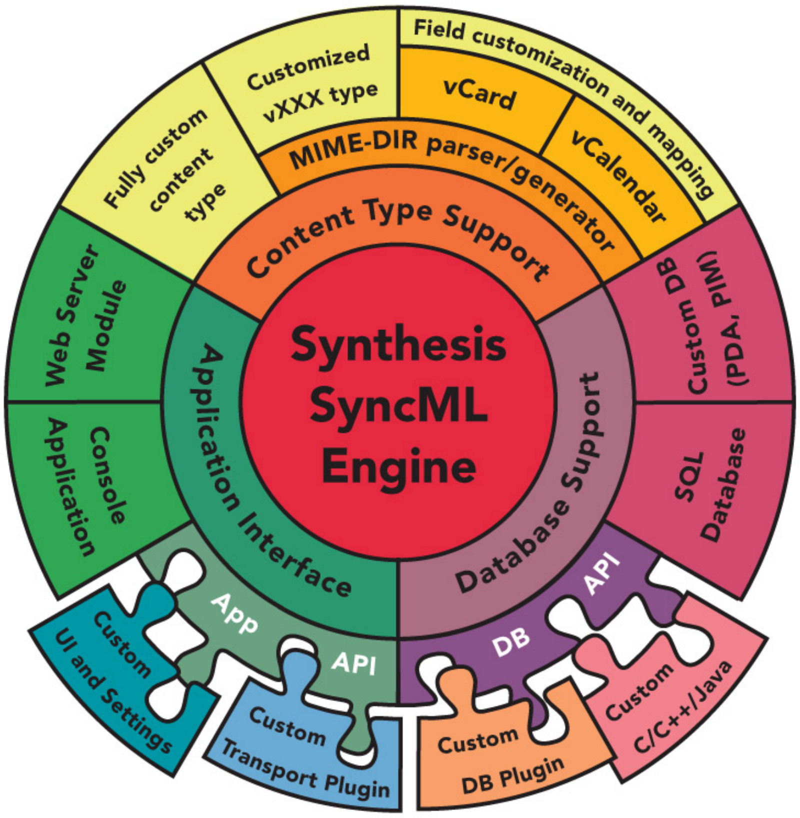
Sustainable Software Concepts

SYNTHESIS AG

www.synthesis.ch

Installation Manual for Synthesis SyncML Client 3.2



This manual was written for Synthesis SyncML Client Version 3.2.0.11

Synthesis Sync Client uses parts of the following software:

expat - XML parser

(see http://expat.sourceforge.net/)

SyncML toolkit

(see http://www.syncml.org and http://sourceforge.net/projects/syncml-ctoolkit/)

Disclaimer

1. Introduction

Thank you for choosing Synthesis Sync Client as your SyncML client solution. It provides you with a very efficient compliant SyncML client with many advanced features and especially a high configurability.

If you have used earlier versions of the Synthesis SyncML client, please refer to Chapter 2 for a quick overview of the changes.

Synthesis Sync Client exists in different versions for different database interfaces. This manual covers the SQL/ODBC with custom database adaptor plugin version of Synthesis Sync Client.

This manual covers installation based on the sample configuration provided by Synthesis. Please refer to the **Synthesis SyncML Config Reference** document for details about all the configuration options.

Synthesis Sync Client – SQL/Plugin Version is designed to bring SyncML synchronisation functionality to almost any SQL-accessible database using the ODBC or SQLite3 interface directly.
With the help of custom database adaptor plugins that can be developed using our SDK in C/C++ or Java, any application or database backend can be SyncML enabled using Synthesis SyncML Client (PRO version).

To allow using existing database tables, the Synthesis Sync Client is highly configurable. With its XML-based configuration file, the mapping between SyncML data formats (normally vCard and vCalendar) and database tables can be defined on a field by field basis.

As this customized configuration to a specific database layout is not trivial and needs some time to successfully be completed and tested, Synthesis Sync Client comes with sample files for creating a standard layout. This allows you to set-up a working SyncML solution in a short time. It also provides a starting point for customizing the client to your specific needs.

Therefore, the setup guide in this manual describes how to use the standard layout. If you are new to Synthesis Sync Client, we recommend that you first install and test a standard layout before doing your own configuration.

Synthesis Sync Client - DEMO Version is a Demo version. It is available for free and provides full SyncML functionality, but stores data in TAB-separated text files or SQLite3 databases. While this version is not indended (nor suitable) for productive use, it is installed in seconds and very useful to test SyncML servers or get an insight into Synthesis Sync engine's configuration capabilities.
See section 5.2 for how to install this version quickly.

Contents

2. Changes between Version 3.0 and 3.2

Version 3.2 is an update of the Synthesis SyncML engine which affects all products. So please refer to paragraph "New in SyncML Engine 3.2 compared to 3.0" in the configuration reference manual (SySync_config_reference.pdf).

Basically, existing 3.0 installations can be run with the new 3.2 server with no change to the database layout and minor or even no updates (depending on the existing configuration) in the configuration file. Instructions on how to update the configuration can be found in paragraph "How to migrate from 3.0 to 3.2" in the configuration reference manual (SySync_config_reference.pdf).

3. Changes between Version 2.1 and 3.x

This paragraph is indended for existing users of the previous 2.1 version of the Synthesis SyncML server. It lists the most important differences between the two versions.

Basically, SyncML Server 3.0 (and 2.9.x) is upwards compatible with existing 2.1.x configurations. This means that most configuration files written for the 2.1.x release will also work with version 3.0 and higher. There might be cases where minor adaptions are required.

Please note however, that the new sample database layouts provided for MySQL, MS-SQL and Interbase with this 3.0 server release are **not fully compatible with the old 2.1 sample layouts** (because they were update to include new features and datatypes).

