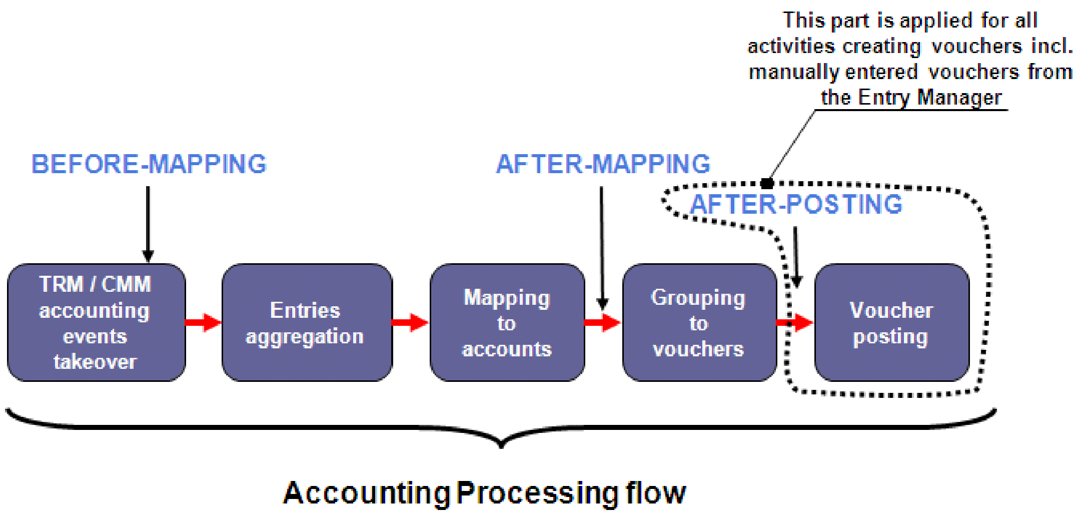
8.3.1.1 Introduction

8.3.1.1.1 CSDs overview

Accounting events from TRM and CMM are processed in ACM by running the Accounting Processing activity. Accounting processing in ACM consists of the following basic steps:

- Accounting event takeover and its transformation to an accounting (ACM) entry
- Entries aggregation
- Mapping to accounts
- Grouping to vouchers
- Voucher posting

The processing steps are configurable via the editors (as described in the *ACM User Guide*). This chapter describes some exceptional customer specific developments (CSDs, aka customizations) done outside the editors. The CSDs in accounting processing can be done at three different stages as shown in the picture below:


This part is applied for all activities creating vouchers incl. manually entered vouchers from the Entry Manager
BEFORE-MAPPING
AFTER-MAPPING
AFTER-POSTING
TRM / CMM accounting events takeover
Entries aggregation
Mapping to accounts
Grouping to vouchers
Voucher posting
Accounting Processing flow

The supported stages are:

- BEFORE-MAPPING – enables customizing of entries that have not been mapped yet. More precisely, the entries are customized already during the takeover step, i.e. already before the entries aggregation so the entries created in state NEW are already customized.

Warning: CSD BEFORE-MAPPING is just a one time process since each entry goes via that hook place just once; in other words, when running the Undo Accounting Processing followed by Accounting Processing, the entry is not affected by that CSD anymore.

- BEFORE-POSTING – enables customizing of entries after mapping but before posting to vouchers, i.e. entries in state MAPPED
- AFTER-POSTING – enables customizing of entries posted into vouchers but before the voucher is shifted to FINAL state. You can customize entries as well as vouchers.

8.3.1.1.2 Enabling CSD per ledger vs. all ledgers

Each customization type is implemented as a stored procedure. ACM allows to specify the ledgers for which a specified customization type takes place. By default, no customization is done. To enable customization for a ledger at a particular stage, the following ledger properties (defined in the Ledger and/or Chart of Accounts editors) must be set to **Yes**:

- ACM-CSD-BEFORE-MAPPING=Yes
- ACM-CSD-BEFORE-POSTING=Yes
- ACM-CSD-AFTER-POSTING=Yes

To apply customizations to entries in all ledgers, the following environment variables can be set to **true**:

- `ACM_CSD_BEFORE_MAPPING`=true
- `ACM_CSD_BEFORE_POSTING`=true
- `ACM_CSD_AFTER_POSTING`=true

The properties set on the ledger have priority over the properties specified in the environment. The following example describes a scenario where the customization is done at the BEFORE-MAPPING stage for all ledgers, with the exception of ledger WSS-WORLD:

- ACM-CSD-BEFORE-MAPPING=No (for ledger WSS-WORLD only – ACM Ledger Editor)
- ACM_CSD_BEFORE_MAPPING=true (environment variable)

8.3.1.1.3 CSD_NOT_PROCESSED flag usage

The customization of entries and vouchers is controlled by flag `CSD_NOT_PROCESSED` (value 134217728, column flags in `ACMVoucher` and `ACMEntry` tables). All entries and vouchers to be processed are marked by ACM with this flag once the above mentioned property is set to Yes/True. An entry or voucher with such flag does not enter further processing stages. Therefore, the customization routine must clear the flag when the customization is done.

Warning: The flag is set by ACM on all entries and vouchers so the customization routine must clear the flag from all the entries and vouchers, not only from those which are the subject of the customization.

8.3.1.1.4 Useful tips

The following are tips you should follow when creating the CSD:

- ACM uses optimistic locking to prevent data corruption in a concurrent environment. The customization routine should always increment the value of the version column (ACMEntry and ACMVoucher table) for each updated entry. ACM does not use the automatic timestamp optimistic locks provided by the Sybase and MSSQL platforms.
- The customization procedures are always called within an existing database transaction context. The customization should never attempt to commit or roll back the transaction. A non-zero return value from the procedure indicates an error, and instructs ACM to roll back the database transaction. The routine may provide additional error information using the output variable @fail_reason.
- Command print (MSSQL, Sybase) or RESULTS.PRINT (Oracle) can be used in customization procedures to create messages what are written to thee ACM server log file. If message precedes character '!' then it is also written to Accounting Activity Log and can be viewed by the Accounting Activity Log report.

The next sub-chapters describe the principles of the customization in detail.

8.3.1.2 BEFORE-MAPPING and BEFORE-POSTING customizations

The BEFORE-MAPPING and BEFORE-POSTING customizations should be used whenever customization applies to entries.

The customizations are of two types:

- BEFORE-MAPPING – enables customizing of entries that have not been mapped yet. More precisely, the entries just after the takeover, but before aggregation, i.e. the entries in state NEW.
- BEFORE-POSTING – enables customizing of entries after mapping but before posting to vouchers, i.e. entries in state MAPPED

Warning: These two CSDs are applied only for entries processed by Accounting Processing activity. They are not applied for manually entered vouchers/entries via the Accounting Entry Manager.

The BEFORE-MAPPING and BEFORE-POSTING customizations should be implemented in stored procedures `ACMCSDBeforeMapping` and `ACMCSDBeforePosting` respectively. Both procedures are called with the same parameters:

```
  @ledger_id           numeric(19,0),
  @instrument_group_id UMPathId = null,
  @portfolio_id        PortfolioId = null,
  @instrument_id       nvarchar(30) = null,
  @number              numeric(19,0) = null,
  @origin_group_id     int = null,
  @bookkeeping_type_id int,
  @fail_reason         nvarchar(255) = null output)
```

The routine should always apply the customization only to entries that satisfy the provided parameter conditions. Whether the parameters are provided or they are null depends on the parameters of the executed Accounting Processing activity. The following code illustrates a typical implementation:

```
update ACMEntry set
  param_19="CSD Before Mapping",
  flags=flags-134217728,
  version=version+1
where
  ledger_id=@ledger_id and … and … and …
  and flags&134217728 = 134217728
  and state_id=100   --Only state NEW entries in BEFORE-MAPPING
```

ACM is installed with default `ACMCSDBeforeMapping` and `ACMCSDBeforePosting` procedures.

- Procedure `ACMCSDBeforeMapping` fills ACM entry parameter 19 by value "CSD Before Mapping" and removes the flag `CSD_NOT_PROCESSED` from entries.
- Procedure `ACMCSDBeforePosting` fills ACM entry parameter 18 by value "CSD Before Posting" and removes the flag `CSD_NOT_PROCESSED` from entries.

8.3.1.3 AFTER-POSTING customization

The AFTER-POSTING customization should be used whenever customization applies to entries posted to vouchers and can be done on entries and/or vouchers.

Warning: This CSD is applied to any posting voucher, i.e. for voucher created by any activity including the vouchers entered manually via the Accounting Entry Manager.

When customization is specified, it must be applied to all vouchers including those manually posted via ACM Accounting Manager. The `ACMCSDAfterPosting` procedure has the following signature:

```
create procedure ACMCSDAfterPosting
 (@ledger_id           numeric(19,0),
  @target_voucher_state numeric(19,0),
  @process_version      int,
  @fail_reason          nvarchar(255) = null output)
as  …
```

Customization at this stage takes a different approach to identify vouchers to be customized. Again, the flag `CSD_NOT_PROCESSED` is set on vouchers that have not been customized yet. In addition, the provided parameter `@process_version` identifies those vouchers that are currently being processed either in the ACM Processing or ACM Shift Voucher State activity or are being manually shifted to another state using ACM Entry Manager. A voucher with the `CSD_NOT_PROCESSED` flag set on will never reach FINAL state. The following code shows a simple customizing description:

```
update ACMVoucher set
      flags=flags-134217728,
      description='CSD'+description,
      version=version+1
where
      flags&134217728=134217728
      and process_version=@process_version
      and not voucher_state_id=1000 -- sanity check
      and ledger_id=@ledger_id
```

ACM automatically marks with the appropriate process version those vouchers which are currently being processed. The flag `CSD_NOT_PROCESSED` is assigned to the voucher when the voucher is created. The customization procedure defines whether the flag will be cleared during the shift to FINAL state, or earlier. The additional parameter `@target_voucher_state` provides the target voucher state of the processed vouchers.

For certain types of customization, it can be convenient to temporarily exclude certain vouchers from the processing. This can be achieved by using the voucher state PARKED. ACM does not attempt to move such vouchers to any higher state (you can only move such voucher to state OPEN via the Accounting Entry Manager, the **Reject** button). Additional custom functionality must be implemented (e.g. new activity) to un-park these vouchers – i.e. to set the UNPARKED state on such vouchers. ACM then allows shifting of such vouchers to FINAL state using either the Accounting Entry Manager (the **Accept** button) or via the Shift Voucher State activity.

The meaning of these special voucher states is summarized in the table below:

Visible ID in ACM	Simulation	State description
PARKED		By default a voucher cannot be posted to this state. You can post the vouchers to this state only by using the *CSD after posting*. Vouchers in this state can be viewed in the reports or Accounting Entry Manager, but they cannot be shifted to a higher state (neither by running the Shift Voucher State activity nor from the Accounting Entry Manager). You can only move such vouchers to state OPEN via the Accounting Entry Manager (the **Reject** button). Additional custom functionality must be implemented (e.g. new activity) to un-park these vouchers and to set them to the UNPARKED state, from which they can be shifted to FINAL either by running the Shift Voucher State activity or from the Accounting Entry Manager (the **Accept** button).
UNPARKED		By default a voucher cannot be posted to this state. You must use this state for vouchers previously posted to state PARKED (typically by *CSD after posting*). Additional custom functionality must be implemented (e.g. new activity) to un-park the PARKED vouchers and to set them to the UNPARKED state. Only from this state the previously parked vouchers can be shifted to FINAL. You can shift them either by running the Shift Voucher State activity or from the Accounting Entry Manager (the **Accept** button).

Note: ACM is installed with default `ACMCSDAfterPosting` procedure, but it only removes the flag `CSD_NOT_PROCESSED` from entries and vouchers.

Note: ACM provides Undo functionality, allowing you to undo-post and undo-map entries. When such process takes place, customizations of certain entries can take place more than once. The same applies to the Reject button in the Accounting Entry Manager. When a voucher is rejected, it is automatically assigned the flag `CSD_NOT_PROCESSED`, i.e. the voucher must be customized again. ACM may attempt to customize certain entries and vouchers more than once.

8.3.1.4 Customizations vs. Undo Accounting

None customization done by the BEFORE-MAPPING hook is the subject of the Undo Accounting Processing activity.

Those customizations, which are done by the BEFORE-POSTING hook on the newly created entries, e.g. entries resulting from the entries aggregation (Originating State = AGGREGATED) or from mapping actions split (Originating State = MAPPED), are removed since such entries are removed by the Undo Accounting Processing activity. The customizations done on the entries with Originating State NEW are not removed by the Undo Accounting Processing.

Very the same applies for the customizations done by the AFTER-POSTING hook. Any customization done on the simulated vouchers (voucher part) can be undone. Only changes done on the newly created entries (Originating State different from NEW) are removed by running the Undo Accounting Processing activity.

8.3.1.5 Using csd_auxiliary field

On the ACM entry level you can use a field csd_auxiliary for your specific development. The system does not use this field by default for anything. The field is of type numeric (19,0).

8.3.1.6 Using CSD Auxiliary flag

On the ACM entry level (ACMEntry.flags) as well as the ACM voucher level (ACMVoucher.flags) you can use specific flags dedicated just for the CSDs. They both have the same value, which is

1,073,741,824. These two flags will stay available for CSDs forever and will not be used by the system by default.

It is possible to define own labels for these two CSD flags be setting the **ACM_ENTRY_CSD_AUX_DESC** and **ACM_VOUCHER_CSD_AUX_DESC** ACM server environment variables.

These labels will be displayed everywhere in the system where the flag value appears (e.g. reports, Accounting Entry Manager, Balance monitor) by loading the configuration from the ACM server.

8.3.1.7 Troubleshooting

It may happen that you request a customization via the ledger ACM-CSD-xxx properties, but the appropriate customization procedures are defined incorrectly. The affected entries and/or vouchers get the flag "CSD not processed" and due to the incorrectly defined procedure this flag is not removed. The entries are stopped within the accounting processing. You can see this flag in the Accounting Entry State report. The following table shows how the failed entries are identified at the particular accounting processing stages:

Processing stage	Entry State	Voucher State	Flags (on entry)	Voucher Flags
Before mapping	NEW	-	CSD not processed	-
Before posting	MAPPED	-	CSD not processed	-
After posting	POSTED	VERIFY	-	CSD not processed

The only way, how to solve such situation is to correct the appropriate customization procedure.

If the customization is required on the level of a ledger (i.e. the appropriate ACM-CSD-xxx property is set to Yes), but the appropriate procedure is totally missing, the Accounting Processing activity fails with the following Failure Reason:

java.sql.SQLException: Could not find stored procedure ’ACMCSDBeforeMapping’

or

java.sql.SQLException: Could not find stored procedure ’ACMCSDBeforePosting’

or

java.sql.SQLException: Could not find stored procedure ’ACMCSDAfterPosting’

8.3.2 Customizing transaction zero-sweeping granularity

Zero-sweeping is a functionality supporting the booking of entries or balances to some account depending on a booking currency balance of some associated account. In other words, if the system posts some entry or balance to account A, it checks the booking currency balance on account B.

ACM supports three types of zero-sweeping differing in the granularity of checked balances:

- **Transaction zero-sweeping**
- **Position zero-sweeping**
- **Account zero-sweeping**

For details see the *ACM User Guide*: Using advanced operations: Applying zero-sweeping.

Important: The possible customization relates only the transaction zero-sweeping.

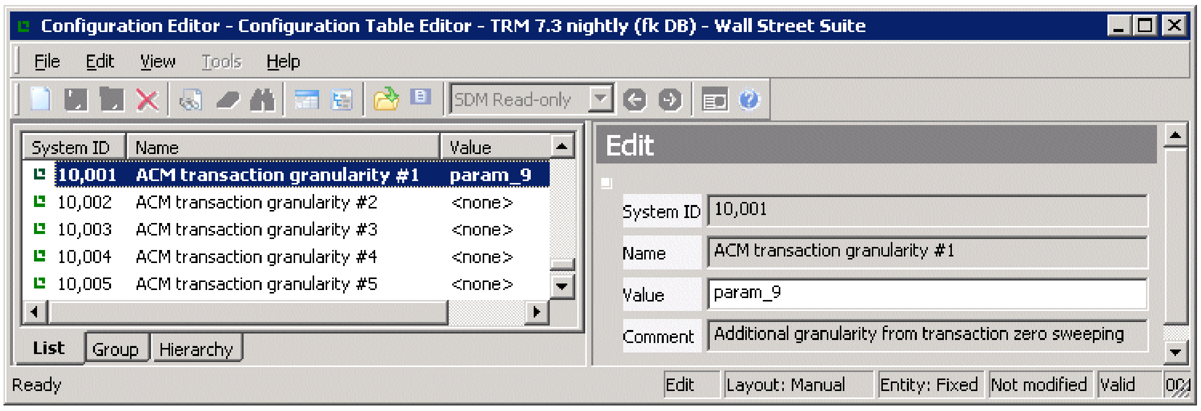
The rebooking between the zero-swept accounts is represented by entries, which are created in some default granularity. This chapter is about customizing the granularity of those rebooking entries. It is not about customizing the granularity of the swept account balances calculation.

The granularity of zero-sweeping entries for transaction zero-sweeping can be extended by 5 other dimensions.

For such customization the database table *Configuration* has 5 records allowing to add maximally 5 additional new dimensions. The names of these records are *ACM transaction granularity #X,* where *X* goes from 1 to 5. The customization should be done via the Configuration Table editor.

Since the editor does not allow to search by other field than value, it is recommended to sort the rows in the left hand part of the editor by System ID. Those 5 rows relating the transaction zero-sweeping customization have IDs from 10,001 to 10,005.

The editor allows to change only the field Value. By default it contains value <*none*>. Replace this value by the *ACMEntry* table column name, which should be added to the transaction zero sweeping granularity, e.g. param_9 as shown in the picture below.



System ID	Name	Value
10,001	ACM transaction granularity #1	param_9
10,002	ACM transaction granularity #2	<none>
10,003	ACM transaction granularity #3	<none>
10,004	ACM transaction granularity #4	<none>
10,005	ACM transaction granularity #5	<none>

Edit

System ID	10,001
Name	ACM transaction granularity #1
Value	param_9
Comment	Additional granularity from transaction zero sweeping

Note: Typically only a user with administrator rights can see the content in the Configuration editor.

8.3.3 Activity Parallelization

For a better performance, the Accounting Processing activity can be run in a multi threaded way if executed for a ledger group.

The Configuration table contains two lines where you can switch multithreading for the Accounting Processing activity. You can also define the number of allowed threads.

The lines are:

- **id = 10,007, ACM Activity Multithreading**

  By default false, set **true** for enabling the multithreading.

- **id = 10,008, ACM Activity Thread Count**

  By default 4.

You can change these configurations via the Configuration Table Editor. Just note that you must have appropriate permissions for modifying the Configuration table.

It is not necessary to restart the ACM Server after this change.

Once you change the "Use Threads in ACM Activities" configuration to true (System ID 10007), next Accounting Processing activity execution for a ledger group will be processed in the ACM Server multi threaded way. **The ACM Server internally splits all ledgers from the ledger group into a few sets of ledgers according the ledger owners**; these sets are then processed in parallel with the degree specified in the "Number of allowed threads" configuration (System ID 10008).

Any time the ACM Server runs the activity in the parallel mode, each record in the ACM Activity Log contains an Entity Key column populated by Ledger ID to which that record refers to.

Warning: Only ledgers having different owners may be processed in parallel!

8.4 Accounting reports

8.4.1 Customizing report parameters and output columns

8.4.1.1 Report files

It is possible to customize some ACM reports. You can enhance existing reports or create new ones. An ACM report is defined by two file types:

- Report type file (extension frd) - located on the client side in a directory defined by the variable FK_REPORT_TYPES. It defines connection to report definition, report parameters, column names.
- Report definition file (extension xml) - located on the server side, by default inside the war file (`inside WEB-INF\lib\DataModel-gen-<version>.jar and there in directory reports\xml`). It defines the Java class that executes report generator request, and other parameters.

For better customizing definitions they could be read from the external location specified by variable ACM_REPORT_DEFINITIONS_DIR (outside war file).

Reports can be customized as follows:

- Modify existing report

On client side - modify report type file (frd).

On server side - modify the report definition inside the war file, or create new XML file with the same name in the external location. Both definitions (from war and from external locations) are then merged. Properties from the definition in the external location have higher priority.

- Create brand new report

On client side - create new report type file (frd).

On server side - create new report definition file in the external location or inside the war file.

- Create new one based on existing report

On client side - create new report type file (frd).

On server side - create new report definition file in the external location. In the header of this definition (tag <report>) set property @parent-type="parent-report-definition". The property "parent-report-definition" is then merged with the new definition. Properties from the new definition have higher priority.

8.4.1.2 Customizing columns

You can customize columns only for reports executed by the following classes:

`com.trema.acm.reporting.sql.SQLReport`,
`com.trema.acm.reporting.java.TrialBalanceDetailReport` or
`com.trema.acm.reporting.sql.SQLProcedureReport`.

These reports use the database views or stored procedures to retrieve data. Depending on the used class you can do the following:

For `com.trema.acm.reporting.sql.SQLReport`

- **For adding a column**

Add a column to the view in the database and add the column to the report type (frd) on the client side. Column name must be in lower case although in the database its name contains capitals. No change to the server definition file is needed.

- **Removing a column**

  Remove the column from the view in database. No change on the server nor client side is needed.

For `com.trema.acm.reporting.java.TrialBalanceDetailReport`

- **Adding a column**

  Add column renaming in report type (frd) on client side. Column name must be in lower case although in the database its name contains capitals. On sever side create the report definition in an external location and add your new column definition to the tag <columns>. You can add any column available in ACMJournalView.

`For com.trema.acm.reporting.sql.SQLProcedureReport`

- **Adding a column**

  Add a column to the output of stored procedure and add column renaming in report type (frd) on client side. Column name must be in lower case although in the database its name contains capitals. No change to the server definition file is needed.

- **Removing a column**

  Remove the column from the output of stored procedure. No change to server or client files is needed.

Example - Adding an entry parameter 0 to the Trial Balance Detail report.

Create an external definition location file acm-trial-balance-detail.xml with the following content:

```
<?xml version="1.0" encoding="utf-8"?>
<report type="acm-trial-balance-detail">
  <columns>
    <column>
      <id>ACMEntry_param_0</id>
    </column>
  </columns>
</report>
```

8.4.1.3 Customizing startup parameters

You can customize startup parameters only for reports executed by classes `com.trema.acm.reporting.sql.SQLReport` or `com.trema.acm.reporting.sql.SQLProcedureReport`. These reports use the database view or stored procedure to retrieve data.

To add a new startup parameter do the following:

- Add the new parameter to the report type (frd) on client side
- Create in the external location the file <report-type>.xml. You can use the file from the war as a template.

For `com.trema.acm.reporting.sql.SQLReport`

- In the XML file, create the tag <report-conditions>. Specify the condition to be added to the end of the original one, or create the tag report-conditions with property @mode="overwrite" and specify the whole condition. Condition is a piece of SQL code and the query is constructed as "select * from <view> where <report-conditions>". In condition, you can use special operators such as #=, #<=. #>= (in the XML file you must use #&lt;=, #&gt;= syntax) which are converted to =, <=, >= if input parameter is not null or to (1=1) if null. If you use these enhancers, the parameter must be on the right side of the condition. This access significantly improves the performance of reports.

For `com.trema.acm.reporting.sql.SQLProcedureReport`

- In database add to proper stored procedure new starting parameter.
- In the XML file create tag <procedureParameters>. This tag must contain comma separated list of all parameters (constraints) what are sent to the stored procedure (in order as they are accepted by this procedure). Number of parameters must be exactly same as the number of parameters of procedure.

Example (SQLReport): how to add to the journal report a new parameter to allow filtering of rows based on Parameter 0.

- Change acm-journal.frd as follows:

```
Parameters=ACMLedger_id, ACMPeriod_id … ,AdditionalGrouping, ACMEntry_parameter0
..
..
[Parameter ACMEntry_parameter0]
Name=Entry Parameter 0
Type=TEXT
```

- Create an external definition location file acm-journal.xml with the following content:

```
<?xml version="1.0" encoding="utf-8"?>
<report type="acm-journal">
  <report-conditions>
    and (ACMEntry_param_0 #= :ACMEntry_parameter0)
  </report-conditions>
</report>
```

Example (SQLProcedureReport): how to add to the acm-off-balance-calculation-period report new parameter to allow filtering of rows based on Parameter 0.

- Change the stored procedure ACMReportOTFBalCalcPeriod to accept a new parameter parameter0 and a filter based on it.
- Change acm-journal.frd as follows:

```
Parameters=ACMLedger_id, ACMPeriod_id … ,AdditionalGrouping, ACMEntry_parameter0
..
..
[Parameter ACMEntry_parameter0]
Name=Entry Parameter 0
Type=TEXT
```

- Create an external definition location file acm-off-balance-calculation-period.xml with the following content:

```
<?xml version="1.0" encoding="utf-8"?>
<report type="acm-otf-balance-calculation-period">
  <procedureParameters>ACMLedger_id,ACMPeriod_id,ACMAccount_id,zero_balances,
  ACMVoucherState_id,ACMEntryOriginEntity_id1,number1,ACMEntryOriginEntity_id2,
  number2,ACMCostCenter_id,ACMProject_id,counterparty_code,elem_one,elem_one_value,
  elem_two,elem_two_value,report_type,ACMEntry_parameter0</procedureParameters>
</report>
```

8.4.1.4 Customizing postprocessors

Postprocessors allow you to modify fetched data before it is sent to the report generator. In ACM there are post-processors for translating bit flags to string representation, convert numeric system ID to human readable ID, to convert direct/indirect currencies, etc.

You can add a postprocessor to any report. Create the report definition in an external location and add your postprocessor definition to the tag <postprocessors>. The ACM Server then will use all the postprocessors from original report along with the new postprocessor.

Example: postprocessor translating entry flags to its string representation

```
<postprocessor class="com.trema.acm.reporting.FlagsPostProcessor">
  <sourceColumn>ACMEntry_flags</sourceColumn>
  <entity>ACMEntry.flags</entity>
</postprocessor>
```

8.4.2 Customizing maximum rows in report

The maximum number of displayed rows in the ACM reports is driven by the ACM server property **reports.maxRows**. This property is by default set to 100,000 and applies for all ACM reports.

If a report output has more rows than the specified maximum, the output is not provided and the following message is displayed: "*There are more rows meeting the input criteria. Current limits is 100,000 rows. Please, specify more restrictive input parameters. Note: Do not use this output as a final report for business purposes as its interpretation could be incorrect.*"

If you want to change that limit globally for all ACM reports, you must change that property and to restart the ACM server.

Or, you can change that limit per particular report by adjusting the appropriate report XML file by defining the tag <MaxRows>. For details about that XML file see *8.4.1.1 Report files* on page 63. If the tag is set to 0, the number of rows is unlimited. If the MaxRows tag is missing, the value from the ACM server property **reports.maxRows** is used.

Example for adjusting the MaxRows report tag

Let us assume you want to set the maximum number of fetched rows for the Accounting Journal report to 200,000 while for all other reports you want to keep a default value.

You must create an external definition location file acm-journal.xml with the following content:

```
<?xml version="1.0" encoding="utf-8"?>
  <report type="acm-journal">
  <MaxRows>200000</MaxRows>
</report>
```

8.4.3 Customizing drill-downs

Drill down functionality is a native functionality of the report generator. The configuration for ACM reports is stored in the acm_drilldown.ini in the directory specified by the variable FK_REPORT_LAYOUTS.

For details of customizing drill-downs, see *TRM System Administration Guide*.

8.4.4 Restricting list of offered ledgers

By default, all ACM reports offer in the selection list for ledgers all the ledgers defined in the system with the only restriction given by the domain check. If you want to restrict the list only to those ledgers for which the user has the appropriate ledger report permissions, you must change the configuration in the Configuration table, id 10009. Note that this configuration applies to the restriction of ledgers in Accounting Manager too.

Note: The only ACM report, for which no ledger permission verification is done, is the Accounting Dynamic Data Verification report (see details in *6.1 Accounting dynamic data verification report* on page 35).

8.5 Accounting Manager

8.5.1 Customizing drill-downs to the report generator

Drill-down definitions for Accounting Manager are stored on the client side in AccountingClient####.jar (`resources\ReportLaunches.xml`), where #### represents the Wallstreet Suite version number. If you want to add your own drill-downs, you must create the drill-down definitions in a file outside the jar file and set the environment variable ACM_AM_DRILLDOWN_DEFINITIONS be referring to this file.

Definitions are read during Accounting Manager startup and added to the end of drill-downs list. The format of the drill-down definition file is following:

```
<ReportLaunches>

<ReportLaunch>

  <Label></Label> - label of drilldown

  <Type></Type> - report type

  <Layout></Layout> - report layout - only one of type/layout tags can be used

  <SourceEntity></SourceEntity> - table where drilldown is active. Allowed values are
    ACMEntry, ACMVoucher, ACMMatchEntry.

  <SourceUse>

    <Use></Use> - if there are more tables (grids) based on same entity this allows
      you to specify in what drilldown is active
        - for ACMMatchEntry it could be USE_MATCHED, USE_POSITIVE, USE_NEGATIVE
        - for rest leave it empty

  </SourceUse>

  <Parameters>

    <Parameter SourceProperty=""|Constant="" Selection="all" Destination=""/>
    - parameters to be send to report
      - Destination is name of parameter in destination report
      - SourceProperty is name of property in source table
      - Constant - constant - e.g. 1000 for Voucher State FINAL
      - Selection="all" - if there is more selected rows in grid it creates comma
      separated list of values
      - AllowNull="true" - by default drilldown is enabled only if "SourceProperty"
        is not null. If this property is set to "true" then it is enabled although
        "SourceProperty" is null.

    </Parameter>

  </Parameters>

</ReportLaunch>

</ReportLaunches>
```

8.5.2 Customizing memory settings

You can set up memory usage limit in the *Accounting Manager*. By default the value is set to 128 MB. This value might not be high enough for large data amounts, such as *Import*/*Export* operations. In such cases, an *Out of memory* message will be shown. If this happens, use a text editor to open this file:

```
%FK_HOME%\share\java\acm\client\AccountingManager.ini
```

and change the value in the expression *-Xmx128m* to a bigger value (*-Xmx256m* and similar).

Note: Since the Accounting Manager runs as java process like the ACM server, you may look for relevant information about the memory settings to chapter *4.2 Memory settings* on page 27 too.

8.5.3 Restricting list of offered ledgers

By default, Accounting Manager offers in the selection list for ledgers all the ledgers defined in the system with the only restriction given by the domain check. If you want to restrict the list only to those ledgers for which the user has the appropriate ledger permission, you must change the configuration in the Configuration table, id 10009. Note that this configuration applies to the restriction of ledgers in ACM reports too.

8.6 Hedge Manager

Hedge Manager can be customized in the following areas:

- **Hedge relation flow**

  What are the possible hedge relation states and how a hedge relation can flow between the states.

- **Hedge Manager modes**

  You can define multiple Hedge Manager modes. For **each mode** you can define:

  – Available hedge relation states.

  – How particular states are restricted for editing within the mode.

  – What commands are available per particular state within the mode.

- **Hedge Managers per result modes**

  A possibility to view the hedge relations just from one specified result mode (e.g. IFRS, FAS, Local, etc.).

This chapter describes how to customize all these areas.

8.6.1 Hedge relation flow

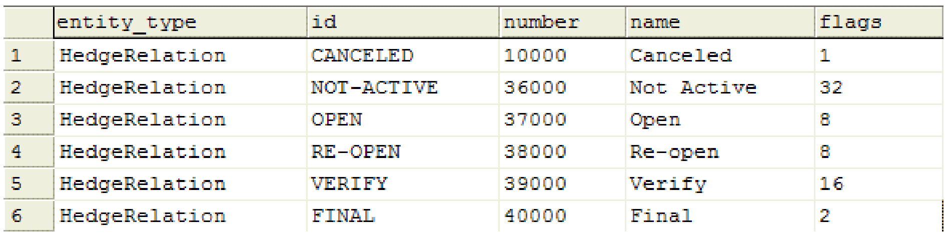
The process of hedge relation handling and control, from initial entry to final acceptance, is referred to in ACM as a hedge relation flow. The hedge relation flow logic is very close to the transaction flow logic available in TRM.

Different commands available in the Hedge Manager can be configured for different modes and hedge relation states.

8.6.1.1 Default flow

At all points in the hedge relation flow, a hedge relation has a state.

The states are defined in the EntityState table:



	entity_type	id	number	name	flags
1	HedgeRelation	CANCELED	10000	Canceled	1
2	HedgeRelation	NOT-ACTIVE	36000	Not Active	32
3	HedgeRelation	OPEN	37000	Open	8
4	HedgeRelation	RE-OPEN	38000	Re-open	8
5	HedgeRelation	VERIFY	39000	Verify	16
6	HedgeRelation	FINAL	40000	Final	2

As you can see, each state has some flag. Their meaning is following:

- 1 = cancelled
- 2 = final
- 8 = provisional
- 16 = intermediate
- 32 = not active

Note that hedge relation states are closely linked to the commands available in the Hedge Manager application. This link is defined via the hedge manager modes. For details see *8.6.2 Hedge Manager modes* on page 73.

8.6.1.2 Flow customization

The hedge relation flow configuration is driven from two files:

- `hedge_relation.sql` located in $FK_HOME/share/<database>/setup folder.

  The script consists from several calls of another procedures. The procedure related to the hedge relation flow definition **SetupEntityState**, which defines the particular hedge relation states.

- `hedge_relation.py` located in $FK_HOME/share/python folder.

  This file defines the state flow via agents, i.e. it defines next state for a specified state and applied action. The hedge relation flow can be conditions based, i.e. a hedge relation can be sent to some state only if some condition is fulfilled. For details see the next chapter.

The default versions of these two files represent the default flow as shown above.

By customizing these files (and executing the appropriate SQL commands) you can customize the hedge relation flow.

Warning: The flow definition is very closely link to the definition of modes. Read *8.6.2 Hedge Manager modes* on page 73 for details!

Note: Since the hedge relation flow logic is very close to the flow logic implemented for transactions in TRM, you can refer for many details to the *TRM System Administration Guide*, chapter Transaction flow.

8.6.1.3 Conditioned flow

The hedge relation flow can be conditions based, i.e. a hedge relation can be sent to some state only if some condition is fulfilled. For instance, you can define some flow shortcuts (e.g. direct movement from OPEN to FINAL) if some conditions are fulfilled. Such conditions can be used for defining the exceptions in the hedge relation flow.

The conditions are represented by special entity rules stored in two database tables:

- EntityRulesHeader
- EntityRules.

Warning: The tables **must not** be filled by direct database inserts! They can be filled by executing a Python script only, e.g. you can adjust and execute the `hedge_relation.py` script. Note that this file is always replaced by new hedge_relation.py file from the WSS package.

The data in the column *value* of the EntityRules table are kept in certain internal format and cannot be read. Therefore you cannot read easily the structure of the rules and the definition in the Python file is the only place from where you can read them.

A Python syntax for defining the condition rules is as follows:

```
from entity_agents import *

define_rule ("HedgeRelation", "rule_id", "rule_name", (

   ("column_id", "value"),

   ("column_id", "value"),

   ...

   ))
```

where:

Column id	Column meaning
rule_id	Mandatory field. Unique key. Each rule must have its id.
rule_name	Optional field. You can define some name for the rule too.
column_id	Mandatory field. In this field you specify the column name of the field from the HedgeRelation table, e.g. strategy_code, param_0, param_1, etc.
value	Mandatory field. In this field you specify a value, which should be filled for the particular column defined above, so the rule is considered as a matching rule. Note that as value you can specify also non-string values, e.g. for column `critical_terms_match` the specified value could be 0 (does not match, i.e. No) or 1 (does match, i.e Yes). In such a case the value in the Python script would not be specified in double quotes.

Note: The `hedge_relation.py` file is located in $FK_HOME/share/python folder and you can execute it from the Wallstreet Suite Shell by command `python hedge_relation.py`.

Entity rules might be then referenced in the hedge_relation.py file, where you define for

- what action,
- what initial state, and
- under what condition (rule and not_rule),

what target state is allowed.

Example

Let us assume that you want to skip the state VERIFY for hedge relations with strategy code SIMPLE_FX not having a value Test in the hedge relation parameter 1 and not having a value Test2 in the hedge relation parameter 2.

The `hedge_relation.py` file must be modified as follows:

```
from entity_agents import *

define_rule ("HedgeRelation", "SC-SIMPLE_FX", "", (

   ("strategy_code", "SIMPLE_FX"),

   ))
```

```
define_rule ("HedgeRelation", "P1-TEST", "", (
    ("param_1", "Test"),
    ("param_2", "Test2")
    ))

# Move forward in the flow.
define_op ("HedgeRelation", "ACCEPT", "Accept Hedge Relation", (
    (state('NOT-ACTIVE'), action_mask(0), not_mask(0),
    set_state('OPEN', 'Accept')),
    (state_between('OPEN', 'RE-OPEN'), action_mask(0), not_mask(0),
    rule('SC-SIMPLE_FX'), not_rule('P1-TEST'),
    set_state('FINAL', Verify')),
...
```

8.6.1.4 Calling CSDs from the flow

The flow configuration in the `hedge_relation.py` file allows to define CSDs which can be executed at the moment of hedge relation state transitions. The CSDs should be created as stored procedures which are called by command

    call_proc('Procedure_name')

Hedge ID is a key parameter provided to the procedure.

Some example is shown below. It calls the procedure HAMAcceptCSD from ACCEPT action:

```
# Move forward in the flow.
define_op ("HedgeRelation", "ACCEPT", "Accept Hedge Relation", (
...
    txn_begin(),
    service('service/entity-board/hedge-manager@validate'),
    service('service/entity-board/hedge-manager@save'),
    service('service/entity-board/hedge-manager@check-designation'),
    call_proc('HAMAcceptCSD'),
    txn_commit(),
    ))
...
```

The CSD procedure examples are provided in the next two sub-chapters

8.6.1.4.1 Sybase_HAMAcceptCSD.pl

```
&header ('$wss$ ');
use utf8;
```

```
&procedure ('HAMAcceptCSD', '
Make CSD actions when accepting a hedge relation.',
            '@id', 'ID',
            '@state_id', 'StateId',
            '@current_state_id', 'StateId',
            '@action_id', 'varchar(32)', '= null',
            '@date', 'datetime', '= null',
            '@param0', 'varchar(32)', '= null',
            '@param1', 'varchar(32)', '= null',
            '@param2', 'varchar(32)', '= null',
            '@param3', 'varchar(32)', '= null',
            '@param4', 'varchar(32)', '= null',
            '@param5', 'varchar(32)', '= null',
            '@param6', 'varchar(32)', '= null',
            '@param7', 'varchar(32)', '= null',
            '@param8', 'varchar(32)', '= null',
            '@param9', 'varchar(32)', '= null',
            '@stamp', 'timestamp', '= null',
            'as
-- Put the CSD code here
return ', &status_OK, '
');
&grant ('execute');
1;
__END__
```

8.6.1.4.2 Oracle_HAMAcceptCSD.pl

```
&header ('$wss$ ');
use utf8;
&procedure ('HAMAcceptCSD', '
Make CSD actions when accepting a hedge relation.',
            'Pid IN numeric',
            'Pstate_id IN varchar2',
            'Pcurrent_state_id IN varchar2',
            'Paction_id IN varchar2 := null',
            'Pdate IN date := null',
            'Pparam0 IN varchar2 := null',
            'Pparam1 IN varchar2 := null',
            'Pparam2 IN varchar2 := null',
            'Pparam3 IN varchar2 := null',
            'Pparam4 IN varchar2 := null',
            'Pparam5 IN varchar2 := null',
            'Pparam6 IN varchar2 := null',
            'Pparam7 IN varchar2 := null',
            'Pparam8 IN varchar2 := null',
            'Pparam9 IN varchar2 := null',
            'Pstamp IN number := null',
            ' RETURN NUMBER  as BEGIN SYBEMU.pushProcName(\'HAMAcceptCSD\');
-- Put the CSD code here
SYBEMU.popProcName;
return ',&status_OK,';END;');
&grant ('execute');
1;
__END__
```

8.6.2 Hedge Manager modes

Different modes of the Hedge Manager can be configured for different purposes.

You can define multiple Hedge Manager modes. For **each mode** you can define:

- Available hedge relation states.
- How particular states are restricted for editing within the mode.
- What commands are available per particular state within the mode.

8.6.2.1 Hedge relation states vs. Hedge Manager commands

The definition of available commands per each hedge relation state is fully configurable. However, you should always keep in your mind the following rule:

Commands changing a hedge relation can be performed only in states Open, Re-open or Not Active.

The commands changing a hedge relation are:

- Delete Hedge Relation
- Add Hedge Relation Risk
- Add Hedge Key Figure
- Add Hedge Relation Type
- Dedesignate Hedge Relation (Full)

- Dedesignate Hedge Relation (Partial)
- Dedesignated Transaction
- Adjust Discharge Date

All other commands can be performed at any state. However, for some hedge manager modes, e.g. QUERY or CANCELLED, some additional commands restrictions are applied. For details see the next chapter.

8.6.2.2 Default modes

In the default ACM installation, different Hedge Manager modes are configured by default. They are:

- FINAL (only for state Final)
- VERIFY (only for state Verify)
- NOT-ACTIVE (only for state Not Active)
- CANCELLED (only for state Cancelled)
- OPEN (only for states Open and Re-Open)
- ADMIN
- QUERY

For details see the ACM User Guide.

8.6.2.3 Modes customization

The modes are primarily defined in the sql script located in
$FK_HOME/share/<database>/setup/hedge_relation.sql.

Note: It is the same file as for defining the hedge relation states.

The procedure consists from several calls of another procedures. The procedures related to the modes definition are:

- **SetupEntityMode**

  It defines the Hedge Manager modes (mode_id) including:

  - The definition of states visible in the mode.
  - Default behavior for fields, where **grant_grant_p** = 1 means all fields are frozen, zero enables all.
  - Default available commands and Flow actions, where **action_grant_p** = 1 means all commands (actions) as well as shifts in flow are disabled, zero enables all.

  All these defaults can be adjusted by the next procedure.

- **SetupModeAction**

  It specifies the exceptions from the default behavior defined by the SetupEntityMode procedure, for instance, for mode FINAL you enable all the commands (@action_grant_p = 0), but then you disable for this mode all the commands listed in *8.6.2.1 Hedge relation states vs. Hedge Manager commands* on page 73.

The particular modes can be used for starting the Hedge Manager. The definition is done via the FKApplicationManagerMenu.xml file.

Note: Since the modes logic is very close to the modes logic implemented for transaction flow and Transaction Manager in TRM, you can refer for many details to the *TRM System Administration Guide*, chapter Transaction flow.

8.6.3 Hedge Managers per result modes

If no result mode is specified in the Configure Hedge Manager dialog, the Hedge Manager allows to fetch and display the hedge relations for all the result modes at once. In order to start the Hedge Manager in specific result mode only, an extra parameter "--group" has to be passed to the appropriate hedge manager starting command used by the Application Manager. You do this customization in the FKApplicationManagerMenu.xml file.

Chapter 9 Interfacing with ERP

ACM is typically used as a sub-ledger from which you typically export the entries or account balances to a target ERP system for further processing and reporting. Such export functionality in ACM is called as ERP Interface.

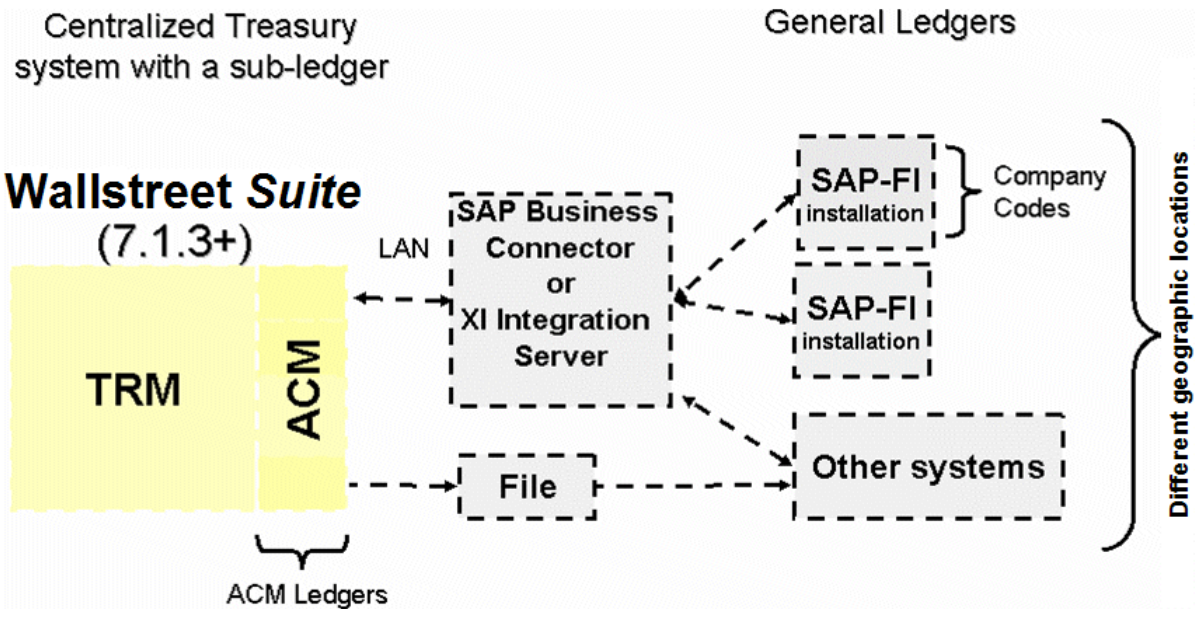
This chapter provides information related to this interface.

Note: ACM has integrated interface with other Wallstreet Suite modules - TRM and CMM. After successful ACM installation these interfaces does not require any special administration tasks.

9.1 Technical overview

This chapter provides a technical overview of the ACM/ERP interface. For a business aspects look to the ACM User Guide.

The overall technical architecture of the ERP interface looks as follows:


Centralized Treasury system with a sub-ledger
General Ledgers
Wallstreet Suite (7.1.3+)
TRM
ACM
ACM Ledgers
LAN
SAP Business Connector or XI Integration Server
File
SAP-FI installation
SAP-FI installation
Company Codes
Other systems
Different geographic locations

As you can see there are two methods of an integration:

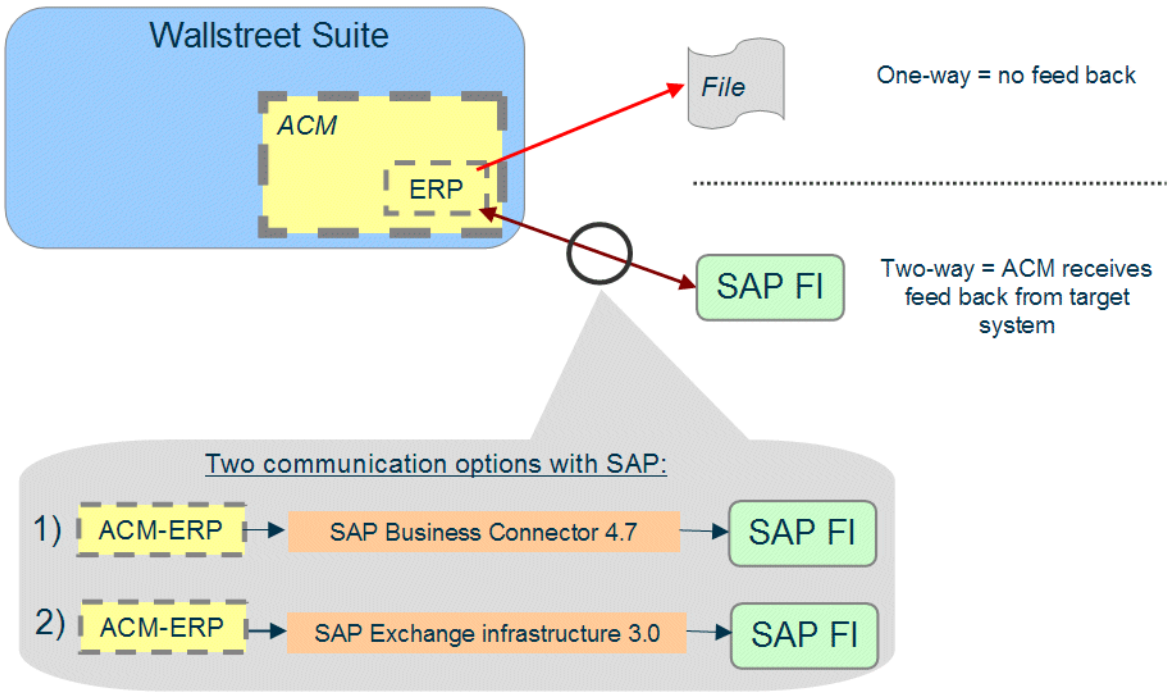
- FILE based
- SAP_FI based

The method is specified by selecting a target type in the ERP System Definition editor. While the FILE target type besides the definition of the ACM_ERP_FILE_OUTPUT_PATH variable does not require any other special technical configuration, the SAP_FI target type requires a lot of configuration on the side of ACM as well as SAP.

The next chapter provides a very brief overview of that ACM/SAP set-up.

9.2 Interfacing with SAP

For SAP_FI (connection to SAP system) we provide two methods of communication as follows:


Wallstreet Suite
ACM
ERP
File
One-way = no feed back
SAP FI
Two-way = ACM receives feed back from target system
Two communication options with SAP:
1) ACM-ERP
SAP Business Connector 4.7
SAP FI
2) ACM-ERP
SAP Exchange infrastructure 3.0
SAP FI

Based on system variable ACM_ERP_SAP_CONNECTION system uses SAP Business Connector (ACM_ERP_SAP_CONNECTION = BC) or SAP Exchange Infrastructure (ACM_ERP_SAP_CONNECTION = XI).

ACM ERP interface is compatible with the following SAP versions:

- **R/3**

  4.6, 4.7

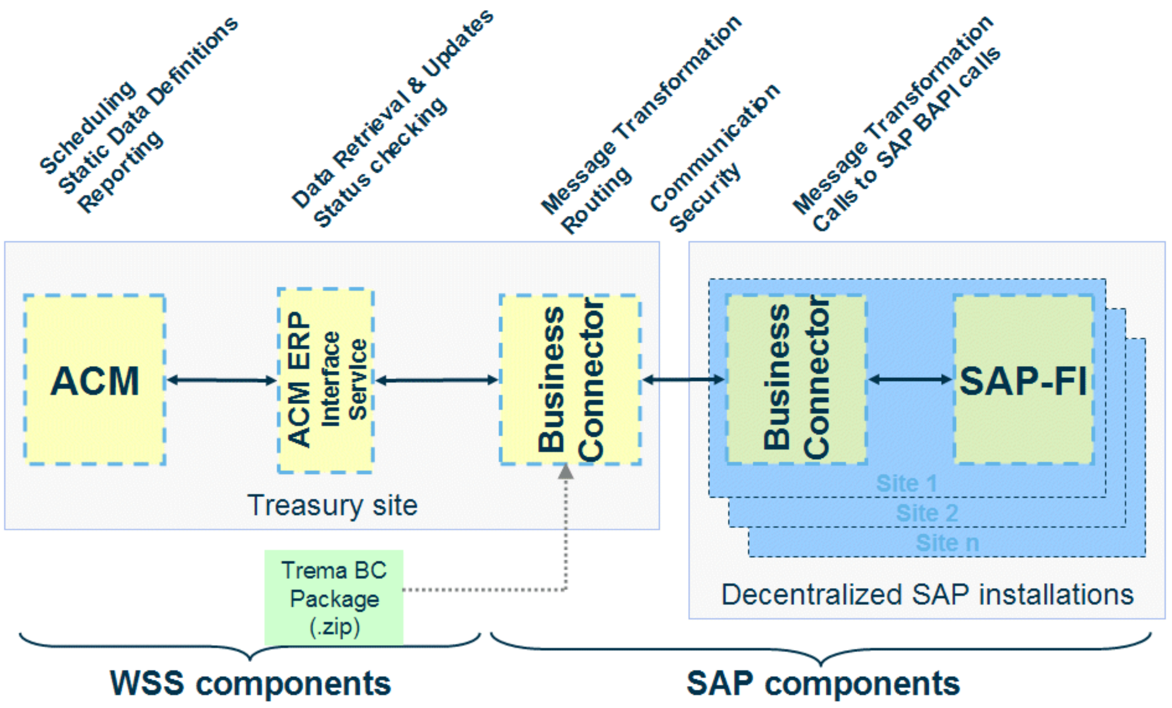
- **ERP Components:**

  5.0, 6.0

For details about the ACM/SAP interface configuration see the *ACM to SAP Configuration Guide*, which is distributed upon request (it cannot be downloaded from the Wall Street Systems Customer Support site http://www.trema.com/support).

9.2.1 Using SAP Business Connector

If you decided to use SAP Business Connector (BC) for connection to SAP system, then you must perform a special step required during the installation. The following picture shows the connection where SAP BC is used:

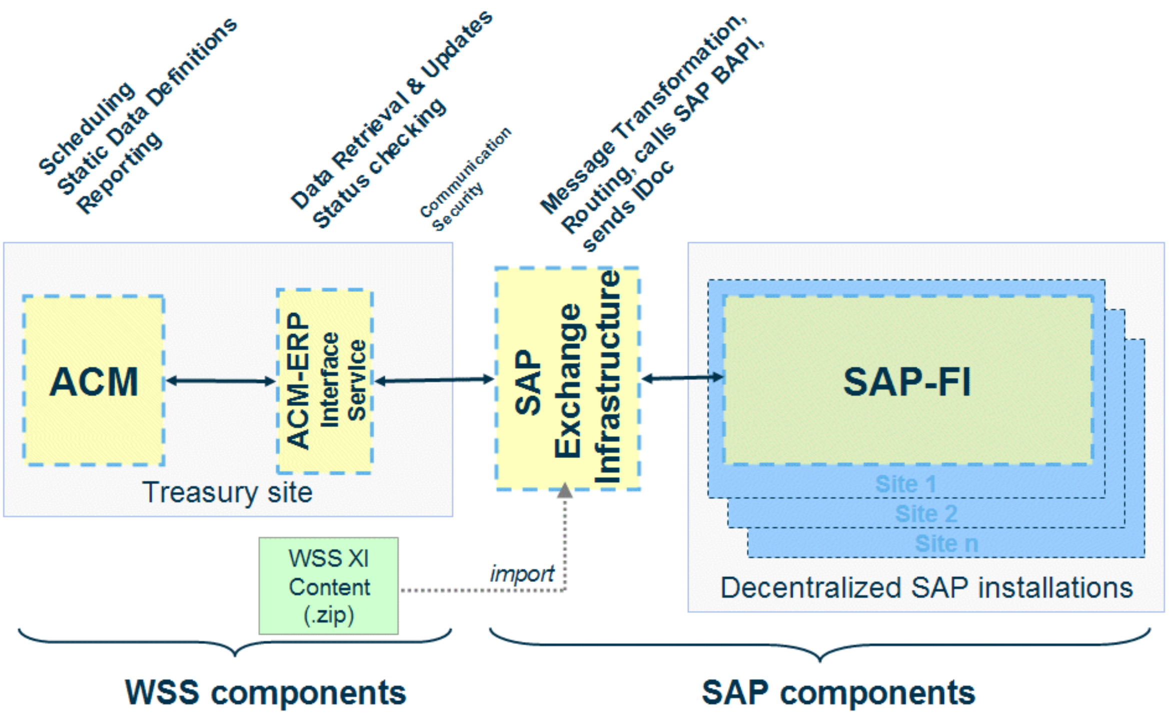


The BC Package is a zip file which is provided on a request. It inclused also a documentation for importing this package into SAP Business Connector including the description of required entities setup in SAP R/3 system.

9.2.2 Using SAP Exchange Infrastructure

If you decided to use SAP XI all libraries are packaged with ACM so there are no other libraries necessary comparing to SAP BC.

A complete overview of the technical architecture is provided by the following picture:



The XI Content *.tpz is provided on request together with documentation how to install and set all required objects in SAP R/3 system.

Communication with the SAP Exchange Infrastructure uses the HTTP protocol.

9.3 ERP Document Key adjustment

This section explanes a mechanism of ERP Document Key generation, its usage including a description of required manual adjustments in situations when moving/migrating the WSS databases across the environments.

9.3.1 ERP Document Key mechanism

When the ACM ERP Interface performs the export, it always packages a set of accounting entries into the ERP Document. Each such document has two IDs:

- **ID**

  This is an internal ID of a document within ACMERPDoc table and is used in many ERP Interface reports in ACM

- **ERP Document Key**

  "External" ID by which the document can be looked up in the target system, e.g. in SAP

The interface keeps one independent sequence of the ERP Document Key for each ERP System Definition and per other entities specified in the Document Key Format (like fiscal year if $y macro is used). When a new document is being created a new ERP Document Key is generated using the Document Key Format and Document Key Prefix defined for the Target.

Note: Setup of the Document Key Format should be done in cooperation with the ERP system administrator.

9.3.2 ERP Document Key example

If you define the Document Key Format as "$y_$I5" and Document Key Prefix as "T1_", the ACM Server will generate Document Keys for 2008 period as follows:

T1_2008_00001

T1_2008_00002

…

T1_2008_99999

If counter specified by $I is consumed, an exception is thrown and no new documents are created unless you modify the Format, you change the ACM ERP System Definition or update the ACMNumberSequence table according to the instructions mentioned later in this document.

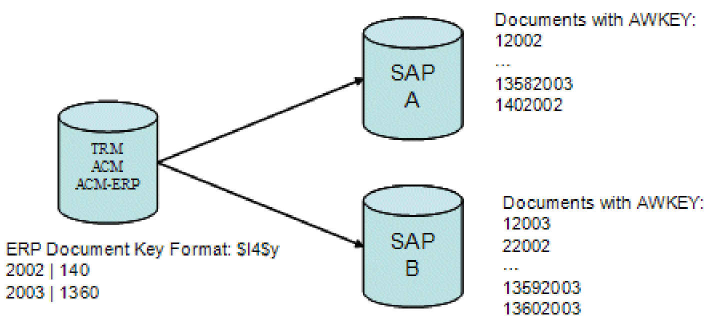
9.3.3 Possible problems if exporting to SAP

9.3.3.1 WSS database vs. SAP containing some previous exports

9.3.3.1.1 Problem description

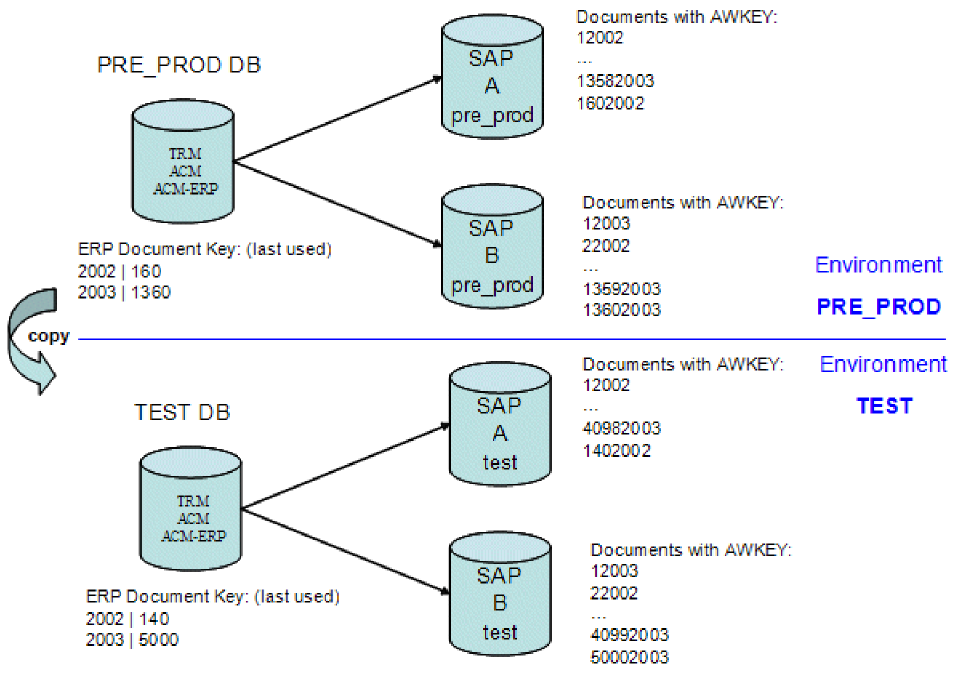
As long as there is a new installation or reinstallation of the ACM ERP Interface that exports data to SAP installations to which the ACM ERP interface did not export any data before, there is no problem and nothing has to be adjusted.

The following diagram shows an example of such environment after posting 1500 documents from the WSS System.



However, during the implementation phases, it happens that WSS databases are migrated from one environment to another. Let us assume the following setup:

When a copy of the PRE_PROD DB is loaded into the TEST environment and ERP-TARGETs of Ledgers are reconfigured to the SAP TEST machines, then if there is no adjustment performed, the interface may not work properly in the case that higher ERP Document Keys have already been used in the TEST environment. See the picture below:



A problem is that when the TEST is restarted with the copy of the PRE_PROD database, the interface would export a new document for fiscal year 2003 with the ERP Reference Key 13612003; but such a document may already exist in the target SAP! And if the new document contained an error and it was not posted by SAP, SAP would reply that it has been posted! A reason for such behavior is that the ERP Document Key is the input parameter to the SAP function which provides information about a posting success/failure and a document with this key may be already posted from the "old" test environment.

9.3.3.1.2 Problem solution

A solution for previously described problem consists from two steps:

1. Find out the highest used keys in the "old" environment

   Option A (setup on WSS ACM):

   – Retrieve some information from the WSS database before the new DB is loaded by running the following select:

     ```
     select sd.system_name,ns.* from ACMNumberSequence ns join ACMERPSystemDef sd
     on (ns.system_definition_id=sd.id) where ns.system_definition_id is not null
     order by ns.id
     ```

     You should see columns "system_name","id", "sequence_format" and "last_index". Store values and once you load the new WSS database, execute the same command and align the values appropriately.

   Option B (setup in SAP):

   This option is needed when the previous WSS DB, that was linked to the SAP systems, is no longer available:

   – Within each target SAP system find a maximum value of AWKEY (or XBLNR which is equal) for each fiscal year that has been posted into. An easy way to do this is:

   a. SAP transaction SE16, Enter

   b. Table name: BKPF, Enter

c. GJAHR - enter Fiscal Year, AWTYP - enter ZGLM (to filter document only from Trema), Execute
d. Sort by BELNR (because XBLNR is sorted as text, not numbers)
e. Get the highest (last) AWKEY/XBLNR
– For each fiscal year, take the maximum from all individual SAP systems

2. If the value for any of the fiscal years found in step 1 is higher than the value in the new DB, you must do the following:
   a. update the fiscal year by running

      ```
      update ACMNumberSequence set last_index=xxx where id=xy,
      ```

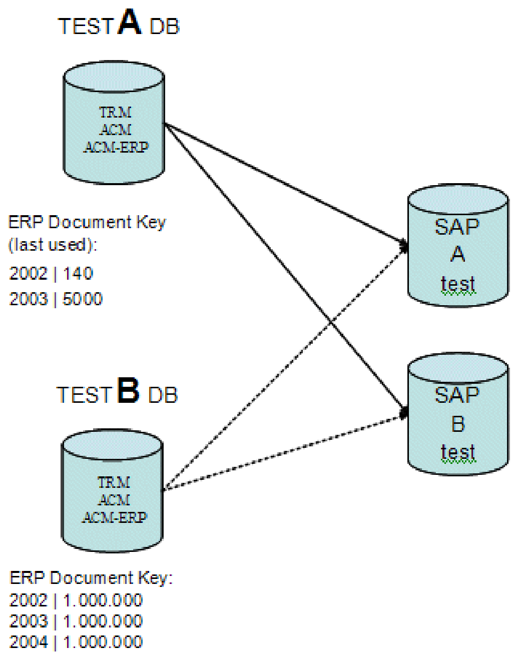
      where *xxx* represents new last_index you need and xy represents id of a row which you need to align.
   b. or inserted new line if it does not exist (set the STEP attribute to 1).

Warning: Any manual update in the ACMNumberSequence table can be done just when the ACM Server is stopped!

9.3.3.2 Multiple WSS to one SAP

9.3.3.2.1 Problem description

There might be a requirement that multiple WSS environments are linked to a shared SAP installation. This is shown on the next diagram:



In such case there might be a conflict in generated Document Keys.

9.3.3.2.2 Problem solution

When linking the second WSS environment - TEST"B" (with ACM ERP Interface) - to the same SAP systems, a possible solution is to "separate the ERP Document Key namespace" before starting to use the ACM ERP interface by moving the appropriate values in the ACMNumberSequence table (see the previous problem solution for details) to numbers that will not be reached from the TEST"A" WSS database.

Warning: This scenario can become dangerous when DB migrations into TEST"A" or TEST"B" are performed because it might not be easy to find the maximum numbers of the keys from the SAP target systems for each fiscal year and each "namespace" as described in the previous Option B problem solution.

In any case, it is essential that all such decisions taken for the environments are carefully documented by the implementation project.

9.4 Using Targets 2 and 3 in Account Mode

If you want to use Target 2 or 3 in the Account Mode, from the performance reasons it is recommended to build additional index using the following command:

```
create unique index amerp1XY on ACMEntry
(ledger_id,account_id,dr_cr,erp_target_XY,cost_center_id,project_id,voucher_id, id)
```

where XY is 2 if using Target 2 and is 3 if using Target 3.

9.5 Customizations

A default output of the ERP interface (the ERP document) produced by the ACM server is called TremaXMLDocument. Its structure is described in *Appendix C ERP document structure* on page 135.

Under the normal circumstances there is no need to modify the default TremaXMLDocument structure since it contains all the fields from a voucher and entries. However, sometimes a further processing of the ERP documents requires a modified structure. For such situations the ERP interface offers several places where the TremaXMLDocument can be modified. They are:

- XSL Transformation.
- Customization of reports used for the export of ERP Document and ERP Document Items.
- Customization outside the ACM Server.

The customization outside the ACM Server is not a subject of further description since it is only a client specific; the first two options are described in the following 2 sub-chapters.

9.5.1 XSL transformation

The customization of the TremaXMLDocument can be done by the XSL transformation (XSLT). Using this approach you can define an XSL file, which is attached to the system definition via the ERP System Definition Editor. Once you set this XSL file, the ACM Server processes all the ERP documents (in XML form) for given system definition by the specified XSLT. This transformation might be applied to both target types - SAP-FI as well as FILE.

Using XSLT you can freely modify the output format. If no XSL file is defined for the used system definition, the generated ERP documents have the structure of the default TremaXMLDocument.

The XSLT takes a place inside the ACM server. For keeping the ACM server stable we strongly recommend to keep in mind the following tips:

- **Memory**

  The XSLT transformation performed on huge ERP documents (like documents with hundreds items) is a memory very intensive task. Therefore:

- Keep the number of document items small enough.
- Assign enough memory to the ACM server.
- Manage the XSLT outside of the ACM server.

- **Character set**

  Keep a character set aligned and correctly set over all the systems. Especially if you use non-US characters, align a character set across all the servers and PCs accessing the system. ACM server is running in UTF-8 normally. In this case the ACM server process on the unix style operating systems should have set the LC_CTYPE or LC_ALL variables to value "en_US.UTF-8", otherwise an exception occurs during the XSL transformation. To ensure a specific output code page, use the standard <xsl:output … encoding="UTF-8"/>.

- **XSL file**

  The XSL file path defined in the ERP System Definition Editor should be accessible by the ACM server process. The ACM server must have a read permission for the XSL file.

Obviously there are some functional restrictions if the XSLT is applied for the SAP-FI target type: the ERP document after the applied XSLT has to be aligned with the interfaces implemented on the BC or XI servers. On the other hand there are no functional restrictions for customizing the ERP documents using the XSLT for the target type FILE.

9.5.2 Customization via reports

Once the ERP document is exported by the ERP Export activity the document waits in the queue inside the ACM server to be processed. Once this happens the ACM server fetches the ERP document and its ERP Document Items from the database using the standard ACM reports, which can be customized. See the table below for details.

Report	View	Usage
acm-IS-doc	ACMERPISDocView	This report and the underlying database view is used for the ERP document fetch. The returned result from this report determines header fields of final ERP Documents.
acm-erp-document-item	ACMERPISItemsView	This report and the underlying database view is used for the ERP Document Item fetch. Output from this report determines the item fields. If you need more or less columns to be exported on the item level, you can modify this report for achieving this.

9.5.3 Support for special characters in the ERP document

There are two system variables recognized by the ACM Server:

- ACM_ERP_XI_OUTPUT_ENCODING
- ACM_ERP_FILE_ENCODING.

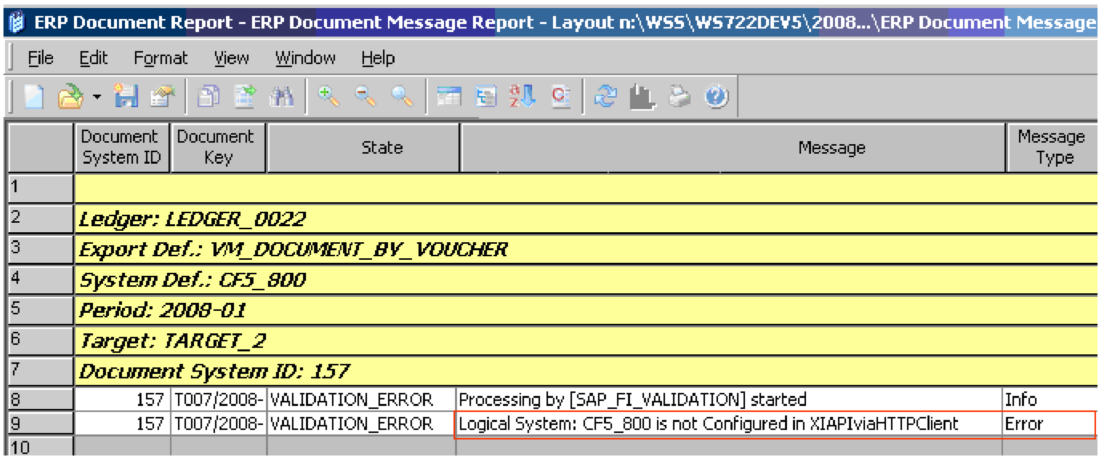
Administrator can specify encoding to be used for XI request and the file format. Default encoding for XI is ISO-8859-1. Default encoding for file is UTF-8.

9.6 Troubleshooting

9.6.1 Logical System: *XY* is not configured in XIAPIviaHTTPClient

9.6.1.1 Symptoms

In the ERP Document Report you can see the following error:



ERP Document Report - ERP Document Message Report - Layout n:\WSS\WS722DEV5\2008...\ERP Document Message

File Edit Format View Window Help

	Document System ID	Document Key	State	Message	Message Type
1					
2	*Ledger: LEDGER_0022*				
3	*Export Def.: VM_DOCUMENT_BY_VOUCHER*				
4	*System Def.: CF5_800*				
5	*Period: 2008-01*				
6	*Target: TARGET_2*				
7	*Document System ID: 157*				
8	157	T007/2008-	VALIDATION_ERROR	Processing by [SAP_FI_VALIDATION] started	Info
9	157	T007/2008-	VALIDATION_ERROR	Logical System: CF5_800 is not Configured in XIAPIviaHTTPClient	Error
10					

If this error occurs you can see the following in the ACM server log:

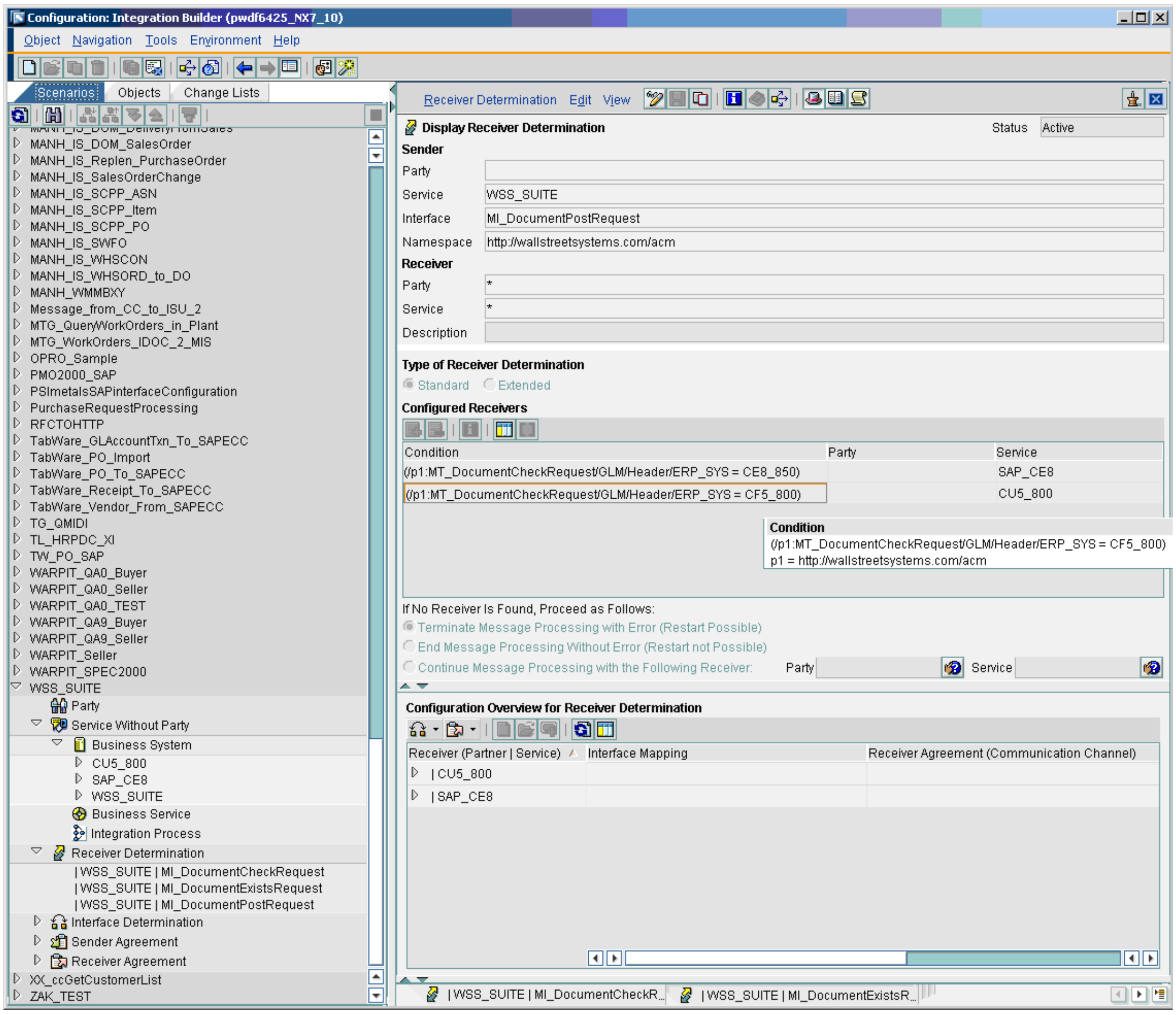
```
INFO [Thread-48[SAPValidationTask]] 05 Feb 2009 14:01:27,592 (XIAPIviaHTTPClient.java:293) -
Problematic System:CF5_800 Document:152

 WARN [Thread-48[SAPValidationTask]] 05 Feb 2009 14:01:27,592 (XIAPIviaHTTPClient.java:294) - No
receiver could be determined(RCVR_DETERMINATION.NO_RECEIVER_CASE_BE)

 WARN [Thread-48[SAPValidationTask]] 05 Feb 2009 14:01:27,592 (XIAPIviaHTTPClient.java:296) -
Response:HTTP/1.1 500 Internal Server Error|500
null
 INFO [Thread-48[SAPValidationTask]] 05 Feb 2009 14:01:27,592 (XIAPIviaHTTPClient.java:297) -
URL:http://172.24.64.48:50080/sap/xi/adapter_plain?namespace=http://wallstreetsystems.com/acm&int
erface=MI_DocumentCheckRequest&service=WSS_SUITE&qos=BE
 INFO [Thread-48[SAPValidationTask]] 05 Feb 2009 14:01:27,592 (XIAPIviaHTTPClient.java:298) -
<?xml version="1.0" encoding="ISO-8859-1"?>
<ns0:MT_DocumentCheckRequest xmlns:ns0="http://wallstreetsystems.com/acm">
      <GLM><Header><ID>152</ID><OBJ_KEY>T002/2008-
S</OBJ_KEY><DOC_DATE>20090202</DOC_DATE><PSTNG_DATE>20080101</PSTNG_DATE><FISC_YEAR>2008</FISC_YE
AR><FIS_PERIOD>01</FIS_PERIOD><COMP_CODE>1000</COMP_CODE><ERP_SYS>CF5_800</ERP_SYS><ERP_TYPE>SAP_
FI</ERP_TYPE><GLM_OWNER>LEDGER_0022</GLM_OWNER><STATE_ID>150</STATE_ID><EXP_USER>MCEKAL</EXP_USER
><DOC_TYPE>1</DOC_TYPE><DOC_FLAGS>0</DOC_FLAGS><LEDGER_ID>55</LEDGER_ID><ACM_PERIOD>2008-
01</ACM_PERIOD></Header>
<Item><DOC_CURR>USD</DOC_CURR><DOC_AMT>-1000.0000</DOC_AMT><BOOK_CURR>EUR</BOOK_CURR><BOOK_AMT>-
800.0000</BOOK_AMT><ERP_ACC>0000230000</ERP_ACC><GLM_ACC>3111-
C</GLM_ACC><GLM_ACC_ID>759</GLM_ACC_ID><GLM_ACC_TXT>Gain from balance revaluation -
C</GLM_ACC_TXT><ENTRY_ID>138104</ENTRY_ID><VOUCH_ID>BR-2008-
0002</VOUCH_ID><V_ID>52602</V_ID><VOUCH_TYPE>BAL-REVAL</VOUCH_TYPE><TRAN_NUM>ACM-
2502</TRAN_NUM><BOOK_DATE>20090202</BOOK_DATE><VAL_DATE>20080101</VAL_DATE><ENTRY_DESC>Balance
Revaluation</ENTRY_DESC><ORIG_ID>ACM-
TRANSACTION</ORIG_ID><ORIG_CODE>ACM</ORIG_CODE><DU_ID>152</DU_ID><DI_ID>153</DI_ID><DR_CR>-
1</DR_CR><DR_CR_NAME>CR</DR_CR_NAME><D_ID>152</D_ID><ITEM_TYPE>BY_VOUCHER</ITEM_TYPE><UNIT_TYPE>B
Y_VOUCHER</UNIT_TYPE><VOUCH_REV>NONE</VOUCH_REV><ITEM_STATE>OK</ITEM_STATE><LEDGER_ID>55</LEDGER_
ID><LEDGER_ID_USER>LEDGER_0022</LEDGER_ID_USER><VOUCHER_DESC>Balance
Revaluation</VOUCHER_DESC></Item>
<Item><DOC_CURR>USD</DOC_CURR><DOC_AMT>1000.0000</DOC_AMT><BOOK_CURR>EUR</BOOK_CURR><BOOK_AMT>800
.0000</BOOK_AMT><ERP_ACC>0000113290</ERP_ACC><GLM_ACC>1111-
C</GLM_ACC><GLM_ACC_ID>768</GLM_ACC_ID><GLM_ACC_TXT>EUR C/A -
C</GLM_ACC_TXT><ENTRY_ID>138103</ENTRY_ID><VOUCH_ID>BR-2008-
0002</VOUCH_ID><V_ID>52602</V_ID><VOUCH_TYPE>BAL-REVAL</VOUCH_TYPE><TRAN_NUM>ACM-
2502</TRAN_NUM><BOOK_DATE>20090202</BOOK_DATE><VAL_DATE>20080101</VAL_DATE><ENTRY_DESC>Balance
Revaluation</ENTRY_DESC><ORIG_ID>ACM-
TRANSACTION</ORIG_ID><ORIG_CODE>ACM</ORIG_CODE><DU_ID>152</DU_ID><DI_ID>154</DI_ID><DR_CR>1</DR_C
R><DR_CR_NAME>DR</DR_CR_NAME><D_ID>152</D_ID><ITEM_TYPE>BY_VOUCHER</ITEM_TYPE><UNIT_TYPE>BY_VOUCH
ER</UNIT_TYPE><VOUCH_REV>NONE</VOUCH_REV><ITEM_STATE>OK</ITEM_STATE><LEDGER_ID>55</LEDGER_ID><LED
GER_ID_USER>LEDGER_0022</LEDGER_ID_USER><VOUCHER_DESC>Balance Revaluation</VOUCHER_DESC></Item>
</GLM>
<FUNCTION_NAME>ValidateLines</FUNCTION_NAME>
<PARTNER_PROFILE>XI_TREMA</PARTNER_PROFILE>
<OBJECT_TYPE>ZACM</OBJECT_TYPE>
<DOCUMENT_TYPE>TR</DOCUMENT_TYPE>
</ns0:MT_DocumentCheckRequest>

 WARN [Thread-48[SAPValidationTask]] 05 Feb 2009 14:01:27,592 (XIAPI.java:400) -
java.lang.Exception: No receiver could be determined(RCVR_DETERMINATION.NO_RECEIVER_CASE_BE)
      at com.trema.acm.erp.sap.xi.XIAPIviaHTTPClient.getResponse(XIAPIviaHTTPClient.java:239)
      at com.trema.acm.erp.sap.xi.XIAPI.validateDocumentLines(XIAPI.java:382)
      at sun.reflect.NativeMethodAccessorImpl.invoke0(Native Method)
      at sun.reflect.NativeMethodAccessorImpl.invoke(NativeMethodAccessorImpl.java:39)
      at sun.reflect.DelegatingMethodAccessorImpl.invoke(DelegatingMethodAccessorImpl.java:25)
      at java.lang.reflect.Method.invoke(Method.java:585)
      at com.trema.acm.erp.sap.DynamicSapAPILoader.invoke(DynamicSapAPILoader.java:54)
      at $Proxy83.validateDocumentLines(Unknown Source)
      at com.trema.acm.erp.sap.SAPValidationTask.processMessage(SAPValidationTask.java:50)
      at com.trema.acm.sf.OneThreadDaemon.performQueueProcessing(OneThreadDaemon.java:82)
      at com.trema.acm.sf.SFThreadControllerBase.run(SFThreadControllerBase.java:147)
      at java.lang.Thread.run(Thread.java:595)
```

9.6.1.2 Resolution

This error is caused by the fact that your Receiver Determination cannot find a proper service. For fixing the error you must modify the Receiver Determination for all three Interfaces:

- MI_DocumentPostRequest
- MI_DocumentExistsRequest
- MI_DocumentCheckRequest

See the example below:


Configuration: Integration Builder (pwdf6425_NX7_10)
Object Navigation Tools Environment Help
Scenarios Objects Change Lists
Receiver Determination Edit View

MANH_IS_DOM_SalesOrder
MANH_IS_Replen_PurchaseOrder
MANH_IS_SalesOrderChange
MANH_IS_SCPP_ASN
MANH_IS_SCPP_Item
MANH_IS_SCPP_PO
MANH_IS_SWFO
MANH_IS_WHSCON
MANH_IS_WHSORD_to_DO
MANH_WMMBXY
Message_from_CC_to_ISU_2
MTG_QueryWorkOrders_in_Plant
MTG_WorkOrders_IDOC_2_MIS
OPRO_Sample
PMO2000_SAP
PSImetalsSAPInterfaceConfiguration
PurchaseRequestProcessing
RFCTOHTTP
TabWare_GLAccountTxn_To_SAPECC
TabWare_PO_Import
TabWare_PO_To_SAPECC
TabWare_Receipt_To_SAPECC
TabWare_Vendor_From_SAPECC
TG_QMIDI
TL_HRPDC_XI
TW_PO_SAP
WARPIT_QA0_Buyer
WARPIT_QA0_Seller
WARPIT_QA0_TEST
WARPIT_QA9_Buyer
WARPIT_QA9_Seller
WARPIT_Seller
WARPIT_SPEC2000
WSS_SUITE
- Party
- Service Without Party
  - Business System
    - CU5_800
    - SAP_CE8
    - WSS_SUITE
  - Business Service
  - Integration Process
- Receiver Determination
  - | WSS_SUITE | MI_DocumentCheckRequest
  - | WSS_SUITE | MI_DocumentExistsRequest
  - | WSS_SUITE | MI_DocumentPostRequest
- Interface Determination
- Sender Agreement
- Receiver Agreement

XX_ccGetCustomerList
ZAK_TEST

Display Receiver Determination Status: Active

Sender

Party	
Service	WSS_SUITE
Interface	MI_DocumentPostRequest
Namespace	http://wallstreetsystems.com/acm

Receiver

Party	*
Service	*
Description	

Type of Receiver Determination

[✓ Standard] Extended

Configured Receivers

Condition	Party	Service
(/p1:MT_DocumentCheckRequest/GLM/Header/ERP_SYS = CE8_850)		SAP_CE8
(/p1:MT_DocumentCheckRequest/GLM/Header/ERP_SYS = CF5_800)		CU5_800

Condition
(/p1:MT_DocumentCheckRequest/GLM/Header/ERP_SYS = CF5_800)
p1 = http://wallstreetsystems.com/acm

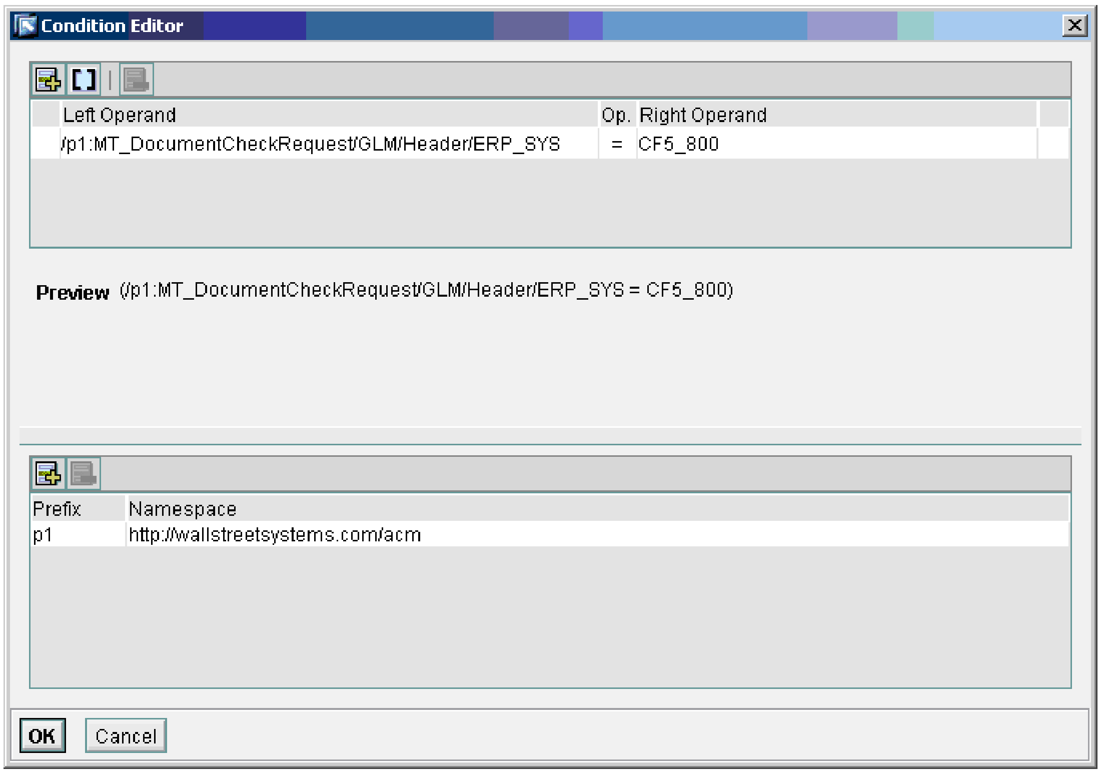
If No Receiver Is Found, Proceed as Follows:
[✓ Terminate Message Processing with Error (Restart Possible)]
End Message Processing Without Error (Restart not Possible)
Continue Message Processing with the Following Receiver: Party Service

Configuration Overview for Receiver Determination

Receiver (Partner | Service)	Interface Mapping	Receiver Agreement (Communication Channel)
| CU5_800		
| SAP_CE8		

| WSS_SUITE | MI_DocumentCheckR... | WSS_SUITE | MI_DocumentExistsR...

In the condition section you must specify a condition, for instance,
`/p1:MT_DocumentCheckRequest/GLM/Header/ERP_SYS = CF5_800`

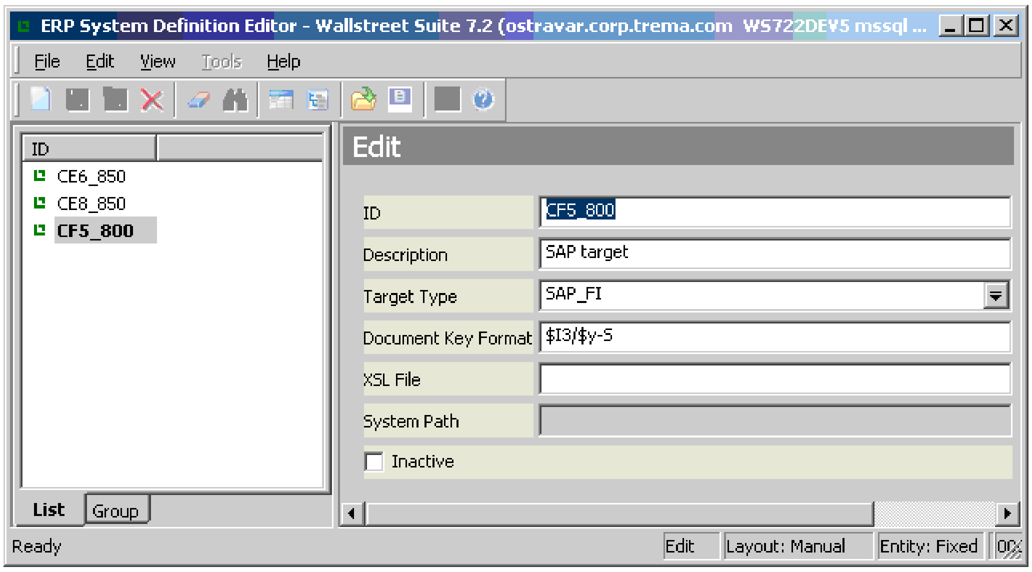


Left Operand	Op.	Right Operand
/p1:MT_DocumentCheckRequest/GLM/Header/ERP_SYS	=	CF5_800

Preview (/p1:MT_DocumentCheckRequest/GLM/Header/ERP_SYS = CF5_800)

Prefix	Namespace
p1	http://wallstreetsystems.com/acm

The ERP_SYS value is your ID for specifying in the ERP System Definition Editor:


ERP System Definition Editor - Wallstreet Suite 7.2 (ostravar.corp.trema.com WS722DEV5 mssql ...
File
Edit
View
Tools
Help
ID
CE6_850
CE8_850
CF5_800
List
Group
Edit
Ready
Edit
Layout: Manual
Entity: Fixed
0%

ID	CF5_800
Description	SAP target
Target Type	SAP_FI
Document Key Format	$I3/$y-S
XSL File	
System Path	
☐ Inactive	

Such system definition you can use then in the ERP Target Editor and assign it to any ERP target.

Warning: All the changes done in the Integration Builder (XI) must be activated before the first use!

Chapter 10 **Accounting Balances Snapshot API**

ACM contains an API for providing accounting balances. The API is called Accounting Balances Snapshot API.

The API provides **all** accounting balances in the most detail granularity of all fixed, flexible and trade level balances dimensions. The snapshot is provided for any requested period (open or closed).

The snapshot is suited for use cases when closing balance at a given point in time is essential while the incremental changes are not needed. The API does not provide the incremental changes; it provides total balances only (as opposed to the ERP interface in the Account Export Mode by balances). It can be ensured that last snapshot may be easily deleted when loading a new snapshot.

The API inputs and outputs are provided in the XML format via the JMS queue requests using an ActiveMQ message broker.

The input parameters are:

- ledger
- period

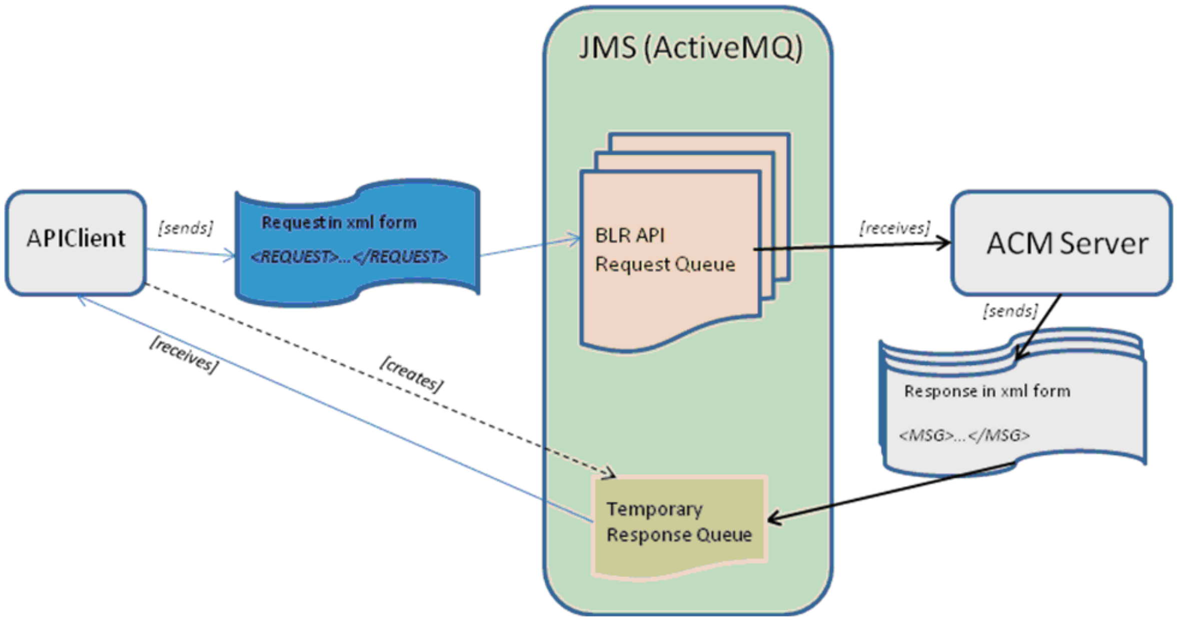
The output may by sent in multiple result messages to optimize a memory consumption and performance.

The API is language and HW neutral, so it may be accessed from Java or C++ from a distributed machine.

10.1 Technical overview

The API allows transferring of accounting balances in XML format from the ACM Server to any target system via JMS system. WSS Suite is delivered with ActiveMQ (an open source message broker ich implementing JMS), so after a clean installation of ACM the Accounting Balances Snapshot API is exposed via the delivered ActiveMQ.

Basic overview of data flow and components acting in this scenario are shown in the following picture:



For getting the Balance Snapshot data a client has to send a request (tag REQUEST) to a specific queue for Balance Snapshot Requests created on the JMS server. Also the Client has to create a temporary queue on the JMS server and to set a reference to this temporary queue via the "setJMSReplyTo()" method on the request message.

```
…

Session session = connection.createSession(transacted, ackMode);

Destination tempDest = session.createTemporaryQueue();

TextMessage txtMessage = session.createTextMessage();

txtMessage.setJMSReplyTo(tempDest);

…
```

All messages in REQUEST queue are consumed by ACM Server. Once request is consumed by ACM Server, balances for given Ledger and Period are fetched and sent as messages in form of XML to TEMPORARY Queue created by client. One request for balance snapshot may be answered by multiple messages sent to temporary queue by ACM Server. Each message (root tag for message is MSG) contains several balances (tag BALANCEBLR). For performance reasons balances are technically fetched by batch size and each batch size is represented by one message. The structure of message will be explained later in this document. The number of balances in one message is determined by variable (BalanceSnapshotExportTask.batchSize) which has default value 10000. Very last message sent by ACM Server to temporary queue is special "closing" message (tag BLR_FINISHED). This message informs the client about total count of balances and total count of messages so client can be sure that whole snapshot has been transferred.

To develop a client application it may be convenient to start from the sample application provided with the ACM package. There is acm_blr_api_client.jar which contains fully working example including a java source code. There are two java classes only for showing a small client example:

```
BLRSnapshotRequestParameters.java

BLRSnapshotAPIClient.java
```

The BLRSnapshotRequestParameters provides fields and method for a request generation.

The BLRSnapshotAPIClient is simple Accounting Balances Snapsot API client application which make request for balance snapshot and saves received messages into files. The client application is written to demonstrate basic function how to develop client applications/services. The ultimate solution developed by clients will likely parse the xml documents and update their custom data structures in their target database. The sample application concentrates only on the communication part including demonstration how to receive multi-message response from the server. For details see section BLRSnapshotRequestParameters usage.

10.2 Message structure

All communication in JMS is done via messages. This document describes several structures. There is one single structure for Request (root tag REQUEST) and two type of structures for responses:

- Regular Response Message (root tag MSG)
- Closing Response Message (root tag BLR_FINISHED)

Each of these structures are explained by individual chapter below.

10.2.1 Request Message Structure

A client has to send a Request to a queue where the ACM Server is listening for handling Balance Snapshot Requests. The request has XML form and contains input parameters for which balance snapshot is requested.

```
<REQUEST><LEDGER>LEDGER_0023</LEDGER><PERIOD>2004-08</PERIOD></REQUEST>
```

In the example above the client is requesting Balance Snapshot for Ledger "LEDGER_0023" and Period "2004-08". Tag names are fixed in Java Class called BLRSnapshotRequestParameters. This class is contained on the ACM Server as well as in "ACM BLR API Client package" to ensure compatibility.

The request can contain following input fields(tags):

TAG	Database Field	Explanation
LEDGER	ACMLedger.id_user	ACM Ledger Name
LEDGER_ID	ACMLedger.id	ACM Ledger System ID
PERIOD	ACMPeriod_user	ACM Period Name
PERIOD_ID	ACMPeriod.id	ACM Period System ID

Both types of information (period and ledger) have to be specified in the request. The client can choose whether to specify in form of name or system id.

10.2.2 Response message structure

One request is answered by one or more regular response messages and one final *closing* response message. The balance snapshot can be split to several batches if there is a huge number of balances for given request criteria. Default batch size is 10 000 of balances, however, this default can be changed (if necessary) via the ACM System Variable "*BalanceSnapshotExportTask.batchSize*".

10.2.2.1 Regular response message structure

Each regular response message has one root tag MSG and several BALANCEBLR tags; each represents one single Balance row.

```
<MSG>

   BALANCEBLR>…</BALANCEBLR>

   <BALANCEBLR>…</BALANCEBLR>

   …
```

```
    <BALANCEBLR>…</BALANCEBLR>

/MSG>
```

10.2.2.2 Stable vs. not stable snapshot

When the balance snapshots via this API are provided for an open accounting period to which new vouchers can be posted, no two API calls for the same parameters (ledger and period) return the same snapshot. This is due to the fact, that the balance snapshot can be provided from not closed accounting periods too.

You are informed about the snapshot stability via the tag MSG, which can have value *true* or *false*.

Therefore, if two API calls should return two identical snapshots, then the requested period must be closed and all its previous periods must be closed too. Such result contains a tag `MSG snapshot_stable=true` and we talk about a stable snapshot.

Another condition for the stable snapshot is the fact, that the Balance Updater finished its work. Therefore the Balance Snapshot service waits until the Balance Updater finishes its work. There is a configurable timeout specified for this waiting (property "BalanceSnapshotExportTask.periodClosingTimeout" in ms; its default value is 10 minutes). If timeout expires and balances are not fully calculated for all closed periods, an error message is sent to the client.

If requested period and all previous periods are closed and balances are calculated, the Balance Snapshot Service marks each message sent to the client as stable using attribute on MSQ tag.

```
<MSG snapshot_stable="true" run_at_date="01/11/2011 09:34:10">

    <BALANCEBLR>…</BALANCEBLR>

    <BALANCEBLR>…</BALANCEBLR>

    …

    <BALANCEBLR>…</BALANCEBLR>

</MSG>
```

If the period (or some previous period) is open, the snapshot result may change between the 2 calls. The provided snapshot is still consistent (sum = 0), however, since the result from another call might be different, the tag `MSG snapshot_stable="false"`.

As you can see in the example above, the MSG tag contains also an attribute providing information about the date and time, when the balance was obtained too. This information is available as attribute run_at_date. The format of the date is the same as in the ACM balance reports and is configurable by a server environment variable ACM_BLR_SNAPSHOT_API_DATE_FORMAT. Rules for specifying format are driven by Java class SimpleDateFormat. The default is defined as ACM_BLR_SNAPSHOT_API_DATE_FORMAT=MM/dd/yyyy HH:mm:ss. For more information about the date and time setup (including some examples), see the Java SDK documentation at http://download.oracle.com/javase/6/docs/api/java/text/SimpleDateFormat.html.

10.2.2.3 Balance Structure

Balances are transferred in the XML format. The following example shows one single balance.

```
<BALANCEBLR>

    <LEDGER>LEDGER_0023</LEDGER>

    <LEDGER_ID>57</LEDGER_ID>

    <B_CL>100000.0200</B_CL>

    <D_CR_MV>0.0000</D_CR_MV>

    <D_DR_MV>3000000.0000</D_DR_MV>
```

```
   <B_MV>100000.0200</B_MV>

   <D_OP>0.0000</D_OP>

   <D_CL>3000000.0000</D_CL>

   <B_OP>0.0000</B_OP>

   <B_CR_MV>0.0000</B_CR_MV>

   <D_MV>3000000.0000</D_MV>

   <B_DR_MV>100000.0200</B_DR_MV>

   <DOC_CURR>CZK</DOC_CURR>

   <ACM_PERIOD>2004-08</ACM_PERIOD>

   <GLM_ACC>EOY-X111-D</GLM_ACC>

   <PROJ_ID>PROJECT-D2</PROJ_ID>

   <COSTCENTER>COST-CENTER-D</COSTCENTER>

</BALANCEBLR>
```

Each balance (tag BALANCEBLR) contains only fields relevant for specific balance. Balances are provided in most detailed granularity as stored by ACM system according to Flex Balance Definitions as well as Balances By Logical Reference switch.

In the following table you can find full list of tags which may appear as a child tag of the BALANCEBLR tag. Obviously, the presence of a field is determined by setup of the balances dimensions in the system. As there is only 6 fixed and 10 flexible and 3 trade level dimensions, which can be defined, the balance tag will never contain more than 30 child tags.

Tag names are compatible with the tag names used by the ACM-ERP Export Interface.

TAG	Explanation
LEDGER	Ledger ID*
LEDGER_ID	Ledger System ID
B_CL	Booking Closing Balance
D_CL	Document Closing Balance
B_OP	Booking Opening Balance
D_OP	Document Opening Balance
D_MV	Document Movement
B_MV	Booking Movement
B_CR_MV	Booking Credit Movement
B_DR_MV	Booking Debit Movement
D_DR_MV	Document Debit Movement
D_CR_MV	Document Credit Movement
FIGURE_ID	Figure ID

TAG	Explanation
GLM_ACC **Note:** „GLM" is used to meet compatibility with ACM-ERP Interface.	Account ID*
PROJ_ID	Project ID*
COSTCENTER	Cost Center ID*
DOC_CURR	Document Currency
BOOK_CURR	Booking Currency
T_CCY_ID	Transaction Currency
T_CCY_2_ID	Trnasaction Currency 2
CPTY	Counterparty
CPTY_GRP	Counterparty group
PORTF	Portfolio
ISSR	Issuer
ISSR_GRP	Issuer Group
BANK	Bank
INSTR_GRP	Instrument Group
INSTR	Instrument
LEG	Leg
LEG_GROUP	Leg Group
BANK_ACC	Bank Account
BANK_ACC_CCY	Bank Account Currency
TRADE_ID	Counterparty Code
O_T_CPTY	One Time Counterparty
BANK_ACC_HOLDER	Bank Account Holder
P0 ... P19	Entry paramerer 0 ... Entry parameter 10
ACM_PERIOD	ACM Period ID*
ORIG_ID	Entity Origin
LOG_REF	Logical Reference ID

*User visible ID in the applications (id_user column in the database).

All decimal values are formatted using a pattern "0.0000". This pattern can be changed by the ACM system administrator via variable "BalanceSnapshotExportTask.numberFormat". For detailed explanation of patterns see the following web page on the Internet: http://java.sun.com/j2se/1.4.2/docs/api/java/text/DecimalFormat.html.

Note: There are no processing instructions (<?xml ...) inserted into any XML message.

10.2.2.4 Error Message

Once the snapshot is requested via a request message, a client waits for a response on a temporary queue, which the client created. Received message can have two forms:

1. Form as described in the previous chapter.
2. Form "Error" message, which indicates some troubles with completing the request (for instance, the user has no permissions for fetching the balances, balances are not calculated in the given timeout, etc.). Such message has the following structure:

```
<ERROR>The request has not been sent by autenticated user. Request is
ignored.</ERROR>
```

10.2.2.5 Closing message structure

Once the ACM Server finishes sending of messages with balances, it sends to same temporary queue last closing message. The purpose of this message is to inform the client that the balances snapshot transfer ends. It specifies how many balances and how many messages were produced as an answer to the request. The client should use these numbers for checking that the transfer was successful.

```
<BLR_FINISHED><BAL_COUNT>34</BAL_COUNT><MSG_COUNT>3</MSG_COUNT></BLR_FINISHED>
```

TAG	Explanation
BAL_COUNT	Total count of balances transferred as result of request
MSG_COUNT	Total count of balance messages sent by ACM Server to temporary queue. The last "closing" message is not counted.

10.3 API usage example

There is a biz.wss.acm.blr.api.jar file which contains working example of the client application which sends requests to the server and receives answers. Default usage is shown below:

```
Usage:java BLRSnapshotAPIClient ledger=MyLedger period=2008-01
broker=http:\localhost:61616 queue=ACM_BAL_SNAPSHOT_API_REQUEST user=MyUser
pass=MyPass file=MyFile

Parameters:

ledger Specify Ledger Name

period Specify Period Name

broker Specify Broker URL ($FK_MQ_BROKER_URL_JAVA)

queue Specify Queue Name ($ACM_BAL_SNAPSHOT_API_QUEUE_NAME) Default is defined
as:${wss_env_name}.acm.blr.api.request.queue

user Specify username with permissions to log into WSS Suite and having
ACM-BALANCE-QUERY-LEDGER-REPORT permission for given ledger.

pass Password

file Output filename prefix including path.
```

As you can see you have to specify ledger, period, broker, queue, user, pass and file prefix.

If you have installed Wallstreet Suite, you can simply open *shell* and proceed the following procedure:

```
   cd %FK_HOME%\etc\acm\blr-api
   java -jar biz.wss.acm.blr.api ledger=MyLedger period=2008-01
   broker=http:\localhost:61616 queue=ACM_BAL_SNAPSHOT_API_REQUEST user=MyUser
   pass=MyPass file=MyFile

Taking parameter:ledger=LE_DCAGF_LOC_DE

Taking parameter:period=2008-08

Taking parameter:broker=failover:tcp://velvet.eu.wallstreetsystems.com:20460

Taking parameter:queue=MC72.acm.blr.api.request.queue

Taking parameter:user=dbo

Taking parameter:pass=fk71

Taking parameter:file=c:\2008_08.xm

Starting BLRSnapshotAPIClient

Oct 14, 2008 10:48:35 AM

org.apache.activemq.transport.failover.FailoverTransport doReconnect

INFO: Successfully connected to tcp://velvet.eu.wallstreetsystems.com:20460

<REQUEST><LEDGER>LE_DCAGF_LOC_DE</LEDGER><PERIOD>2008-08</PERIOD></REQUEST>

OK! Messages Received:9

Exiting. Run time:170s.
```

In the previous example the client application produced one file MyFile.xm0. If there are more messages returned, also other files will be generated (MyFile.xm1, MyFile.xm1, …).

10.4 Other configuration parameters

The ACM Server needs to know some basic configuration variables to properly connects to and serve requests. Following table summarizes this:

System Property	Value	Explanation
ACM_BAL_SNAPSHOT_API_QUEUE_NAME	MC72.acm.blr.api.request.queue	Name of queue where ACM Server listen for request
jms_broker_url	failover:tcp://hostname:port	Broker URL
ActiveMQExporter.time_to_live	900000	Time to Live for response messages in miliseconds
ActiveMQExporter.delivery_mode	2	Delivery Mode according to JMS Specification 1=non-persistent, 2=persistent.
BalanceSnapshotExportTask.batchSize	5000	Response Batch Size None response message contains more then specified number of balances.
BalanceSnapshotExportTask.numberFormat	0	Number format for amount values. See http://java.sun.com/j2se/1.4.2/docs/api/java/text/DecimalFormat.html for examples how to define number format.

System Property	Value	Explanation
BalanceSnapshotExportTask.periodClosingTimeout	600000	Timeout for wainting for finalizing balance calculation on closed periods. Value is in miliseconds.
jms_broker_url	failover:tcp://hostname:port	Broker URL

Chapter 11 Factory accounting data upload

Factory data is the a general name for default static data which may be distributed with Wallstreet Suite.

11.1 Loading factory accounting data

Factory accounting data can be loaded into the ACM database tables in two ways:

- Via the DBLoader tool distributed with Wallstreet Suite.
- Via the comKIT programming interface distributed with Wallstreet Suite (for details see *comKIT Programming Guide*).

When using these tools to load the static data (charts, accounts, ledgers etc.) all the entities are verified prior their upload into the database tables. One of the checks is about the verification whether the entity is loaded within an accounting perspective. For details about the accounting perspective (i.e. what it is, what entities are handled via the perspectives, how accounting perspectives are set up) see the *ACM User Guide*, chapter *Accounting Perspective Editor*.

In order to make the usage of these tools more convenient and straight forward, a special option can be set up to disable the verification of the accounting perspective.

To check the value of the option, connect to the database using the appropriate tool:

- Sybase

  ```
  isql -S <server> -D <database> -U <user> -P <password>
  ```

- Microsoft SQL Server

  ```
  isql -S <server> -d <database> -U <user> -P <password>
  ```

- Oracle

  ```
  sqlplus <user>/<password>@<database>
  ```

Then issue the following query (for all database platforms):

```
select value from Configuration where id = 10000
```

Result value can be one of the following two:

- **true**

  Perspective checking is disabled

- **false**

  Perspective checking is enabled

For chaning the value run the following command:

```
update Configuration set value = ’true’ where id = 10000
```

or

```
update Configuration set value = ’false’ where id = 10000
```

The first update disables the perspective check for the both loading tools as well as for the client editors, the second update enables the check.

Warning: The default value is *false*, i.e.the accounting perspective checking is performed. You can change it to *true* only for the data upload and must be always set back to *false* after the data upload. **Leaving the option set to true can result in inconsistent static data setup!**

Chapter 12 Tips for migration to ACM

This chapter provides a few tips, how to migrate from the non-Wallstreet Suite accounting system to ACM.

12.1 Using the Migration Date

After migrating/upgrading the TRM transactions you may want to block the accounting events generation before a certain date. That date is called Migration Date and is configurable in the **Configuration** table; by default its id is 950 and it is specified in format yyyyMMdd.

That blocking logic is implemented in the following 3 activities:

- TRM Daily Events Generation
- TRM CTB Events Generation
- TRM Daily Delta Events Generation

These activities don't create events before/equal to the migration date, i.e. these activities can be executed only for dates > than the migration date. If any of them is started with Due Date before the migration date, it fails with a failure message "*Due date <= migration date. Not allowed to start process before migration date.*"

Besides the accounting events generation, it also blocks the selling and FX conversion logic before the migration date (since this logic is the part of the TRM Daily Events Generation activity).

12.2 Upload of manual vouchers for open TRM transactions

It is assumed that all open (live) transactions at the moment of the go-live date are captured in TRM. For these transactions it is necessary to post into ACM all daily (RUNNING, OFF-BALANCE, DAILY DELTA) bookings till the go-live date. If you decide to do it manually via the Accounting Entry Manger Export Tool, fill the origins on the accounting entry level as follows:

- **Origin Group** = TRM,
- **Origin** = TRM-GENERIC,
- **Origin Entity** = TRM-TRANSACTION

Their **Reference ID** must be equal to the appropriate transaction number in TRM.

Voucher Source will by automatically set to ENTRY-MANAGER (you cannot influence on that) so it will be always obvious it was entered manually.

You can use a special voucher type for them too, so you can really easily identify them in the system.

When uploading the opening balances for active TRM transactions by Import function in Accounting Manager in the granularity of trade level balances, besides the other fields you should fill the following ones in this way:

- Reference ID = number of the TRM transaction for which you are entering the opening balances
- Logical Reference ID = Continuation Number of the transaction for which you are entering the opening balances (note1: it should be used only for non-sellable instruments)
- Leg Group = Leg Group of the appropriate leg from Transaction/Cashflows
- Leg = Leg (Instrument) of the appropriate leg from Transaction/Cashflows
- Continuation Number = continuation number of the transaction for which you are entering the opening balances (note: all TRM transactions have Continuation Number even if it equals to Transaction Number – these should be filled to Accounting Entries as well)
- Origin = TRM-GENERIC
- Origin = TRM (or might be empty since the TRM value is derived automatically based on the origin TRM-GENERIC)
- Origin Entity = TRM-TRANSACTION

Note that you can easily recognize a manual voucher by its source (Voucher Source = ENTRY-MANAGER)

12.3 Manually entered PRE-BOOKED vouchers

You may need to enter some vouchers, which must be posted to the FINAL state just in future (e.g. the OCI account discharges). This requirement is supported by ACM via the voucher state PRE-BOOKED, to which the system posts all vouchers with Required Booking Date in future.

Warning: For booking of vouchers to state PRE-BOOKED the corresponding future periods must be open; otherwise the future vouchers are posted to state VERIFY.

Note: Required Booking Date is filled by a maximum event date from the entries grouped to a voucher.

Vouchers from the PRE-BOOKED state can be posted to FINAL (by running the Accounting Processing or Shift Voucher State activity, or manually from the Entry Manager) earliest on a day, when the voucher Required Booking Date is equal to a current date.

You have two ways for entering the PRE-BOOKED vouchers into the system:

1. **Manually to enter voucher by voucher.**

   You can enter the appropriate voucher for each future (e.g. discharge) date; when entering such voucher you must set the event date of the entries to that future date.

2. **Using the rebooking feature.**

   If you want to rebook some initial balance from account A to account B periodically within some period, you may do it via the rebooking feature (for details see the *ACM User Guide*, chapter *Rebooking Accounting Entries*). You can assign the rebooking feature for the initial balance (entry) and to leave the system generate the future vouchers. For the initial entry you must fill the following fields:

   - Special Treatment = REBOOKING
   - Special Type ... the type of rebooking defined in the Rebooking Type editor
   - Secondary Account ... the account to which the initial balance should be re-booked
   - Transaction Opening/Value/Maturity Date ... for defining the From/To period of rebooking

The PRE-BOOKED vouchers cannot be rejected nor canceled. They can only be reversed. If you want to post some PRE-BOOKED voucher immediately, it is best to apply on it Reverse+Copy action from the Entry Manager. For the initial PRE-BOOKED voucher its exact reverse will be created and on the new (OPEN) voucher you can adjust the event dates to the current date. Such voucher will be posted to FINAL.

Note that initial PRE-BOOKED voucher as well as its reversing voucher (also posted in state PRE-BOOKED) remain in the system and will be posted to FINAL on their Required Booking Date. But since they offset each other they have no impact on account balances.

12.4 Starting-up time to maturity feature

You must post the entries, which should be the subject of the time to maturity feature, to the leading account and not to the particular time band accounts; posting to the particular time band accounts breaks the feature! To the particular time band accounts the exact balances are posted by running the Time to Maturity Processing activity

Warning: Don’t forget to fill the Transaction Maturity Date for the manually entered entries!

Appendix A Event DIMs structure

Accounting of TRM transactions and CLM payment advices and allocations is based on accounting events. They are stored in two tables:

- **AccountingEvent**

  This table is used for storing the events related the TRM transactions.

- **EntityEvent**

  This table is used for storing the events related the CLM payment advices and payment allocations.

Key information like event dates, amounts and FX rates contained in events are filled automatically from the events generation process itself. Additional information like instrument, counterparty, bank account, etc., are enriched by stored procedures. The information are populated into so called DIM fields. The following two subchapters describe a default content of those DIM fields.

A.1 Accounting Event

Since the structure of the accounting event (table AccountingEvent) is generic and might be changed by CSDs, some columns are named in a generic way. They are of 5 types for 5 different data types:

- **date** ... ddim_0 - ddim_9
- **float** ... fdim_0 - fdim_9
- **integer** ... idim_0 - idim_14
- **money** ... mdim_0 - mdim_9
- **string** ... sdim_0 - sdim_39

Values into these fields are retrieved via 4 SQL procedures:

- **GetGeneralEventDims**, which is called for all three input types (RUNNING, CTB, OBS).
- **GetRunningEventDims**, which is called for RUNNING inputs/events only.
- **GetClosingEventDims**, which is called for CLOSING (CTB) inputs/events only.
- **GetOffBalanceEventDims**, which is called for Off Balance (OBS) inputs/events only.

The table below shows what SQL procedure fills what field.

Table column name	Description	Get General	Get Running	Get Closing	Get OffBalance
ddim_0	Transaction opening date	transactions. opening_date			
ddim_1	Transaction value date	transactions. value_date			

Table column name	Description	Get General	Get Running	Get Closing	Get OffBalance
ddim_2	Transaction maturity date	transactions.maturity_date			
ddim_3	Cashflow payment date		@ref_date		
ddim_4 .. 9	NOT USED				
fdim_0	Deal rate	transactions.deal_rate	transactions.deal_rate		
fdim_1	Deal price	transactions.deal_price	transactions.deal_price		
fdim_2 .. 9	NOT USED				
idim_0	Transaction Number	transactions.number			
idim_1	Package ID		transactions.package_id	transactions.package_id	transactions.package_id
idim_2	Transaction Type ID	transactions.type_id			
idim_3	Cashflow Type / Key Figure ID		CashFlowType.key_id	@figure_id	**Note:** For OBS events this field is populated by @event_subtype_id (directly in the MakeOffBalanceEvent procedure)
idim_4	Transaction Sign	transactions.sign_id			
idim_5	Continuation number	transactions.logical_number			
idim_6	NOT USED				
idim_7	Instrument group path ID	UMI.path_id			
idim_8	PL Category (cashflow kind)	The value might be assigned by the events generation process (for some special cases, e.g. time to maturity).	cashflow@kinds		
idim_9	Transaction Kind ID	transactions.kind_id			
idim_10	NOT USED				

Table column name	Description	Get General	Get Running	Get Closing	Get OffBalance
idim_11	Accounting Treatment		@acc_treatment		@acc_treatment
idim_12 .. 14	NOT USED				
mdim_0	Nominal Amount (local)	transactions.amount			
mdim_1	Book Value (Local)	transactions.book_value			
mdim_2 .. 9	NOT USED				
sdim_0	Market	transactions.market_id			
sdim_1	Transaction Instrument ID	transactions.instrument_id			
sdim_2	Base Currency of Transaction	transactions.currency_id			
sdim_3	Quote Currency of Transaction	transactions.currency_2_id			
sdim_4	Portfolio ID	transactions.portfolio_id			
sdim_5	Counterparty ID	isnull(sdim_5,transactions.cp_client_id)	isnull(sdim_5,transactions.cp_client_id)	@payment_client_id	
sdim_6	Counterparty Group ID	ClientGroupMap.group_id			
sdim_7	Issuer ID	transactions.issuer_id			
sdim_8	Package Type ID		PackageMap.type_id	PackageMap.type_id	PackageMap.type_id
sdim_9	Instrument Type ID	UMI.type_id			
sdim_10	Transaction Parameter #0, 1, 2, 3, 4	transactions.param_0, 1, 2, ,3, 4			
sdim_11					
sdim_12					
sdim_13					
sdim_14					
sdim_15	Our Bank ID		ClientAccount.bank_id	@local_bank_id, but is is mostly empty for the CTB events/entries.	

Table column name	Description	Get General	Get Running	Get Closing	Get OffBalance
sdim_16	Our Account ID		ClientAccount.id	@local_account_id, but it is mostly empty for the CTB events/entries.	
sdim_17	Issuer Group ID	ClientGroupMap.group_id			
sdim_18	Classification	ClassificationMap.classification_id			
sdim_19	Facility ID	transactions.sattr_1			
sdim_20	Bank Account Holder as the owner of the accounting event		@owner_id	@owner_id	
sdim_21	Bank Account Currency		@payment_currency_id	ClientAccount.currency_id; If more accounts for given client, bank and bank account exist, then it is null	
sdim_22	Third Party Owner of the portfolio; if not specified then Owner	if sdim_4 is not null then value from PropertyMap where key_id = @sdim_4 and type_id = "THIRD-PARTY-OWNER" and object_id = "Portfolio" if sdim_22 is null then owner_id			
sdim_23	Leg, i.e. the instrument of a leg (Cashflow.leg_id)		@leg_id	@leg_id	@leg_id
sdim_24	Bank Account Holder (sdim_20) Client Group ID	ClientGroupMap.group_id			

Table column name	Description	Get General	Get Running	Get Closing	Get OffBalance
sdim_25	Transaction Parameter #5, 6, 7, 8, 9	transactions. param_5, 6, 7, 8, 9			
sdim_26					
sdim_27					
sdim_28					
sdim_29					
sdim_30	Branch Code #0, 1, 2, 3, 4 of a given transaction instrument. Branch codes are read from the UMIBranch table considering a time validity for given event date and date range defined via the Active From/Active To dates in the Instrument editor.	UMIBranch.branch_id where branch_number = 0, 1, 2, 3, 4 and umi_id= @sdim_1			
sdim_31					
sdim_32					
sdim_33					
sdim_34					
sdim_35	Branch Code #0, 1, 2, 3, 4 of a given leg instrument. Branch codes are read from the UMIBranch table considering a time validity for given event date and date range defined via the Active From/Active To dates in the Instrument editor.	UMIBranch.branch_id where branch_number = 0, 1, 2, 3, 4 and umi_id = leg_id			
sdim_36					
sdim_37					
sdim_38					
sdim_39					

A.2 Entity Event

Since the structure of the entiy event (table EntityEvent) is generic and might be changed by CSDs, some columns are named in a generic way. They are of 5 types for 5 different data types:

- **date** ... ddim_0 - ddim_9
- **float** ... fdim_0 - fdim_9
- **integer** ... idim_0 - idim_14
- **money** ... mdim_0 - mdim_9
- **string** ... sdim_0 - sdim_39

Values into these fields are retrieved via 2SQL procedures with the excpetion of a few columns to which the values are sent directly when creatin the events (see the column *Filled out of the Get... procedures* below in the table). The 2basic SQL procedures are:

- **GetPaymentEventDims**, which is used for payment advices and payment side of allocations
- **GetAllocCashflowEventDims**, which is used for cashflow side of allocations

The table below shows what SQL procedure fills what field.

Table column name	Description	Filled out of the Get ... procedures	Get Payment	Get AllocCashflow
ddim_0	Transaction opening date			transactions.opening_date
ddim_1	Transaction value date			transactions.value_date

Table column name	Description	Filled out of the Get ... procedures	Get Payment	Get AllocCashflow
ddim_2	Transaction maturity date			transactions.maturity_date
ddim_3	Cashflow payment date	YES		
ddim_4 .. 9	NOT USED			
fdim_0	Deal rate			transactions.deal_rate
fdim_1	Deal price			transactions.deal_price
fdim_2 .. 9	NOT USED			
idim_0	Transaction Number			transactions.number
idim_1	Package ID			transactions.package_id
idim_2	Transaction Type ID			transactions.type_id
idim_3	Cashflow Type / Key Figure ID	YES		
idim_4	Transaction Sign			transactions.sign_id
idim_5	Continuation number		@payment_id	transactions.logical_number
idim_6	NOT USED			
idim_7	Instrument group path ID		Settlement.instrument_path ID	UMI.path_id
idim_8	NOT USED			
idim_9	Transaction Kind ID			transactions.kind_id
idim_10	NOT USED			
idim_11	Accounting Treatment	YES		
idim_12 .. 14	NOT USED			
mdim_0	Nominal Amount (local)			transactions.amount
mdim_1	Book Value (Local)			transactions.book_value
mdim_2 .. 9	NOT USED			
sdim_0	Market			transactions.market_id
sdim_1	Transaction Instrument ID		settlement.instrument_id	transactions.instrument_id
sdim_2	Base Currency of Transaction			transactions.currency_id
sdim_3	Quote Currency of Transaction			transactions.currency_2_id

Table column name	Description	Filled out of the Get ... procedures	Get Payment	Get AllocCashflow
sdim_4	Portfolio ID		settlement.portfolio_id	transactions.portfolio_id
sdim_5	Counterparty ID		settlement.other_client_id	isnull(sdim_5,transactions.cp_client_id)
sdim_6	Cparty Group ID		ClientGroupMap.group_id	ClientGroupMap.group_id
sdim_7	Issuer ID			transactions.issuer_id
sdim_8	Package Type ID			if idim_1 not null then sdim_8 = PackageMap.type_id
sdim_9	Instrument Type ID			UMI.type_id
sdim_10	Transaction Parameter #0, 1, 2, 3, 4			transactions.param_0, 1, 2, ,3, 4
sdim_11				
sdim_12				
sdim_13				
sdim_14				
sdim_15	Our Bank ID		settlement.local_bank_1_id	if @payment_id not null then sdim_15 = ClientAccount.bank_id
sdim_16	Our Account ID		settlement.local_account_1_id	if @payment_id not null then sdim_15 = ClientAccount.id
sdim_17	Issuer Group ID			ClientGroupMap.group_id
sdim_18	NOT USED			
sdim_19	Facility ID		settlement.param_0	transactions.sattr_1
sdim_20	Bank Account Holder as the owner of the accounting event		@owner_id	@owner_id
sdim_21	Bank Account Currency		settlement.currency_id	@payment_currency_id
sdim_22 .. 23	NOT USED			
sdim_24	Bank Account Holder (sdim_20) Client Group ID	ClientGroupMap.group_id		

Table column name	Description	Filled out of the Get ... procedures	Get Payment	Get AllocCashflow
sdim_25	Transaction Parameter #5, 6, 7, 8, 9			transactions.param_5, 6, 7, 8, 9
sdim_26				
sdim_27				
sdim_28				
sdim_29				
sdim_30	Branch Code #0, 1, 2, 3, 4 of a given transaction instrument. Branch codes are read from the UMIBranch table considering a time validity for given event date and date range defined via the Active From/Active To dates in the Instrument editor.			UMIBranch.branch_id where branch_number = 0, 1, 2, 3, 4 and umi_id= @sdim_1
sdim_31				
sdim_32				
sdim_33				
sdim_34				
sdim_35	Branch Code #0, 1, 2, 3, 4 of a given leg instrument. Branch codes are read from the UMIBranch table considering a time validity for given event date and date range defined via the Active From/Active To dates in the Instrument editor.			UMIBranch.branch_id where branch_number = 0, 1, 2, 3, 4 and umi_id = leg_id
sdim_36				
sdim_37				
sdim_38				
sdim_39				

Appendix B Object permissions

This appendix contains details for the Permission Editor, used to grant and revoke access to the ACM objects.

Warning: For a complete set of permissions for a particular user/user group do not forget to view the Permission Editor with activated view option **Inherited Permissions**.

B.1 ACM permissions

The following table contains the list of permissions that can be set on the ACM objects:

Permission	Description
ACTIVITY	*Not applicable for the ACM objects.*
ALL	All permissions
CREATE	Permission to create entities
MODIFY	Permission to modify entities
READ	Permission to read entities
REMOVE	Permission to remove entities.
SELF-VERIFY	Self-verify object modifications.
START	*Not applicable for the ACM objects.*
VERIFY	*Not applicable for the ACM objects.*

B.2 ACM objects with permissions per ACM user groups

The following table describes all the accounting related objects for which you can set the user permissions. The objects are of the following 3 basic kinds:

1. Data: tables representing key static data which can be manipulated via the editors; for such objects the table contains the name of the appropriate editor too.
2. Lists: tables representing the selection lists (for editors, reports, activities) or key values (e.g. voucher states, bookkeeping types, etc.)
3. System: tables used by the system for the processing.

The table also includes the configuration of the particular objects in relation to the default (installed automatically with ACM) accounting user groups. Since the permissions of the ACM-ADMIN user group are the sub-set of the ACM-MAIN-ACCOUNTANT user group, the permissions for the

ACM-ADMIN group are listed just as the exceptions from the ACM-MAIN-ACCOUNTANT group in the next chapter.

Object	Kind	Primary Editor	Description	ACM-CONTROLLER	ACM-ACCOUNTANT	ACM-MAIN-ACCOUNTANT
ACMAccount	Data	Account Editor	Determines the user permission for manipulating with data in Account Editor; since the accounts are used in some other ACM applications and activities, each user should have at least READ permission. Permissions for the Attribute Overriding page are defined separately.	READ	ALL	ALL
ACMAccountCategory	List		System table containing values for a selection list used in ACM applications or the table of key values; you should only read data from this table.	READ	ALL	ALL
ACMAccountTreeFlat	System			READ	ALL	ALL
ACMAccountType	Data	Account Type Editor	Determines the user permission for manipulating with data in Account Type Editor; since this data are used in some other ACM applications and activities, each user should have at least READ permission.	READ	ALL	ALL
ACMActivityLog	System			READ	ALL	ALL
ACMActivityParameter	System			READ	ALL	ALL
ACMActivityRequest	System			READ	ALL	ALL
ACMAuditLog	System			ALL	ALL	ALL
ACMBalance	System		System table to which the account balances are stored.	ALL	ALL	ALL
ACMBalanceCheckpoint	System			READ	ALL	ALL
ACMBalanceType	List		System table containing values for a selection list used in ACM applications or the table of key values; you should only read data from this table.	READ	READ	ALL
ACMBookkeepingType	List		System table containing values for a selection list used in ACM applications or the table of key values; you should only read data from this table.	READ	ALL	ALL

Object	Kind	Primary Editor	Description	ACM-CONTROLLER	ACM-ACCOUNTANT	ACM-MAIN-ACCOUNTANT
ACMCMMState	List		System table containing values for a selection list used in ACM applications or the table of key values; you should only read data from this table.	READ	ALL	ALL
ACMCMMViewType	List		System table containing values for a selection list used in ACM applications or the table of key values; you should only read data from this table.	READ	ALL	ALL
ACMCTBHistory	System			READ	ALL	ALL
ACMCTBHistoryType	System		System object; a permission for this object is not checked by default.			
ACMCTBSequence	System		System object; a permission for this object is not checked by default.			
ACMCTBSequenceBalance	System		System object; a permission for this object is not checked by default.			
ACMCTBSequenceState	System		System object; a permission for this object is not checked by default.			
ACMCTBTypeRule	Data	Closing Book Type Editor	Determines the user permission for manipulating with data in Closing Book Type Editor; since this data are used in some other ACM applications and activities, each user should have at least READ permission.	READ	ALL	ALL
ACMChart	Data	Chart of Accounts Editor	Determines the user permission for manipulating with data in Chart of Accounts Editor; since this data are used in some other ACM applications and activities, each user should have at least READ permission. Permissions for the Domains page are defined separately.	READ	ALL	ALL
ACMChartM	Data	Chart of Accounts Editor	Determines the user permission for manipulating with additional domains setup in Chart of Account Editor, page Domains. Each user should have at least READ permission.	READ	READ	ALL

Object	Kind	Primary Editor	Description	ACM-CONTROLLER	ACM-ACCOUNTANT	ACM-MAIN-ACCOUNTANT
ACMChartPersAccType	Data	Account Type Editor	Determines the user permission for making the attribute overriding setup in Account Type Editor, page Attribure Overriding, in the chart perspective. Each user should have at least READ permission.	READ	ALL	ALL
ACMChartPersAccount	Data	Account Editor	Determines the user permission for making the attribute overriding setup in Account Editor, page Attribure Overriding, in the chart perspective. Each user should have at least READ permission.	READ	ALL	ALL
ACMChartPersCC	Data	Cost Center Editor	Determines the user permission for making the attribute overriding setup in Cost Center Editor, page Attribure Overriding, in the chart perspective. Each user should have at least READ permission.	READ	ALL	ALL
ACMChartPersProject	Data	Project Editor	Determines the user permission for making the attribute overriding setup in Project Editor, page Attribure Overriding, in the chart perspective. Each user should have at least READ permission.	READ	ALL	ALL
ACMChartTreeFlat	System			READ	ALL	ALL
ACMCostCenter	Data	Cost Center Editor	Determines the user permission for manipulating with data in Cost Center Editor; since this data are used in some other ACM applications and activities, each user should have at least READ permission.	READ	ALL	ALL
ACMCptyRelation	List		System table containing values for a selection list used in ACM applications or the table of key values; you should only read data from this table.	READ	ALL	ALL

Object	Kind	Primary Editor	Description	ACM-CONTROLLER	ACM-ACCOUNTANT	ACM-MAIN-ACCOUNTANT
ACMCritValueMapping	Data	Breakdown Criterion Editor	Determines the user permission for manipulating with data in Breakdown Criterion Editor, page Criterion Components; since this data are used in some other ACM applications and activities, each user should have at least READ permission.	READ	ALL	ALL
ACMCriterion	Data	Breakdown Criterion Editor	Determines the user permission for manipulating with data in Breakdown Criterion Editor; since this data are used in some other ACM applications and activities, each user should have at least READ permission. Permissions for the Domains page are defined separately.	READ	ALL	ALL
ACMCriterionM	Data	Breakdown Criterion Editor	Determines the user permission for manipulating with additional domains setup in Breakdown Criterion Editor, page Domains. Each user should have at least READ permission.	READ	READ	ALL
ACMCurrencyBalance	System			READ	ALL	ALL
ACMCurrencyBalanceKind	List		System table containing values for a selection list used in ACM applications or the table of key values; you should only read data from this table.	READ	ALL	ALL
ACMCurrencyBalanceType	List		System table containing values for a selection list used in ACM applications or the table of key values; you should only read data from this table.	READ	ALL	ALL
ACMCurrencyEntryState	List		System table containing values for a selection list used in ACM applications or the table of key values; you should only read data from this table.	READ	ALL	ALL
ACMCurrencyPosition	Data		Determines the user permission for manipulating with data in the Currency Position Editor; since this data are used in some other ACM applications and activities, each user should have at least READ permission.	READ	ALL	ALL

Object	Kind	Primary Editor	Description	ACM-CONTROLLER	ACM-ACCOUNTANT	ACM-MAIN - ACCOUNTANT
ACMCurrencyRule	Data		Determines the user permission for manipulating with data in Currency Position Editor, page Rules; since this data are used in some other ACM applications and activities, each user should have at least READ permission.	READ	ALL	ALL
ACMDataVersion	System			ALL	ALL	ALL
ACMDynamicDataVerification	System		System table needed for the execution of the Accounting Dynamic Data Verification report. Note that by default only users from the ACM-ADMIN group have permission ALL for this object.			
ACMDrCr	List		System table containing values for a selection list used in ACM applications or the table of key values; you should only read data from this table.	READ	ALL	ALL
ACMERPDoc	System			READ	READ	ALL
ACMERPDocItem	System			READ	READ	ALL
ACMERPDocItemState	System		System object; a permission for this object is not checked by default.	READ	READ	ALL
ACMERPDocState	System		System object; a permission for this object is not checked by default.	READ	READ	ALL
ACMERPDocUnit	System			READ	READ	ALL
ACMERPDocUnitState	System		System object; a permission for this object is not checked by default.	READ	READ	ALL
ACMERPExportDef	Data	ERP Export Definition Editor	Determines the user permission for manipulating with data in ERP Export Definition Editor; since this data are used in some other ACM applications and activities, each user should have at least READ permission.	READ	READ	ALL
ACMERPExportMode	List		System table containing values for a selection list used in ACM applications or the table of key values; you should only read data from this table.	READ	READ	ALL

Object	Kind	Primary Editor	Description	ACM-CONTROLLER	ACM-ACCOUNTANT	ACM-MAIN-ACCOUNTANT
ACMERPJoinHelp	System		System table used by the ERP interface executed in the account mode. A permission for this object is not checked by default.			
ACMERPMessage	System		System object; a permission for this object is not checked by default.	READ	READ	ALL
ACMERPState	System		System object; a permission for this object is not checked by default.	READ	ALL	ALL
ACMERPSystemDef	Data	ERP System Definition Editor	Determines the user permission for manipulating with data in ERP System Definition Editor; since this data are used in some other ACM applications and activities, each user should have at least READ permission.	READ	READ	ALL
ACMERPTarget	Data	ERP Target Editor	Determines the user permission for manipulating with data in ERP Target Editor; since this data are used in some other ACM applications and activities, each user should have at least READ permission.	READ	READ	ALL
ACMERPTargetNumber	List		System table containing values for a selection list used in ACM applications or the table of key values; you should only read data from this table.	READ	READ	ALL
ACMERPTargetType	List		System table containing values for a selection list used in ACM applications or the table of key values; you should only read data from this table.	READ	READ	ALL
ACMERPType	List		System table containing values for a selection list used in ACM applications or the table of key values; you should only read data from this table.	READ	READ	ALL
ACMEntry	System		Key system table containing all informatoin about the accounting entries.	READ	ALL	ALL

Object	Kind	Primary Editor	Description	ACM-CONTROLLER	ACM-ACCOUNTANT	ACM-MAIN-ACCOUNTANT
ACMEntryOrigin	List		System table containing values for a selection list used in ACM applications or the table of key values; you should only read data from this table.	READ	ALL	ALL
ACMEntryOriginEntity	List		System table containing values for a selection list used in ACM applications or the table of key values; you should only read data from this table.	READ	ALL	ALL
ACMEntryOriginGroup	List		System table containing values for a selection list used in ACM applications or the table of key values; you should only read data from this table.	READ	ALL	ALL
ACMEntryState	System		System object; a permission for this object is not checked by default.	READ	ALL	ALL
ACMEntryType	List		System table containing values for a selection list used in ACM applications or the table of key values; you should only read data from this table.	READ	ALL	ALL
ACMEventEntryMap	Data	Event to Entry Mapping Editor	Determines the user permission for manipulating with data in Event to Entry Mapping Editor; since this data are used in some other ACM applications and activities, each user should have at least READ permission. Note that by default only users from the ACM-ADMIN group have permission ALL for this object.	READ	READ	READ
ACMEventType	List		System table containing values for a selection list used in ACM applications or the table of key values; you should only read data from this table.	READ	ALL	ALL
ACMEventTypeMap	System		System object; a permission for this object is not checked by default.			

Object	Kind	Primary Editor	Description	ACM-CONTROLLER	ACM-ACCOUNTANT	ACM-MAIN-ACCOUNTANT
ACMGroupingRule	Data	Accounting Grouping Rule Editor	Determines the user permission for manipulating with data in Accounting Grouping Rule Editor; since this data are used in some other ACM applications and activities, each user should have at least READ permission.	READ	ALL	ALL
ACMGroupingRuleType	List		System table containing values for a selection list used in ACM applications or the table of key values; you should only read data from this table.	READ	ALL	ALL
ACMHedgeEffCat	List		System table containing values for a selection list used in ACM applications or the table of key values; you should only read data from this table.	READ	READ	ALL
ACMHedgeLegCat	List		System table containing values for a selection list used in ACM applications or the table of key values; you should only read data from this table.	READ	READ	ALL
ACMHedgeQualifCat	List		System table containing values for a selection list used in ACM applications or the table of key values; you should only read data from this table.	READ	READ	ALL
ACMHedgeRealized	System		System object; a permission for this object is not checked by default.	READ	READ	ALL
ACMHedgeRealizedState	List		System table containing values for a selection list used in ACM applications or the table of key values; you should only read data from this table.	READ	READ	ALL
ACMId	System		System object; a permission for this object is not checked by default.	ALL	ALL	ALL
ACMInstructionEvent	System		System object; a permission for this object is not checked by default.			
ACMInstructionState	List		System table containing values for a selection list used in ACM applications or the table of key values; a permission for this object is not checked by default.			

Object	Kind	Primary Editor	Description	ACM-CONTROLLER	ACM-ACCOUNTANT	ACM-MAIN-ACCOUNTANT
ACMInstructionType	List		System table containing values for a selection list used in ACM applications or the table of key values; a permission for this object is not checked by default.			
ACMLedger	Data	Ledger Editor	Determines the user permission for manipulating with data in Ledger Editor; since this data are used in some other ACM applications and activities, each user should have at least READ permission. Permissions for the Domains page are defined separately.	READ	ALL	ALL
ACMLedgerGroup	Data	Ledger Group Editor	Determines the user permission for manipulating with data in Ledger Group Editor; since this data are used in some other ACM applications and activities, each user should have at least READ permission. Permissions for the Domains page are defined separately.	READ	ALL	ALL
ACMLedgerGroupM	Data	Ledger Group Editor	Determines the user permission for manipulating with additional domains setup in Ledger Group Editor, page Domains. Each user should have at least READ permission.	READ	READ	ALL
ACMLedgerLedgerGroup	Data	Ledger Group Editor	Determines the user permission for manipulating with data in Ledger Group Editor, page Ledgers; since this data are used in some other ACM applications and activities, each user should have at least READ permission.	READ	ALL	ALL
ACMLedgerM	Data	Ledger Editor	Determines the user permission for manipulating with additional domains setup in Ledger Editor, page Domains. Each user should have at least READ permission.	READ	READ	ALL

Object	Kind	Primary Editor	Description	ACM-CONTROLLER	ACM-ACCOUNTANT	ACM-MAIN-ACCOUNTANT
ACMLedgerPermission	Data	Ledger Editor	Determines the user permission for manipulating with data in Ledger Editor, page Ledger Permission; since this data are used in some other ACM applications and activities, each user should have at least READ permission.	READ	READ	ALL
ACMLedgerPersAccType	Data	Account Type Editor	Determines the user permission for making the attribute overriding setup in Account Type Editor, page Attribure Overriding, in the ledger perspective. Each user should have at least READ permission.	READ	ALL	ALL
ACMLedgerPersAccount	Data	Account Editor	Determines the user permission for making the attribute overriding setup in Account Editor, page Attribure Overriding, in the ledger perspective. Each user should have at least READ permission.	READ	ALL	ALL
ACMLedgerPersCC	Data	Cost Center Editor	Determines the user permission for making the attribute overriding setup in Cost Center Editor, page Attribure Overriding, in the ledger perspective. Each user should have at least READ permission.	READ	ALL	ALL
ACMLedgerPersPeriod	Data	Period Editor	Determines the user permission for making the attribute overriding setup in Period Editor, page Attribure Overriding, in the ledger perspective. Each user should have at least READ permission.	READ	ALL	ALL
ACMLedgerPersProject	Data	Project Editor	Determines the user permission for making the attribute overriding setup in Project Editor, page Attribure Overriding, in the ledger perspective. Each user should have at least READ permission.	READ	ALL	ALL
ACMLock	System		System object; a permission for this object is not checked by default.			

Object	Kind	Primary Editor	Description	ACM-CONTROLLER	ACM-ACCOUNTANT	ACM-MAIN-ACCOUNTANT
ACMMapRuleParamSrc	Data	Accounting Parameter Configuration Editor	Determines the user permission for manipulating with data in Accounting Parameter Configuration Editor; since this data are used in some other ACM applications and activities, each user should have at least READ permission. Note that by default only users from the ACM-ADMIN group have permission ALL for this object.	READ	READ	READ
ACMMappedValue	Data	Value Mapping Editor	Determines the user permission for manipulating with data in Value Mapping Editor; since this data are used in some other ACM applications and activities, each user should have at least READ permission.	READ	ALL	ALL
ACMMappingAction	Data	Accounting Mapping Rule Editor	Determines the user permission for manipulating with data in Accounting Mapping Rule Editor, page Actions; since this data are used in some other ACM applications and activities, each user should have at least READ permission.	READ	ALL	ALL
ACMMappingDataSource	List		System table containing values for a selection list used in ACM applications or the table of key values; you should only read data from this table.	READ	READ	READ
ACMMappingRule	Data	Accounting Mapping Rule Editor	Determines the user permission for manipulating with data in Accounting Mapping Rule Editor; since this data are used in some other ACM applications and activities, each user should have at least READ permission. Permissions for the Actions page are defined separately.	READ	ALL	ALL
ACMMatchingType	List		System table containing values for a selection list used in ACM applications or the table of key values; you should only read data from this table.	READ	READ	ALL
ACMNumberSequence	System		System object; a permission for this object is not checked by default.	READ	ALL	ALL

Object	Kind	Primary Editor	Description	ACM-CONTROLLER	ACM-ACCOUNTANT	ACM-MAIN-ACCOUNTANT
ACMPLCategory	List		System table containing values for a selection list used in ACM applications or the table of key values; you should only read data from this table.	READ	ALL	ALL
ACMParamList	Data	Accounting Parameter List Editor	Determines the user permission for manipulating with data in Accounting Parameter List Editor; since this data are used in some other ACM applications and activities, each user should have at least READ permission.	READ	ALL	ALL
ACMParamListValue	Data	Accounting Parameter List Editor	Determines the user permission for manipulating with data in Accounting Parameter List Editor, page Accounting Parameter List Value; since this data are used in some other ACM applications and activities, each user should have at least READ permission.	READ	ALL	ALL
ACMPeriod	Data	Period Editor	Determines the user permission for manipulating with data in Period Editor; since this data are used in some other ACM applications and activities, each user should have at least READ permission.	READ	ALL	ALL
ACMPeriodCategory	List		System table containing values for a selection list used in ACM applications or the table of key values; you should only read data from this table.	READ	ALL	ALL
ACMPeriodOrderNumber	System		System table containing the order of all the periods defined within a period set.	ALL	ALL	ALL
ACMPeriodSet	Data	Period Set Editor	Determines the user permission for manipulating with data in Period Set Editor; since this data are used in some other ACM applications and activities, each user should have at least READ permission. Permissions for the Domains page are defined separately. Permissions for the Domains page are defined separately.	READ	ALL	ALL

Object	Kind	Primary Editor	Description	ACM-CONTROLLER	ACM-ACCOUNTANT	ACM-MAIN-ACCOUNTANT
ACMPeriodSetM	Data	Period Set Editor	Determines the user permission for manipulating with additional domains setup in Period Set Editor, page Domains. Each user should have at least READ permission.	READ	READ	ALL
ACMPeriodType	List		System table containing values for a selection list used in ACM applications or the table of key values; you should only read data from this table.	READ	ALL	ALL
ACMPerspective	System		System object; a permission for this object is not checked by default.	ALL	ALL	ALL
ACMProject	Data	Project Editor	Determines the user permission for manipulating with data in Project Editor; since this data are used in some other ACM applications and activities, each user should have at least READ permission.	READ	ALL	ALL
ACMRebookingDate	List		System table containing values for a selection list used in ACM applications or the table of key values; you should only read data from this table.	READ	ALL	ALL
ACMRebookingMethod	List		System table containing values for a selection list used in ACM applications or the table of key values; you should only read data from this table.	READ	ALL	ALL
ACMRebookingMoment	List		System table containing values for a selection list used in ACM applications or the table of key values; you should only read data from this table.	READ	ALL	ALL
ACMRebookingRolling	List		System table containing values for a selection list used in ACM applications or the table of key values; you should only read data from this table.	READ	ALL	ALL

Object	Kind	Primary Editor	Description	ACM-CONTROLLER	ACM-ACCOUNTANT	ACM-MAIN-ACCOUNTANT
ACMRebookingType	Data	Rebooking Type Editor	Determines the user permission for manipulating with data in Rebooking Type Editor; since this data are used in some other ACM applications and activities, each user should have at least READ permission.	READ	ALL	ALL
ACMRestrictionMode	List		System table containing values for a selection list used in ACM applications or the table of key values; you should only read data from this table.	READ	ALL	ALL
ACMRevCancelState	System		System object; a permission for this object is not checked by default.	READ	ALL	ALL
ACMRule	Data	Accounting Entry Rule Editor	Determines the user permission for manipulating with data in Rule Editor; since this data are used in some other ACM applications and activities, each user should have at least READ permission. Permissions for the Domains page are defined separately.	READ	ALL	ALL
ACMRuleM	Data	Accounting Entry Rule Editor	Determines the user permission for manipulating with additional domains setup in Rule Editor, page Domains. Each user should have at least READ permission.	READ	READ	ALL
ACMSign	List		System table containing values for a selection list used in ACM applications or the table of key values; you should only read data from this table.	READ	READ	READ
ACMSpecialTreatment	List		System table containing values for a selection list used in ACM applications or the table of key values; you should only read data from this table.	READ	ALL	ALL
ACMStmtReportRow	Data	Accounting Statement Report Template Editor	Determines the user permission for manipulating with data in Accounting Statement Report Editor, page Statement Report Row; since this data are used in some other ACM applications and activities, each user should have at least READ permission.	READ	READ	ALL

Object	**Kind**	**Primary Editor**	**Description**	**ACM-CONTROLLER**	**ACM-ACCOUNTANT**	**ACM-MAIN-ACCOUNTANT**
ACMStmtReportTmpl	Data	Accounting Statement Report Template Editor	Determines the user permission for manipulating with data in Accounting Statement Report Editor; since this data are used in some other ACM applications and activities, each user should have at least READ permission.	READ	READ	ALL
ACMTimeBand	Data	Time Band Set Editor	Determines the user permission for manipulating with data in Time Band Set Editor, page Time Band; since this data are used in some other ACM applications and activities, each user should have at least READ permission.	READ	ALL	ALL
ACMTimeBandSet	Data	Time Band Set Editor	Determines the user permission for manipulating with data in Time Band Set Editor; since this data are used in some other ACM applications and activities, each user should have at least READ permission.	READ	ALL	ALL
ACMTransactionSign	List		System table containing values for a selection list used in ACM applications or the table of key values; you should only read data from this table.	READ	READ	READ
ACMUserPerspective	Data	Accounting Perspective Editor	Determines the user permission for manipulating with data in Accounting Perspective Editor; since this data are used in some other ACM applications and activities, each user should have at least READ permission.	ALL	ALL	ALL
ACMUserSetup	System			ALL	ALL	ALL
ACMUserSetupData	System			ALL	ALL	ALL
ACMValueMapping	Data	Value Mapping Editor	Determines the user permission for manipulating with data in Value Mapping Editor; since this data are used in some other ACM applications and activities, each user should have at least READ permission. Permissions for the Domains page are defined separately.	READ	ALL	ALL

Object	Kind	Primary Editor	Description	ACM-CONTROLLER	ACM-ACCOUNTANT	ACM-MAIN-ACCOUNTANT
ACMValueMappingM	Data	Value Mapping Editor	Determines the user permission for manipulating with additional domains setup in Value Mapping Editor, page Domains. Each user should have at least READ permission.	READ	READ	ALL
ACMVoucher	System		Key system talbe containing all information about the vouchers.	READ	ALL	ALL
ACMVoucherSource	List		System table containing values for a selection list used in ACM applications or the table of key values; a permission for this object is not checked by default.			
ACMVoucherState	List		System table containing values for a selection list used in ACM applications or the table of key values; you should only read data from this table.	READ	ALL	ALL
ACMVoucherType	Data	Voucher Type Editor	Determines the user permission for manipulating with data in Voucher Type Editor; since this data are used in some other ACM applications and activities, each user should have at least READ permission.	READ	ALL	ALL
ACMYesNo	List		System table containing values for a selection list used in ACM applications or the table of key values; you should only read data from this table.	READ	READ	ALL
Administrator Activity	System		System table needed for the execution of the Accounting System Administration Task activity. Note that by default only users from the ACM-ADMIN group have permission ALL for this object.			

Note: In the Object Hierarchy Editor, all the ACM objects are of the Editor type, i.e. they all have ON the switch *Editor* in the lower part of the Object Hierarchy Editor.

B.3 ACM-MAIN-ACCOUNTANT vs. ACM-ADMIN

ACM-MAIN-ACCOUNTANT, the second most powerfull user group, can do almost the same as ACM-ADMIN with the following exceptions:

- Only READ permision ACM-SYSTEM domain
- No possibility to run the Dynamic Data Verification report (for details see *6.1 Accounting dynamic data verification report* on page 35).
- No possibility to run the Accounting System Administration Task activity (for details see *6.2 Accounting System Administration Task activity* on page 38).
- Only READ permission for object(s) needed for defining event to entry mapping via the Event to Entry Mapping Editor.
- Only READ permission for object(s) needed for defining accounting parameters via the Accounting Parameter Editor.

Note that ACM-ADMIN has permission ALL for all the above listed objects and domain.

Besides the first possibility all other permissions are available due to the following object permissions:

Object	Kind	Primary Editor	Description	ACM-MAIN-ACCOUNTANT	ACM-ADMIN
ACMDynamicDataVerification	System		System table needed for the execution of the Accounting Dynamic Data Verification report.	ALL	ALL
ACMEventEntryMap	Data	Event to Entry Mapping Editor	Determines the user permission for manipulating with data in the Event to Entry Mapping Editor.	ALL	ALL
ACMMapRuleParamSrc	Data	Accounting Parameter Configuration Editor	Determines the user permission for manipulating with data in Accounting Parameter Configuration Editor.	ALL	ALL
AdministratorActivity	System		System table needed for the execution of the Accounting System Administration Task activity.	ALL	ALL

B.4 ACM-ACCOUNTANT vs. ACM-MAIN-ACCOUNTANT

ACM-ACCOUNTANT can do almost the same as ACM-MAIN-ACCOUNTANT with the following exceptions:

- Only READ permission for objects having M at the end of their name (e.g. ACMLedgerM) used for assigning multiple domains on some entities (e.g. ledger). Note that these so called secondary domains add a reading permission for the domains defined in the "Domains" tab of the editors, e.g. in the Ledger editor. The READ permission on the M table allows only to see domains defined in that tab, but no editing.
- Only READ permission for objects needed for the definition of ERP static data (ACMERPExport Def, ACMERPSystemDef, ACMERPTarget).

- Only READ permission for defining ledger perissions via the Ledger Editor (object ACMLedgerPermission).
- Only READ permission for defining accounting statement report templates (objects ACMStmtReportRow, ACMStmatReportTmpl).

ACM-MAIN-ACCOUNTANT has permission ALL for all these objects.

B.5 Handling of M tables

Some objects in the table listing the accounting objects end by letter M. Permissions on these objects are checked by the corresponding editors which allow assigning multiple domains on the same entity (e.g. ledger). These secondary domains allow just a reading permission and they are defined in the "Domains" tab of the editors, e.g. in the Ledger editor. The READ permission on the M table allows to see domains defined in that tab. For a possibility to add/remove a domain in the tab you need:

- permission ALL for the appropriate M table,
- at least permission READ for the main object (e.g. ACMLedger), and
- permission ALL for the domain, in which the particular enity (e.g. ledger) is defined.

Appendix C ERP document structure

This appendix contains a default structure of the ERP output (ERP document), so called TremaXMLDocument, as generated by the ACM server. Its whole structure is described by the XSD as well as by the table.

C.1 ERP output - description in table

The following table shows the mapping of exported ACM fields to the XML EPR output structure:

XML level	XML tag	Short description	Long description	ACM source table	ACM source field
Header	ID	ERP Document System ID	Unique internal ID for the export document	ACMERPDoc	id
Header	OBJ_KEY	ERP Document(Object) Key	Unique user ID for the export document	ACMERPDoc	document_number
Header	DOC_DATE	Document Date	Date assigned to export document	ACMERPDoc	document_date
Header	PSTNG_DATE	Posting Date	Posting date of the export document, determines the selection of the posting period in NACOS	ACMERPDoc	posting_date
Header	FISC_YEAR	Fiscal Year	Fiscal year the export document is assigned to. **This tag appears if you defined ERP value for your fiscal year; is mandatory for export to SAP.**	ACMERPDoc	erp_year
Header	FIS_PERIOD	Fiscal Period	Accounting period in SAP the export document is assigned to. **This tag appears if you defined ERP value for your posting period.**	ACMERPDoc	erp_period

XML level	XML tag	Short description	Long description	ACM source table	ACM source field
Header	COMP_CODE	Ledger Code (Company Code)	Target ledger in SAP for the export document. **This tag appears if you defined ERP Ledger Code in the ERP Target editor; is mandatory for export to SAP.**	ACMERPTarget	ledger_code
Header	ERP_SYS	Logical ERP Target System	Technical definition of the target system	ACMERPSystem Def	system_name
Header	ERP_TYPE	ERP Target System Type	Target system type (technical)	ACMERPTargetTy pe	id_user
Header	GLM_OWNER	Ledger user ID	User ID for the ACM ledger the export document is related with	ACMLedger	id_user
Header	STATE_ID	ERP document state	ERP document state system ID	ACMERPDoc	state_id
Header	EXP_USER	Export User	User to initiate the export	ACMERPDoc	last_state_user
Header	DOC_TYPE	ERP document state	ERP document state: 1 = BY_VOUCHER 2 = BY_ENTRY 3 = BY_BALANCCE	ACMERPDoc	erp_type_id
Header	DOC_FLAGS	ERP document flags	ERP document flags	ACMERPDoc	flags
Header	LEDGER_ID	Ledger system ID	Ledger system ID	ACMLedger	id
Header	LEDGER_ID_US ER	Ledger user ID	Ledger user ID	ACMLedger	id_user
Header	ACM_PERIOD	Voucher period user ID	ACM posting period assigned to the voucher	ACMPeriod	id_user
Line Item	VOUCHER_DESC	Voucher description	Description field on the voucher level	ACMVoucher	description
Line Item	DOC_CURR	Document Currency	Document currency for the entry/balance	Currency	id
Line Item	DOC_AMT	Document Amount	Entry/balance amount in document currency	ACMEntry	amount
Line Item	BOOK_CURR	Booking Currency	Booking currency for the entry/balance (EUR for DCAG)	Currency	id
Line Item	BOOK_AMT	Booking Amount	Entry amount in booking currency	ACMEntry	bookingAmount

XML level	XML tag	Short description	Long description	ACM source table	ACM source field
Line Item	ERP_ACC	ERP Account	Target account number. **This tag appears if you defined ERP value for your account; is mandatory for export to SAP. The specified value is automatically completed by zeros from left to reach 10 characters.**	ACMERPDocItem	erp_account
Line Item	GLM_ACC	GLM Account	User account number for the source account in ACM	ACMAccount	id_user
Line Item	GLM_ACC_TXT	GLM Account Name	User account name for the source account in ACM	ACMAccount	name
Line Item	VOUCH_REV	Voucher Reverse State	Flag to indicate reversing voucher	ACMRevCancelState	id_user
Line Item	VOUCH_TYPE	ACM Voucher Type	Voucher type assigned in ACM to indicate purpose of the related posting	ACMVoucherType	id_user
Line Item	ENTRY_ID	ACM Entry ID	Unique reference ID for the entry in ACM	ACMEntry	id
Line Item	VOUCH_ID	ACM Voucher ID	Voucher number (user ID) for the voucher in ACM	ACMVoucher	voucher_number
Line Item	TRAN_NUM	ACM Entry Origin + ACM Reference ID	TRM/CMM/ACM **transaction number** if applicable	ACMEntry	number
Line Item	PORTF	Portfolio	Portfolio	Portfolio	id
Line Item	INSTR	Instrument	Instrument	ACMEntry	instrument_id
Line Item	INSTR_GRP	Instrument Group	Instrument Group	UMPathNode	path
Line Item	CPTY	Counerarty	Counerarty	Client	client_id
Line Item	CPTY_GRP	Counerarty Group	Counerarty Group	ClientGroup	id
Line Item	ISSR	Issuer	Issuer	Client	client_id
Line Item	ISSR_GRP	Issuer Group	Issuer Group	ClientGroup	id
Line Item	T_OPN_DATE	Trans Opening Date	Opening date for the TRM transaction if applicable	ACMEntry	transaction_opening_date
Line Item	T_MAT_DATE	Trans Maturity Date	Maturity date for the TRM transaction if applicable	ACMEntry	transaction_maturity_date

XML level	XML tag	Short description	Long description	ACM source table	ACM source field
Line Item	T_VAL_DATE	Trans Value Date	Value date for the TRM transaction if applicable	ACMEntry	transaction_value_date
Line Item	VAL_DATE	ACM Value Date	Event date in ACM for the booking (used to determine the posting period in ACM)	ACMEntry	event_date
Line Item	BOOK_DATE	ACM Bookung Date	Booking date in ACM (date when the voucher was created)	ACMEntry	booking_date
Line Item	TRADE_ID	Trading Partner	Counterparty Code	ACMERPDocItem	erp_cpty_code
Line Item	COSTCENTER	Cost Center	Cost center attributed to the entry in ACM (used to assign P&L account entries to affiliated entities)	ACMERPDocItem	erp_cost_center
Line Item	ENTRY_DESC	ACM Entry Description	Description field for the entry	ACMERPDocItem	description
Line Item	ORIG_ID	ACM Entry Origin ID	Origin for the entry, e.g. "TRM"	ACMOriginEntity	id_user
Line Item	ORIG_CODE	ACM Entry Origin Code	Shorthand for Origin Entity	ACMOriginEntity	code
Line Item	PROJ_ID	ERP Project	Project attributed to the entry in ACM (might be employed in the future)	ACMERPDocItem	erp_project
Line Item	P0 .. 9	ACM Entry Param_0 .. 9	Entry parameter	ACMEntry	param0 .. 9
Line Item	A_P0 .. 9	ACM Account Param_0 .. 9	Account parameter	ACMAccount	parameter_0 .. 9
Line Item	VOUCHER_DESC	Voucher description	Description field on the voucher level	ACMVoucher	description
Line Item	DU_ID	Document Unit System ID	Document Unit System ID	ACMERPDocUnit	id
Line Item	DI_ID	Document Item System ID	Document Item System ID	ACMERPDocItem	id
Line Item	DR_CR	Item DR/CR Sysrtem ID	Item DR/CR Sysrtem ID	ACMERPDocItem	dr_cr
Line Item	DR_CR_NAME	Item DR/CR	Item DR/CR	ACMDrCr	id_user
Line Item	ITEM_TYPE	Item Type	Item Type	ACMERPType	id_user
Line Item	UNIT_REV	Reversed DU System ID	Reversed DU System ID	ACMERPDocUnit	reversed_du_id

C.2 ERP output - description by XSD

Note: The whole structure is sent to a target system only if SAP-FI is used. For the FILE target type only the GLM element is stored.

```
<?xml version="1.0" encoding="ISO-8859-1" ?>

<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns="http://trema.com/acm" targetNamespace="http://trema.com/acm">

  <xsd:complexType name="TremaXMLDocument">

    <xsd:sequence>

      <xsd:element name="OBJECT_TYPE" type="xsd:string">

        <xsd:annotation>

          <xsd:documentation xml:lang="EN">

          Object Type For Accounting

          </xsd:documentation>

        </xsd:annotation>

      </xsd:element>

      <xsd:element name="PARTNER_PROFILE" type="xsd:string">

        <xsd:annotation>

          <xsd:documentation xml:lang="EN">

          Specifies Partner Profile for which message is intendent

          </xsd:documentation>

        </xsd:annotation>

      </xsd:element>

      <xsd:element name="DOCUMENT_TYPE" type="xsd:string">

        <xsd:annotation>

          <xsd:documentation xml:lang="EN">

          Specifies Document Type

          </xsd:documentation>

        </xsd:annotation>

      </xsd:element>

      <xsd:element name="FUNCTION_NAME" type="xsd:string">

        <xsd:annotation>

          <xsd:documentation xml:lang="EN">

          Specifies Function for which this message is intended for
          </xsd:documentation>
        </xsd:annotation>
      </xsd:element>
      <xsd:element name="GLM">
        <xsd:complexType>
          <xsd:sequence>
            <xsd:element name="Header">
              <xsd:annotation>
                <xsd:documentation xml:lang="EN">
                Header of Trema XML Document
                </xsd:documentation>
              </xsd:annotation>
              <xsd:complexType>
                <xsd:sequence>
                  <xsd:element name="ID" type="xsd:integer">
                    <xsd:annotation>
                    <xsd:documentation xml:lang="EN">
                    Trema ERP Document System ID
                    </xsd:documentation>
                    </xsd:annotation>
                  </xsd:element>
                  <xsd:element name="OBJ_KEY" type="xsd:string">
                    <xsd:annotation>
                    <xsd:documentation xml:lang="EN">
                    ERP Document Key
                    </xsd:documentation>
                    </xsd:annotation>
                  </xsd:element>
                  <xsd:element name="DOC_DATE" type="xsd:string">
                    <xsd:annotation>
                    <xsd:documentation xml:lang="EN">
                    Document Date (yyyyMMdd)
                    </xsd:documentation>
                    </xsd:annotation>
```

```
</xsd:element>
<xsd:element name="PSTNG_DATE" type="xsd:string">
  <xsd:annotation>
  <xsd:documentation xml:lang="EN">
  Posting Date (yyyyMMdd)
  </xsd:documentation>
  </xsd:annotation>
</xsd:element>
<xsd:element name="FISC_YEAR" type="xsd:string">
  <xsd:annotation>
  <xsd:documentation xml:lang="EN">
  Fiscal Year
  </xsd:documentation>
  </xsd:annotation>
</xsd:element>
<xsd:element name="FIS_PERIOD" type="xsd:string" minOccurs="0">
  <xsd:annotation>
  <xsd:documentation xml:lang="EN">
  Fiscal Period
  </xsd:documentation>
  </xsd:annotation>
</xsd:element>
<xsd:element name="GLM_OWNER" type="xsd:string">
  <xsd:annotation>
  <xsd:documentation xml:lang="EN">
  ACM Ledger
  </xsd:documentation>
  </xsd:annotation>
</xsd:element>
<xsd:element name="COMP_CODE" type="xsd:string">
  <xsd:annotation>
  <xsd:documentation xml:lang="EN">
  Ledger Code (Company Code)
  </xsd:documentation>
  </xsd:annotation>
</xsd:element>
<xsd:element name="DOC_TYPE" type="xsd:integer" minOccurs="0">
  <xsd:annotation>
  <xsd:documentation xml:lang="EN">
  Trema ERP Document Type
  </xsd:documentation>
  </xsd:annotation>
</xsd:element>
<xsd:element name="EXP_USER" type="xsd:string">
  <xsd:annotation>
  <xsd:documentation xml:lang="EN">
  Export User
  </xsd:documentation>
  </xsd:annotation>
</xsd:element>
<xsd:element name="ERP_SYS" type="xsd:string">
  <xsd:annotation>
  <xsd:documentation xml:lang="EN">
  Logical ERP Target System
  </xsd:documentation>
  </xsd:annotation>
</xsd:element>
<xsd:element name="ERP_TYPE" type="xsd:string">
  <xsd:annotation>
  <xsd:documentation xml:lang="EN">
  ERP Target System Type
  </xsd:documentation>
  </xsd:annotation>
</xsd:element>
<xsd:element name="STATE_ID" type="xsd:integer" minOccurs="0">
  <xsd:annotation>
  <xsd:documentation xml:lang="EN">
  ERP Document State System ID
```

```
          </xsd:documentation>
          </xsd:annotation>
        </xsd:element>
        <xsd:element name="DOC_FLAGS" type="xsd:integer" minOccurs="0">
          <xsd:annotation>
          <xsd:documentation xml:lang="EN">
          ERP Document Flags
          </xsd:documentation>
          </xsd:annotation>
        </xsd:element>
        <xsd:element name="LEDGER_ID" type="xsd:integer" minOccurs="0">
          <xsd:annotation>
          <xsd:documentation xml:lang="EN">
          ACM Ledger System ID
          </xsd:documentation>
          </xsd:annotation>
        </xsd:element>
        <xsd:element name="ACM_PERIOD" type="xsd:string" minOccurs="0">
          <xsd:annotation>
          <xsd:documentation xml:lang="EN">
          ACM Period
          </xsd:documentation>
          </xsd:annotation>
        </xsd:element>
      </xsd:sequence>
    </xsd:complexType>
  </xsd:element>
  <xsd:element name="Item" minOccurs="2" maxOccurs="unbounded">
    <xsd:annotation>
      <xsd:documentation xml:lang="EN">
      Each document must balance -&gt;at least two items in each document
      </xsd:documentation>
    </xsd:annotation>
    <xsd:complexType>
```

```
        <xsd:sequence>
          <xsd:element name="DOC_CURR" type="xsd:string">
            <xsd:annotation>
            <xsd:documentation xml:lang="EN">
            Document Currency
            </xsd:documentation>
            </xsd:annotation>
          </xsd:element>
          <xsd:element name="BOOK_CURR" type="xsd:string">
            <xsd:annotation>
            <xsd:documentation xml:lang="EN">
            Booking Currency
            </xsd:documentation>
            </xsd:annotation>
          </xsd:element>
          <xsd:element name="DOC_AMT" type="xsd:string">
            <xsd:annotation>
            <xsd:documentation xml:lang="EN">
            Document Amount
            </xsd:documentation>
            </xsd:annotation>
          </xsd:element>
          <xsd:element name="BOOK_AMT" type="xsd:string">
            <xsd:annotation>
            <xsd:documentation xml:lang="EN">
            Booking Amount
            </xsd:documentation>
            </xsd:annotation>
          </xsd:element>
          <xsd:element name="ERP_ACC" type="xsd:string">
            <xsd:annotation>
            <xsd:documentation xml:lang="EN">
            ERP Account
            </xsd:documentation>
```

```
  </xsd:annotation>
</xsd:element>
<xsd:element name="GLM_ACC" type="xsd:string" minOccurs="0">
  <xsd:annotation>
  <xsd:documentation xml:lang="EN">
  ACM Account
  </xsd:documentation>
  </xsd:annotation>
</xsd:element>
<xsd:element name="GLM_ACC_TXT" type="xsd:string" minOccurs="0">
  <xsd:annotation>
  <xsd:documentation xml:lang="EN">
  ACM Account Name
  </xsd:documentation>
  </xsd:annotation>
</xsd:element>
<xsd:element name="V_ID" type="xsd:integer" minOccurs="0">
  <xsd:annotation>
  <xsd:documentation xml:lang="EN">
  ACM Voucher System ID
  </xsd:documentation>
  </xsd:annotation>
</xsd:element>
<xsd:element name="ENTRY_ID" type="xsd:integer" minOccurs="0">
  <xsd:annotation>
  <xsd:documentation xml:lang="EN">
  ACM Entry System ID
  </xsd:documentation>
  </xsd:annotation>
</xsd:element>
<xsd:element name="VOUCH_ID" type="xsd:string" minOccurs="0">
  <xsd:annotation>
  <xsd:documentation xml:lang="EN">
  Voucher Number
  </xsd:documentation>
  </xsd:annotation>
</xsd:element>
<xsd:element name="VOUCH_TYPE" type="xsd:string" minOccurs="0">
  <xsd:annotation>
  <xsd:documentation xml:lang="EN">
  ACM Voucher Type
  </xsd:documentation>
  </xsd:annotation>
</xsd:element>
<xsd:element name="VOUCH_REV" type="xsd:string" minOccurs="0">
  <xsd:annotation>
  <xsd:documentation xml:lang="EN">
  Voucher Reverse State
  </xsd:documentation>
  </xsd:annotation>
</xsd:element>
<xsd:element name="PORTF" type="xsd:string" minOccurs="0">
  <xsd:annotation>
  <xsd:documentation xml:lang="EN">
  Portfolio
  </xsd:documentation>
  </xsd:annotation>
</xsd:element>
<xsd:element name="CPTY" type="xsd:string" minOccurs="0">
  <xsd:annotation>
  <xsd:documentation xml:lang="EN">
  Counterparty
  </xsd:documentation>
  </xsd:annotation>
</xsd:element>
<xsd:element name="INSTR" type="xsd:string" minOccurs="0">
  <xsd:annotation>
  <xsd:documentation xml:lang="EN">
  Instrument
  </xsd:documentation>
  </xsd:annotation>
</xsd:element>
<xsd:element name="CPTY_GRP" type="xsd:string" minOccurs="0">
  <xsd:annotation>
  <xsd:documentation xml:lang="EN">
  Counterparty Group
  </xsd:documentation>
  </xsd:annotation>
</xsd:element>
<xsd:element name="ISSR" type="xsd:string" minOccurs="0">
  <xsd:annotation>
  <xsd:documentation xml:lang="EN">
  Issuer
  </xsd:documentation>
  </xsd:annotation>
</xsd:element>
<xsd:element name="ISSR_GRP" type="xsd:string" minOccurs="0">
  <xsd:annotation>
  <xsd:documentation xml:lang="EN">
  Issuer Group
  </xsd:documentation>
  </xsd:annotation>
</xsd:element>
<xsd:element name="TRAN_NUM" type="xsd:string" minOccurs="0">
  <xsd:annotation>
  <xsd:documentation xml:lang="EN">
  ACM Entry Origin + ACM Reference ID
  </xsd:documentation>
  </xsd:annotation>
</xsd:element>
<xsd:element name="T_OPN_DATE" type="xsd:string" minOccurs="0">
  <xsd:annotation>
  <xsd:documentation xml:lang="EN">
  Transaction Opening Date
  </xsd:documentation>
  </xsd:annotation>
</xsd:element>
<xsd:element name="T_MAT_DATE" type="xsd:string" minOccurs="0">
  <xsd:annotation>
  <xsd:documentation xml:lang="EN">
  Transaction Maturity Date
  </xsd:documentation>
  </xsd:annotation>
</xsd:element>
<xsd:element name="T_VAL_DATE" type="xsd:string" minOccurs="0">
  <xsd:annotation>
  <xsd:documentation xml:lang="EN">
  Transaction Value Date
  </xsd:documentation>
  </xsd:annotation>
</xsd:element>
<xsd:element name="TRADE_ID" type="xsd:string" minOccurs="0">
  <xsd:annotation>
  <xsd:documentation xml:lang="EN">
  Trading Partner
  </xsd:documentation>
  </xsd:annotation>
</xsd:element>
<xsd:element name="BOOK_DATE" type="xsd:string" minOccurs="0">
  <xsd:annotation>
  <xsd:documentation xml:lang="EN">
  ACM Booking Date
  </xsd:documentation>
  </xsd:annotation>
</xsd:element>
<xsd:element name="COSTCENTER" type="xsd:string" minOccurs="0">
```

```
  <xsd:annotation>
  <xsd:documentation xml:lang="EN">
  ERP Cost Center
  </xsd:documentation>
  </xsd:annotation>
</xsd:element>
<xsd:element name="ENTRY_DESC" type="xsd:string" minOccurs="0">
  <xsd:annotation>
  <xsd:documentation xml:lang="EN">
  ACM Entry Description
  </xsd:documentation>
  </xsd:annotation>
</xsd:element>
<xsd:element name="VAL_DATE" type="xsd:string">
  <xsd:annotation>
  <xsd:documentation xml:lang="EN">
  ACM Value Date
  </xsd:documentation>
  </xsd:annotation>
</xsd:element>
<xsd:element name="ORIG_ID" type="xsd:string" minOccurs="0">
  <xsd:annotation>
  <xsd:documentation xml:lang="EN">
  ACM Entry Origin ID
  </xsd:documentation>
  </xsd:annotation>
</xsd:element>
<xsd:element name="ORIG_CODE" type="xsd:string" minOccurs="0">
  <xsd:annotation>
  <xsd:documentation xml:lang="EN">
  ACM Entry Origin Code
  </xsd:documentation>
  </xsd:annotation>
</xsd:element>
```

```
<xsd:element name="PROJ_ID" type="xsd:string" minOccurs="0">
  <xsd:annotation>
  <xsd:documentation xml:lang="EN">
  ERP Project
  </xsd:documentation>
  </xsd:annotation>
</xsd:element>
<xsd:element name="P0" type="xsd:string" minOccurs="0">
  <xsd:annotation>
  <xsd:documentation xml:lang="EN">
  ACM Entry Parameter 0
  </xsd:documentation>
  </xsd:annotation>
</xsd:element>
<xsd:element name="P1" type="xsd:string" minOccurs="0">
  <xsd:annotation>
  <xsd:documentation xml:lang="EN">
  ACM Entry Parameter 1
  </xsd:documentation>
  </xsd:annotation>
</xsd:element>
<xsd:element name="P2" type="xsd:string" minOccurs="0">
  <xsd:annotation>
  <xsd:documentation xml:lang="EN">
  ACM Entry Parameter 2
  </xsd:documentation>
  </xsd:annotation>
</xsd:element>
<xsd:element name="P3" type="xsd:string" minOccurs="0">
  <xsd:annotation>
  <xsd:documentation xml:lang="EN">
  ACM Entry Parameter 3
  </xsd:documentation>
  </xsd:annotation>
```

```
</xsd:element>
<xsd:element name="P4" type="xsd:string" minOccurs="0">
  <xsd:annotation>
  <xsd:documentation xml:lang="EN">
  ACM Entry Parameter 4
  </xsd:documentation>
  </xsd:annotation>
</xsd:element>
<xsd:element name="P5" type="xsd:string" minOccurs="0">
  <xsd:annotation>
  <xsd:documentation xml:lang="EN">
  ACM Entry Parameter 5
  </xsd:documentation>
  </xsd:annotation>
</xsd:element>
<xsd:element name="P6" type="xsd:string" minOccurs="0">
  <xsd:annotation>
  <xsd:documentation xml:lang="EN">
  ACM Entry Parameter 6
  </xsd:documentation>
  </xsd:annotation>
</xsd:element>
<xsd:element name="P7" type="xsd:string" minOccurs="0">
  <xsd:annotation>
  <xsd:documentation xml:lang="EN">
  ACM Entry Parameter 7
  </xsd:documentation>
  </xsd:annotation>
</xsd:element>
<xsd:element name="P8" type="xsd:string" minOccurs="0">
  <xsd:annotation>
  <xsd:documentation xml:lang="EN">
  ACM Entry Parameter 8
  </xsd:documentation>
  </xsd:annotation>
</xsd:element>
<xsd:element name="P9" type="xsd:string" minOccurs="0">
  <xsd:annotation>
  <xsd:documentation xml:lang="EN">
  ACM Entry Parameter 9
  </xsd:documentation>
  </xsd:annotation>
</xsd:element>
<xsd:element name="A_P0" type="xsd:string" minOccurs="0">
  <xsd:annotation>
  <xsd:documentation xml:lang="EN">
  ACM Account Parameter 0
  </xsd:documentation>
  </xsd:annotation>
</xsd:element>
<xsd:element name="A_P1" type="xsd:string" minOccurs="0">
  <xsd:annotation>
  <xsd:documentation xml:lang="EN">
  ACM Account Parameter 1
  </xsd:documentation>
  </xsd:annotation>
</xsd:element>
<xsd:element name="A_P2" type="xsd:string" minOccurs="0">
  <xsd:annotation>
  <xsd:documentation xml:lang="EN">
  ACM Account Parameter 2
  </xsd:documentation>
  </xsd:annotation>
</xsd:element>
<xsd:element name="A_P3" type="xsd:string" minOccurs="0">
  <xsd:annotation>
  <xsd:documentation xml:lang="EN">
  ACM Account Parameter 3
  </xsd:documentation>
  </xsd:annotation>
</xsd:element>
<xsd:element name="A_P4" type="xsd:string" minOccurs="0">
  <xsd:annotation>
  <xsd:documentation xml:lang="EN">
  ACM Account Parameter 4
  </xsd:documentation>
  </xsd:annotation>
</xsd:element>
<xsd:element name="A_P5" type="xsd:string" minOccurs="0">
  <xsd:annotation>
  <xsd:documentation xml:lang="EN">
  ACM Account Parameter 5
  </xsd:documentation>
  </xsd:annotation>
</xsd:element>
<xsd:element name="A_P6" type="xsd:string" minOccurs="0">
  <xsd:annotation>
  <xsd:documentation xml:lang="EN">
  ACM Account Parameter 6
  </xsd:documentation>
  </xsd:annotation>
</xsd:element>
<xsd:element name="A_P7" type="xsd:string" minOccurs="0">
  <xsd:annotation>
  <xsd:documentation xml:lang="EN">
  ACM Account Parameter 7
  </xsd:documentation>
  </xsd:annotation>
</xsd:element>
<xsd:element name="A_P8" type="xsd:string" minOccurs="0">
  <xsd:annotation>
  <xsd:documentation xml:lang="EN">
  ACM Account Parameter 8
  </xsd:documentation>
  </xsd:annotation>
</xsd:element>
<xsd:element name="A_P9" type="xsd:string" minOccurs="0">
  <xsd:annotation>
  <xsd:documentation xml:lang="EN">
  ACM Account Parameter 9
  </xsd:documentation>
  </xsd:annotation>
</xsd:element>
<xsd:element name="INSTR_GRP" type="xsd:string" minOccurs="0">
  <xsd:annotation>
  <xsd:documentation xml:lang="EN">
  Instrument group
  </xsd:documentation>
  </xsd:annotation>
</xsd:element>
<xsd:element name="D_ID" type="xsd:integer" minOccurs="0">
  <xsd:annotation>
  <xsd:documentation xml:lang="EN">
  ACM ERP Document System ID
  </xsd:documentation>
  </xsd:annotation>
</xsd:element>
<xsd:element name="DU_ID" type="xsd:integer" minOccurs="0">
  <xsd:annotation>
  <xsd:documentation xml:lang="EN">
  ACM ERP Document Unit System ID
  </xsd:documentation>
  </xsd:annotation>
</xsd:element>
<xsd:element name="DI_ID" type="xsd:integer" minOccurs="0">
  <xsd:annotation>
  <xsd:documentation xml:lang="EN">
  ACM ERP Document Item System ID
  </xsd:documentation>
  </xsd:annotation>
</xsd:element>
<xsd:element name="DR_CR_NAME" type="xsd:string" minOccurs="0">
  <xsd:annotation>
  <xsd:documentation xml:lang="EN">
  Item DR/CR
  </xsd:documentation>
  </xsd:annotation>
</xsd:element>
<xsd:element name="ITEM_TYPE" type="xsd:string" minOccurs="0">
  <xsd:annotation>
  <xsd:documentation xml:lang="EN">
  ACM ERP Item Type
  </xsd:documentation>
  </xsd:annotation>
</xsd:element>
<xsd:element name="UNIT_REV" type="xsd:integer" minOccurs="0">
  <xsd:annotation>
  <xsd:documentation xml:lang="EN">
  Reversed ACM ERP DU System ID
  </xsd:documentation>
  </xsd:annotation>
</xsd:element>
<xsd:element name="VOUCHER_DESC" type="xsd:string" minOccurs="0">
  <xsd:annotation>
  <xsd:documentation xml:lang="EN">
  ACM Voucher Description
  </xsd:documentation>
  </xsd:annotation>
</xsd:element>
<xsd:element name="LEDGER_ID" type="xsd:integer" minOccurs="0">
```

```
                    <xsd:annotation>
                    <xsd:documentation xml:lang="EN">
                    ACM Ledger System ID
                    </xsd:documentation>
                    </xsd:annotation>
                  </xsd:element>
                </xsd:sequence>
              </xsd:complexType>
            </xsd:element>
          </xsd:sequence>
        </xsd:complexType>
      </xsd:element>
    </xsd:sequence>
  </xsd:complexType>
</xsd:schema>
```

C.3 ERP output in XML format - example

C.3.1 About the structure

A structure of the ERP export file is very close for both ERP export modes: Voucher and Account Mode (either by entries or balances). It differs in a few tags only:

Export Mode	Document Header	Document Item	
	DOC_TYPE	ITEM_TYPE	UNIT_TYPE
VOUCHER	1	BY_VOUCHER	BY_VOUCHER
ACCOUNT by ENTRY	2	BY_ENTRY ... for the account entry ENTRY_MIRROR ... for the technical mirror entry	BY_ENTRY
ACCOUNT by BALANCE	3	BY_BALANCE ... for the account movement BALANCE_MIRROR ... for the balance movement technical mirror entry/item	BY_BALANCE

Depending on the ERP target type the ERP_SYS and ERP_TYPE tags are filled as follows:

Target Type	Document Header	
	ERP_SYS	ERP_TYPE
SAP-FI	SAP BC/XI alias	SAP_FI
FILE	FILE	FILE

C.3.2 Example for Voucher Mode

Below you can see an example of the ERP output file in XML format for the **Voucher Mode** (i.e. all the entries from a voucher are transferred in one ERP document). Note that only tags, for which a value in ACM exists, are filled.

```
<GLM>
    <Header>
        <ID>10</ID>
        <OBJ_KEY>2005_00010</OBJ_KEY>
        <DOC_DATE>20080229</DOC_DATE>
        <PSTNG_DATE>20050414</PSTNG_DATE>
        <FISC_YEAR>2005</FISC_YEAR> .......... this tag appears if you defined ERP value for your
                                               fiscal year; is mandatory for export to SAP.
        <FISC_PERIOD>04</FISC_PERIOD> ...... this tag appears if you defined ERP value for your
                                               posting period.
        <COMP_CODE>1000</COMP_CODE> ..... this tag appears if you defined the ERP Ledger
                                               Code in the ERP Target Editor; is mandatory for
                                               export to SAP.
        <ERP_SYS>FILE</ERP_SYS>
        <ERP_TYPE>FILE</ERP_TYPE>
        <GLM_OWNER>WSS_LMTD</GLM_OWNER>
        <STATE_ID>1210</STATE_ID>
        <EXP_USER>CHARLIE</EXP_USER>
        <DOC_TYPE>1</DOC_TYPE>
        <DOC_FLAGS>0</DOC_FLAGS>
        <LEDGER_ID>142</LEDGER_ID>
        <ACM_PERIOD>2005-04</ACM_PERIOD>
    </Header>
    <Item>
        <DOC_CURR>USD</DOC_CURR>
        <DOC_AMT>500000.0000</DOC_AMT>
        <BOOK_CURR>EUR</BOOK_CURR>
        <BOOK_AMT>342000.0000</BOOK_AMT>
        <ERP_ACC>0000075000<ERP_ACC> ...... this tag appears if you defined ERP value for your
                                               account; is mandatory for export to SAP;
                                               the specified values is completed by zeros from
                                               left to reach 10 characters.
        <GLM_ACC>111</GLM_ACC> ............... user ID from ACM
        <GLM_ACC_ID>947</GLM_ACC_ID> ..... system ID
        <GLM_ACC_TXT>EUR C/A</GLM_ACC_TXT>
        <ENTRY_ID>3935</ENTRY_ID>
        <VOUCH_ID>DL-2005-003</VOUCH_ID>
        <V_ID>1401</V_ID>
        <VOUCH_TYPE>DL</VOUCH_TYPE>
        <TRAN_NUM>TRM-486</TRAN_NUM>
        <PORTF>WSS_SHORT_DEP_PTF_1</PORTF>
        <INSTR>SHORT_DEPO_EUR</INSTR>
        <CPTY>ABNAMRO</CPTY>
        <CPTY_GRP>EXTERNAL</CPTY_GRP>
        <ISSR>KOBA</ISSR>
        <ISSR_GRP>EXTERNAL</ISSR_GRP>
        <T_OPN_DATE>20050112</T_OPN_DATE>
        <T_MAT_DATE>20050414</T_MAT_DATE>
        <T_VAL_DATE>20050114</T_VAL_DATE>
        <BOOK_DATE>20080229</BOOK_DATE>
        <VAL_DATE>20050414</VAL_DATE>
        <ENTRY_DESC>WSS-CITI-EUR113</ENTRY_DESC>
        <ORIG_ID>TRM-TRANSACTION</ORIG_ID>
        <ORIG_CODE>TRM</ORIG_CODE>
        <DU_ID>14</DU_ID>
        <DI_ID>27</DI_ID>
        <DR_CR>1</DR_CR>
        <DR_CR_NAME>DR</DR_CR_NAME>
        <D_ID>10</D_ID>
        <ITEM_TYPE>BY_VOUCHER</ITEM_TYPE>
        <UNIT_TYPE>BY_VOUCHER</UNIT_TYPE>
        <VOUCH_REV>NONE</VOUCH_REV>
        <ITEM_STATE>OK</ITEM_STATE>
        <LEDGER_ID>142</LEDGER_ID>
        <LEDGER_ID_USER>WSS_IFRS</LEDGER_ID_USER>
        <INSTR_GRP>SHORT</INSTR_GRP>
    </Item>
    <Item>
        <DOC_CURR>USD</DOC_CURR>
        <DOC_AMT>-500000.0000</DOC_AMT>
        <BOOK_CURR>EUR</BOOK_CURR>
        <BOOK_AMT>-342000.0000</BOOK_AMT>
        <ERP_ACC>ERP121</ERP_ACC>
        <GLM_ACC>121</GLM_ACC>
        <GLM_ACC_ID>950</GLM_ACC_ID>
        <GLM_ACC_TXT>Deposits Given</GLM_ACC_TXT>
        <ENTRY_ID>3929</ENTRY_ID>
        <VOUCH_ID>DL-2005-004</VOUCH_ID>
        <V_ID>1402</V_ID>
        <VOUCH_TYPE>DL</VOUCH_TYPE>
        <TRAN_NUM>TRM-487</TRAN_NUM>
        <PORTF>WSS_SHORT_DEP_PTF_1</PORTF>
        <INSTR>SHORT_DEPO_EUR</INSTR>
        <CPTY>ABNAMRO</CPTY>
        <CPTY_GRP>EXTERNAL</CPTY_GRP>
        <ISSR>KOBA</ISSR>
        <ISSR_GRP>EXTERNAL</ISSR_GRP>
        <T_OPN_DATE>20050112</T_OPN_DATE>
        <T_MAT_DATE>20050414</T_MAT_DATE>
        <T_VAL_DATE>20050114</T_VAL_DATE>
        <BOOK_DATE>20080229</BOOK_DATE>
        <VAL_DATE>20050414</VAL_DATE>
        <ENTRY_DESC>Principal</ENTRY_DESC>
        <ORIG_ID>TRM-TRANSACTION</ORIG_ID>
        <ORIG_CODE>TRM</ORIG_CODE>
        <DU_ID>15</DU_ID>
        <DI_ID>31</DI_ID>
        <DR_CR>-1</DR_CR>
        <DR_CR_NAME>CR</DR_CR_NAME>
        <D_ID>10</D_ID>
        <ITEM_TYPE>BY_VOUCHER</ITEM_TYPE>
        <UNIT_TYPE>BY_VOUCHER</UNIT_TYPE>
        <VOUCH_REV>NONE</VOUCH_REV>
        <ITEM_STATE>OK</ITEM_STATE>
        <LEDGER_ID>142</LEDGER_ID>
        <LEDGER_ID_USER>WSS_IFRS</LEDGER_ID_USER>
        <INSTR_GRP>SHORT</INSTR_GRP>
    </Item>
</GLM>
```

C.3.3 Example for Account Mode by balance

Below you can see an example of the ERP output file in XML format for the **Account Mode** by balances (movements), i.e. an account movement in the fixed balances granularity is posted against a technical account. Note that only tags, for which a value in ACM exists, are filled.

```
<GLM>
    <Header>
        <ID>3351</ID>
        <OBJ_KEY>KNO-FILE-2002-000004</OBJ_KEY>
        <DOC_DATE>20110310</DOC_DATE>
        <PSTNG_DATE>20020430</PSTNG_DATE>
        <ERP_SYS>FILE</ERP_SYS>
        <ERP_TYPE>FILE</ERP_TYPE>
        <GLM_OWNER>KNO-IFRS-USD</GLM_OWNER>
        <STATE_ID>1210</STATE_ID>
        <EXP_USER>CHARLIE</EXP_USER>
        <DOC_TYPE>3</DOC_TYPE>
        <DOC_FLAGS>0</DOC_FLAGS>
        <LEDGER_ID>100047</LEDGER_ID>
        <ACM_PERIOD>2002-04</ACM_PERIOD>
    </Header>
    <Item>
        <DOC_CURR>EUR</DOC_CURR>
        <DOC_AMT>1008194.4400</DOC_AMT>
        <BOOK_CURR>USD</BOOK_CURR>
        <BOOK_AMT>1245120.1300</BOOK_AMT>
        <GLM_ACC>11-2-50-015-010</GLM_ACC>
        <GLM_ACC_ID>300</GLM_ACC_ID>
        <GLM_ACC_TXT>EUR in Transit</GLM_ACC_TXT>
        <TRAN_NUM>-</TRAN_NUM>
        <VAL_DATE>20020430</VAL_DATE>
        <ENTRY_DESC>Bal. Mv.</ENTRY_DESC>
        <DU_ID>6901</DU_ID>
        <DI_ID>15451</DI_ID>
        <DR_CR>1</DR_CR>
        <DR_CR_NAME>DR</DR_CR_NAME>
        <D_ID>3351</D_ID>
        <ITEM_TYPE>BY_BALANCE</ITEM_TYPE>
        <UNIT_TYPE>BY_BALANCE</UNIT_TYPE>
        <ITEM_STATE>OK</ITEM_STATE>
        <LEDGER_ID>100047</LEDGER_ID>
        <LEDGER_ID_USER>KNO-IFRS-USD</LEDGER_ID_USER>
    </Item>
    <Item>
        <DOC_CURR>EUR</DOC_CURR>
        <DOC_AMT>-1008194.4400</DOC_AMT>
        <BOOK_CURR>USD</BOOK_CURR>
        <BOOK_AMT>-1245120.1300</BOOK_AMT>
        <ERP_ACC>0999888777</ERP_ACC>
        <TRAN_NUM>-</TRAN_NUM>
        <VAL_DATE>20020430</VAL_DATE>
        <ENTRY_DESC>Bal. Mv. Technical Mirror</ENTRY_DESC>
        <DU_ID>6901</DU_ID>
        <DI_ID>15452</DI_ID>
        <DR_CR>-1</DR_CR>
        <DR_CR_NAME>CR</DR_CR_NAME>
        <D_ID>3351</D_ID>
        <ITEM_TYPE>BALANCE_MIRROR</ITEM_TYPE>
        <UNIT_TYPE>BY_BALANCE</UNIT_TYPE>
        <ITEM_STATE>OK</ITEM_STATE>
        <LEDGER_ID>100047</LEDGER_ID>
        <LEDGER_ID_USER>KNO-IFRS-USD</LEDGER_ID_USER>
    </Item>
</GLM>
```

Appendix D ACM environment variables

The following table provides a list of the ACM environment variables.

Environment variable name	Note	Example value / Default Value
ACM_64BIT_MODE	Set to “true” to run ACM server under 64-bit JVM – 64-bit TRM and JVM (1.4.2_10 or higher) is required.	ACM_64BIT_MODE=false
ACM_ASYNC_LOG	Set this to false if you do not want to run the system logging asynchronously.	true
ACM_CMM_INTERFACE_PROXY_URL	CMM web service URL. Must contain a complete URL of the CMM web service.	http://localhost:8081/cmm/iws/acmcmm
ACM_DEFAULT_SCHEMA	The default schema for ACM database objects	**Dbo**
ACM_ERP_BC_HOST	Host name for SAP business connector.	tolar.prg.trema.com
ACM_ERP_BC_PORT	Port number for SAP business connector.	5555
ACM_ERP_BC_USER	SAP business connector user.	Administrator
ACM_ERP_BC_USER_PASS	SAP business connector user password.	manage
ACM_ERP_SAP_CONNECTION	Type of connection to SAP system. Use value XI or BC.	XI
ACM_ERP_XI_DOCUMENT_TYPE	Document Type created on target ERP system (SAP) and used for all documents posted from ACM. For details see configuration guide “Interface WallstreetSystems Suite to mySAP Financials”.	TR
ACM_ERP_XI_HOST	Hostname for SAP XI service.	xi.prg.trema.com
ACM_ERP_XI_HTTP_PROXY_HOST	If you need to use HTTP proxy server for XI communication specify proxy hostname. Use this variable undefined in all other cases.	proxy.prg.trema.com
ACM_ERP_XI_HTTP_PROXY_PORT	If you need to use HTTP proxy server for XI communication specify proxy port. Use this variable undefined in all other cases.	8080
ACM_ERP_XI_OUTPUT_ENCODING	Specify output encoding used for XML documents sent to XI server.	ISO-8859-1
ACM_ERP_XI_OBJECT_TYPE	Object type for Accounting on target ERP system. For details see configuration guide “Interface WallstreetSystems Suite to mySAP Financials”.	ZGLM

Environment variable name	Note	Example value / Default Value
ACM_ERP_XI_PARTNER_PROFILE	Partner profile defined on target ERP system to be used. For details see configuration guide “Interface WallstreetSystems Suite to mySAP Financials”.	XI_TREMA
ACM_ERP_XI_PORT	Port number for SAP XI service.	50080
ACM_ERP_XI_USER	SAP XI user.	xiuser
ACM_ERP_XI_USER_PASS	SAP XI password	xiuser12
ACM_ERP_USE_SAP	Set "true" if your system will post to the SAP R/3 system. (See step 15 in installation steps for the ACM server). If not, set to false or leave it empty.	TRUE
ACM_HTTP_HOST	HTTP server hostname where Tomcat container for ACM Server is running	localhost
ACM_HTTP_PORT	HTTP server port where Tomcat/WebLogic/WebSphere for ACM Server is running	8080
ACM_JDBC_PASS	JDBC database connection user password. To change the password before creating the user during the database build adjust FK_HOME/share/<platform>/acmconfig.pl	**acmdbo12**
ACM_JDBC_URL	JDBC database connection string. The value is derived from the variables already defined.	**Oracle: jdbc:oracle:thin:@%ACM_MODULES_DB_ADDR%:%ORACLE_SID%** **Sybase: jdbc:sybase:Tds:%ACM_MODULES_DB_ADDR%/%FK_DATABASE%?DYNAMIC_PREPARE=true** **MSSQL: jdbc:jtds:sqlserver://%ACM_MODULES_DB_ADDR%/%FK_DATABASE%**
ACM_JDBC_USER	JDBC database connection user name. This user is used to connect to the database. To create the user with a different username adjust FK_HOME/share/<platform>/acmconfig.pl	**acmdbo**
ACM_JVM_BITMODE	Switch for JVM to set proper bit mode. It is automatically evaluated based on ACM_64BIT_MODE.	
ACM_KEYSTORE	Location of the file containing key-pair (public and private key) of the client. It is used for client authentication setup only.	c:\AccountingManager.jks
ACM_KEYSTORE_PASS	Password of the keystore. It has to match with the key-pair password. It is used for client authentication setup only.	AccountingManager12

Environment variable name	Note	Example value / Default Value
ACM_LIBRARY_PATH	Path where ACM searches for native libraries	**ACM_LIBRARY_PATH=%FK_HOME%\bin**
ACM_LOG_MAIL_FROM	Sender address to send email messages from.	acm_server@trema.com
ACM_LOG_MAIL_IGNORE	Specifies whether the error mails will be sent to the address specified in ACM_LOG_MAIL_TO variable. If you want to suppress sending the error mails set this variable value to "true".	**false**
ACM_LOG_MAIL_MAX_MAILS_PER_DAY	The maximum number of email messages sent per day or following the ACM Server startup. The counter gets reset to 0 after 24 hours or after ACM Server startup. The default is 40.	**40**
ACM_LOG_MAIL_SMTP_HOST	SMTP server hostname to be used to send email messages via Log4j.	labex2.corp.trema.com
ACM_LOG_MAIL_SUBJECT	The email subject text.	%FK_IDENT% ACM error (%FK_RDBMS%, %FK_DATABASE%)
ACM_LOG_MAIL_TO	Recipient address to send the email messages to.	_ACM-build@trema.com
ACM_MODULES_DB_ADDR	Database server location Oracle - <hostname>:<port> Sybase+MSSQL - <servername>:<port>	Oracle: test1.corp.trema.com:1521 Sybase: SYBASE_LABS:4100 MSSQL: DUKAT:1433
ACM_NAMING_HOST	The hostname of the computer where naming service is running - usually set via MODULES_NAMING_HOST (if set in the master TRM environment file)	**%MODULES_NAMING_HOST%**
ACM_NAMING_PORT	The port number where the naming service is listening - usually set via MODULES_NAMING_PORT (if set in the master TRM environment file)	**%MODULES_NAMING_PORT%**
ACM_OPENORB_CONFIG	Path to the OpenORB configuration file	**%ACM_HOME%\etc\acm\OpenORB.xml**
ACM_PROTOCOL	Protocol used for client to server communication (http/https). It is only needed when https is the wanted protocol.	http
ACM_REPORT_DEFINITIONS_BASE_DIR		
ACM_REPORT_DEFINITIONS_DIR	For easier customization of report definitions (xml files) you can override a default setting and force the ACM server load it from an external location specified by this variable.	
ACM_STARTUP_WORKFLOW	The internal components configuration - default is "ACM-STARTUP" which includes all components.	**ACM-STARTUP**

Environment variable name	Note	Example value / Default Value
ACM_TOMCAT_BASE	The first property ACM_TOMCAT_HOME points to the location of the common information, while the other property ACM_TOMCAT_BASE points to the directory where all the instance specific information are held. For the single instance installation these two values equal.	%ACM_TOMCAT_HOME%
ACM_TOMCAT_HOME	ACM_TOMCAT_HOME must point to the home directory of Tomcat installation. This property must be set in order to install ACM into a container	%TOMCAT_HOME%
ACM_TRUSTSTORE	Location of the file containing server public key (if self-signed) or the complete chain of issuing authorities. It is used only if the https protocol is choosen.	c:\truststore.jks
ACM_WEB_CONTEXT	The web context that is used during the ACM Server installation and also during ACM Entry Manager communication with ACM Server, default is "acm"	**acm**
PATH	Add some extra paths to the PATH system variable, namely a path to the ACM Entry Manager executable and also to the java executable (using TRM's JRE)	**%FK_HOME%\jre\bin; %PATH%; %FK_HOME%\share\java\acm\client**

The values in bold are the default values – these are the values used in the acm_config.pl (Unix) or acm_config.bat (Windows) configuration file.

Appendix E Secure Socket Layer (SSL)

ACM can be configured to communicate via a secured connection. It uses the secure socket layer (SSL) standard. This chapter describes basic information about this topic:

SSL provides the following security advantages:

- Ciphers the communication.
- Verifies the client communicates with a right server.
- Verifies the client is allowed to communicate with the server (so called client authentication).

E.1 Basic SSL terms

The following are basic SSL terms:

Certificate

Certificate is a digitally signed statement from one entity (the issuer), saying that the public key (and some other information) of another entity (the subject) has some specific value.

Certification Authority (CA)

Company that issues the certificates. It is the one you can trust. See details below.

http://en.wikipedia.org/wiki/Certificate_authority

Keystore

Keystore is a file storing key entries — pair of public and private key. In ACM context all the keystores are Java keystores (JKS) no matter what the file extention is.

Trustore

Truststore is a keystore that contains certificates that you can trust. It may be the same file as a keystore.

Keytool

keytool.exe is a Java tool (to be found in the J2SDK distribution) that is used to handle keystores and the certificates stored there. It allows certificates to be imported, exported etc. See complete documentation (follow the link below).

OpenSSL

A set of tools to handle certificates similar way as keytool does. It is widely used to provide SSL functionality. In our case the Tomcat server can use this library to manage certificates.

Detailed information about all the terms above can be found on the following addresses.

http://java.sun.com/j2se/1.4.2/docs/tooldocs/solaris/keytool.html

http://java.sun.com/j2se/1.4.2/docs/tooldocs/windows/keytool.html

E.2 System setup

There are two security levels using SSL.

1. **Only server provides its certificate.**

   It’s a simpler way that ensures the server is trusted plus the communication is ciphered.

   To enable this security, the server has to set up its keystore having a key-pair (public and private key of the server) and the keystore password. The password has to be the same as the key pair password stored in the keystore.

   On the client there has to be specified the truststore location with the server public key or the complete certificate chain that issues the server public key.

2. **Both server and client have their certificates** to provide them to the other side to ensure that the server is trusted and the client is allowed to connect to the server.

   To enable this functionality, on the server the administrator needs to specify the keystore location (and keystore password) with the key-pair and the truststore location with the client public key or the client issuing CA public key.

   There are two ways how to get a certificate — create the own one (so called self-signed) or let the CA to create one for you. For our purposes it's enough to have a self-signed certificate — there is absolutelly no need to have a CA signed one. It causes no security issues in case of ACM securing.

See the keytool documentation for details on how to generate the certificates/keystores.

Appendix F TRM FX conversion

TRM Daily Events Generation activity consists from the following basic steps:

a. Average price selling method execution

b. FIFO selling method execution

c. FX conversion (i.e. FX results calculations)

d. Processing of TRM accounting inputs (RUNNING and OFF-BALANCE) and the appropriate events generation.

This appendix provides some details about the FX conversion step.

The FX Conversion process is performed automatically before the daily accounting events are generated. The setup is stored in the FXConversionRule table in the TRM database. The FX conversion process is rule driven and the standard TRM rule logic applies. The TRM cashflows are compared with FX Conversion Rules in order of their priorities. If Rule matches and Not Rule does not match, the stored procedure specified in the conversion_proc field is executed.

Key cashflow rates affected by the FX conversion rules setup are:

– cashflow.fx_rate

  Aka **Base FX Rate** in the Transaction Manager, Cashflow view

– cashflow.fx_book_rate

  Aka **Base Book FX Rate** in the Transaction Manager, Cashflow view

– cashflow.fx_trade_rate

  Aka **Payment Trade FX Rate** in the Transaction Manager, Cashflow view

– cashflow.fx_value_rate

  Aka **Payment Value FX Rate** in the Transaction Manager, Cashflow view

These rates are then used for the conversion of document amounts to booking amounts and for the calculation of FX results.

In general (since there are a few exceptions), cashflow.fx_book_rate is used for creating a CASH event, while cashflow.fx_rate is used for creating a NON-CASH event.

Here is a description of the FXConversion table:

Field Name	Field Description
id	ID of the FX Conversion Rule
description	Short name
priority	Priority of the FX Conversion Rule
contexts	Result contexts for which the rule applies. -1 represents all the contexts.

Field Name	Field Description
date_type_id	Cashflow date type: **1** - Cashflow **Active From** date (Cashflow.active_since) Cashflow.active_since date is usually set to Opening Date of the Transaction **2** - Cashflow **Value Date** (Cashflow.value_date) **3** - Cashflow **Payment Date** (Cashflow.payment_date)… in Transaction Admin only displayed if different from Value Date **4** – Cashflow **Effective Date** (Cashflow.realize_date) **5** – Cashflow **From When** date (Cashflow.since_when) Is usually the Value Date of the Transaction
rule_ID not_rule_id	Rules and Not Rules are defined in Rule Editor.
counter	
conversion_proc	FX Conversion Procedure. In the description below @fx_rate denotes the actual FX rate (for realized_portfolio specified per user in Accounting Configuration Editor). The following procedures are available: **FXSetTradeRate** – updates Payment Trade FX Rate of a cashflow visible in the Transaction Admin (Cashflow view) with the actual FX rate: update Cashflow set fx_trade_rate = @fx_rate where id = @id and fx_value_rate != @fx_rate **FXSetValueRate** – updates Payment Value FX Rate of a cashflow visible in the Transaction Admin (Cashflow view) with the actual FX rate: update Cashflow set fx_value_rate = @fx_rate where id = @id and fx_value_rate != @fx_rate **FXSetBookRate** – updates Base Book FX Rate of a cashflow visible in the Transaction Admin (Cashflow view) with the actual FX rate: update Cashflow set fx_book_rate = @fx_rate where id = @id and fx_book_rate != @fx_rate **FXSetRate** – updates Base FX Rate of a cashflow visible in the Transaction Admin (Cashflow view) with the actual FX rate: update Cashflow set fx_rate = @fx_rate where id = @id and fx_rate != @fx_rate **FXSetRatesAvgBalance** - updates all rates on average transactions if @active_since >< @since_when: update Cashflow set fx_rate = @fx_rate, fx_book_rate = @fx_rate, fx_value_rate = @fx_rate, base_amount = @base_amount where id = @id

Field Name	Field Description

	FXSetReferenceRate – updates Base FX Rate and Base Amount of a cashflow visible in the Transaction Admin (Cashflow view) using the actual FX rate:
	`update Cashflow`
	`set fx_rate = @fx_rate,`
	`base_amount = @base_amount`
	`where id = @id`
	`and (fx_rate != @fx_rate`
	`or base_amount != @base_amount)`
	where @base_amount is amount converted to base currency using @fx_rate:
	`exec ConvertAmount`
	`@from_id = @currency_id,`
	`@to_id = @base_currency_id,`
	`@from_amount = @amount,`
	`@to_amount = @base_amount output,`
	`@fx_rate = @fx_rate`
	Note: This procedure also updates Base FX Rate of all related future interest cashflowsl.
	Note: This procedure does nothing for Principal cashflow of the sell transaction in case of trade date selling.
	FXDoRealize – creates (or updates if already exists) FX Profit cashflow. FX Profit cashflow is calculated as the difference between the 'old' cashflow Base Amount and the 'new' cashflow Base Amount (Amount converted to base currency using actual FX rate):
	`@profit = @new_base_amount - @old_base_amount`
	where @new_base_amount is amount converted to base currency using @fx_rate:
	`exec ConvertAmount`
	` @from_id = @currency_id,`
	` @to_id = @base_currency_id,`
	` @from_amount = @amount,`
	` @to_amount = @new_base_amount output,`
	` @fx_rate = @fx_rate`
	FXDoRealizeFX – works the same way as FXDoRealize procedure, but there are two profit cashflows generated: FX Profit cashflow and IR Profit cashflow. Calculation methods of both profit cashflows (and Cashflow Type of the IR profit cashflow) depend on the Profit Method selected for the instrument (Result FX page of Result Template or Instrument Editors).
	FXDoRealizeVD.

Flags	1 – Stop. If conversion rule is marked with this flag then the processing (for the actual cashflow) stops after the 'stop' rule is processed.

The standard Wallstreet Suite installation contains pre-configured FX Conversion Rules (table FXConversionRule). They refer to some default Rules and Not Rules.

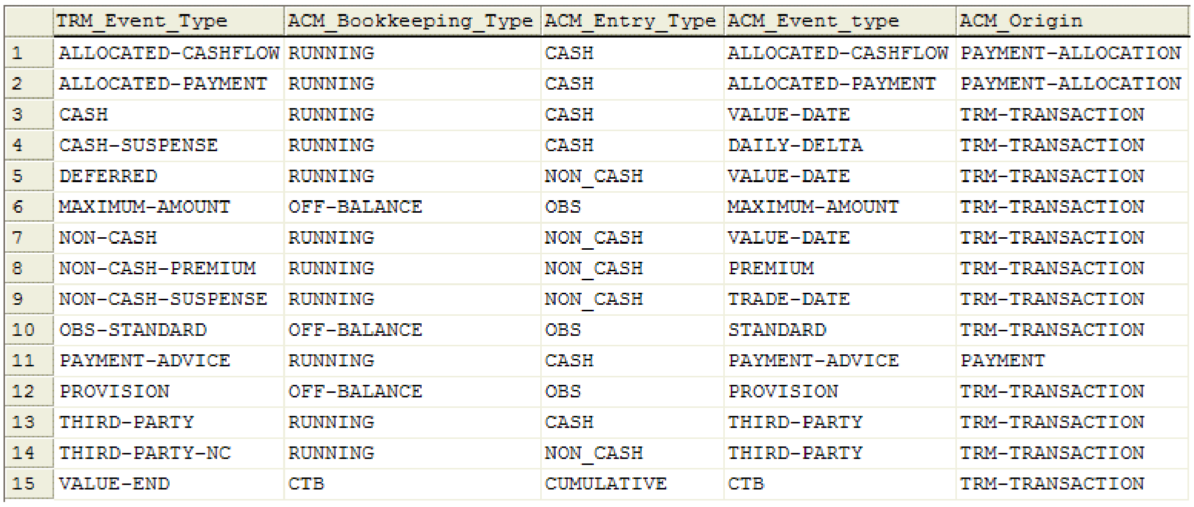
Warning: **All these system rules cannot be either deleted or modified. You should not even add any new rule! Any such change is considered as a customer specific modification will all possible subsequent impacts.** It is always a user's responsibility to modify the rules when any change of the default rules set-up comes with new patch/version. When doing any customization related the FX conversion rules, think about both realized and unrealized figures, specify how selling, buy-back, early expiration, option exercise, roll-over and other actions should work for you, what FX rates should be used for what action, etc.

Appendix G Event Type mapping to ACMEntry

Event types available at acounting event level are mapped to four fields in ACM entry. They are:

- **Bookkeeping Type** (table ACMBookkeepingType),
- **Entry Type** (table ACMEntryType),
- **Event Type** (table ACMEventType),
- **Origin Entity** (table ACMEntryOriginEntity).

This mapping is defined in the ACMEventTypeMap table. Its default content is shown below:



	TRM_Event_Type	ACM_Bookkeeping_Type	ACM_Entry_Type	ACM_Event_type	ACM_Origin
1	ALLOCATED-CASHFLOW	RUNNING	CASH	ALLOCATED-CASHFLOW	PAYMENT-ALLOCATION
2	ALLOCATED-PAYMENT	RUNNING	CASH	ALLOCATED-PAYMENT	PAYMENT-ALLOCATION
3	CASH	RUNNING	CASH	VALUE-DATE	TRM-TRANSACTION
4	CASH-SUSPENSE	RUNNING	CASH	DAILY-DELTA	TRM-TRANSACTION
5	DEFERRED	RUNNING	NON_CASH	VALUE-DATE	TRM-TRANSACTION
6	MAXIMUM-AMOUNT	OFF-BALANCE	OBS	MAXIMUM-AMOUNT	TRM-TRANSACTION
7	NON-CASH	RUNNING	NON_CASH	VALUE-DATE	TRM-TRANSACTION
8	NON-CASH-PREMIUM	RUNNING	NON_CASH	PREMIUM	TRM-TRANSACTION
9	NON-CASH-SUSPENSE	RUNNING	NON_CASH	TRADE-DATE	TRM-TRANSACTION
10	OBS-STANDARD	OFF-BALANCE	OBS	STANDARD	TRM-TRANSACTION
11	PAYMENT-ADVICE	RUNNING	CASH	PAYMENT-ADVICE	PAYMENT
12	PROVISION	OFF-BALANCE	OBS	PROVISION	TRM-TRANSACTION
13	THIRD-PARTY	RUNNING	CASH	THIRD-PARTY	TRM-TRANSACTION
14	THIRD-PARTY-NC	RUNNING	NON_CASH	THIRD-PARTY	TRM-TRANSACTION
15	VALUE-END	CTB	CUMULATIVE	CTB	TRM-TRANSACTION

The real ACMEventTypeMap table uses just system IDs referencing to the appropriate tables. Those system IDs were replaced by user IDs for better comprehension in the picture above.