3.1 Feature Matrix - 2.1 vs 3.x and STD vs PRO

The following matrix shows the most important changes between 2.1 and the new 3.x versions and also shows the difference between the STD and PRO versions. Note that this matrix is not intended to be complete (it would get too long), it just lists the most important new features.

Feature Description	**STD**		**PRO**	
	2.1	**3.x**	**2.1**	**3.x**
SyncML DS 1.2 / OMA DS 1.2: Supports SyncML 1.2 standard (but automatically falls back to 1.1 or 1.0 if server does not support 1.2)		➜		➜
Suspend & Resume: Full support for new SyncML DS 1.2 Suspend & Resume feature – if a sync session is interrupted intentionally or due to external conditions like network coverage loss etc., the sync will just continue at the point it was interrupted, and not restarted from beginning. Even if sync stops while transferring a large item, the part of the item that was already sucessfully transmitted will not be sent again. This greatly enhances end user experience and reduces data volume, especially for large syncs and instable mobile data connections. **Note that Suspend & Resume support needs some adaptations in the database layout and the config file for existing 2.1.x client. See Migration Guide in 3.4.**		➜		➜
SyncML DS 1.2 filtering: Support for sending <filter>				➜

expressions to servers that support it.				
ODBC Datastores: Make data from SQL/ODBC databases available for SyncML.	➜	➜	➜	➜
Plugin Datastores: Develop custom database adapters with our Plugin SDK in C/C++ or Java to communicate with any custom application layer. Can be mixed in the same server with ODBC based datastores. Administrative data can be managed in ODBC/SQL or plugins independently from how the user data is handled. All combinations possible.			➜	➜
Built-in Textfile based plugin:A plugin for textfile based storage of administrative data and/or user data is built-in and can be used in any mix with ODBC and custom plugins.		➜		➜
Resend failed items: Instead of aborting sync when a client returns an error for a Add/Replace or Delete operation, the item can now be marked for resend in the next session. This helps a lot to overcome temporary problems in client databases without interrupting sync.		➜		➜
Hierachically structured HTML logs: Much easier to read debug logs, as nicely colored (e.g. error conditions in bright red to immediately catch the eye) and dynamically folding HTML, as XML or structured plain text. Much faster log writing, better detail control.		➜		➜
Support for BLOB data (PHOTO, email attachments): Store BLOB (binary large object) data in the database and map it to base 64 encoded binary properties like PHOTO in vCard or email attachments.		➜		➜
On-request delayed BLOB reading for high efficiency: To optimize efficiency, large BLOB fields can be fetched from the database separately at the time they are required for transferring to a remote party. This allows efficient handling of large BLOBs such as email attachments.				➜
Scripting: Gain much more flexibility in adapting SyncML to your existing database - create fully custom value conversions and much more using the new built-in, highly efficient C/JavaScript-Style scripting language.			➜	➜
Macros in Scripts: Wrap script parts into a macro and use it in multiple scripts across the configuration.				➜
Email with Attachments: Use the built-in support for RFC822/2822/MIME email encoding and decoding, including multipart bodies and attachments.				➜
Support for vBookmark: The new sample config contains support for vBookmark which is supported by some newer client devices.		➜		➜
Array Fields: For repeating items (such as storing an unlimited number of telephone numbers per contact) the server now supports dynamic array fields that can hold as many values as required.				➜
Master-Detail database Table support: For storing array contents or other data not in the main (master) record but in a related (detail) table, the server provides powerful mapping options.				➜
More than 30 (in 3.2: 50) new configuration tags: Many detail		➜		➜

enhancements and extended configuration options.				
More than 10 (in 3.2: 40) new script functions: Many new built-in functions for more scripting capabilities.				➽

3.2 Changes for end-users (clients) of the SyncML client

- Errors storing or modifying items in the client will no longer abort the sync. Instead, the same item will be retried in the next session. This enhances overall experience also with older (pre SyncML 1.2) servers, as a single problematic item in a sync will no longer make the entire sync fail.
- When using the new 3.0 sample layout / config, the client now supports email sync and vBookmark sync with devices that support these data types.

3.3 Changes in the new 3.0 sample layout (ODBC version):

- SYNC_TARGETS and SYNC_MAPS have a number of new fields to support Suspend & Resume (see 3.4.2 for details).
- SYNC_LOGS has some additional fields for extra information such as number of errors.
- The DEVICEID and DEVICEINFO fields in SYNC_DEVICES are now longer to make sure even very long device IDs don't cause problems.
- New tables SYNC_EMAILS, SYNC_EMAIL_ATTS and SYNC_BOOKMARKS have been added to support email and bookmarks.
- Data tables for contacts, events and tasks (SYNC_CONTACTS, SYNC_EVENTS, SYNC_TASKS) have been enhanced to support some more fields like contacts picture, a third postal address, categories, classification etc.

3.4 Migration Guide – How to upgrade existing 2.1.x installations to 3.0

3.4.1 Overview

What needs to be done to upgrade an existing 2.1.x based installation to 3.0?
There are three possibilities:

1. **Just use the new client version with your existing 2.1 config and database**. This will bring you basic SyncML DS 1.2 compatibility with almost no effort. However, a very important advantage of SyncML DS 1.2, the **Suspend & Resume feature, will not work** this way. You will also get some warnings about <linemap>s that should be moved into <textprofile> when starting up the client or testing the config syntax.
   This is not recommended as a long term solution, but can be done as a first step towards SyncML DS 1.2 support.

2. **Apply minimal updates to enable full SyncML DS 1.2 Suspend & Resume.** This requires some changes in the config file and some new fields in the admin tables in the databases, but **does not touch anything related to your data tables, so it requires no changes in your application.** This is the recommended migration and is explained step by step below (see 3.4.2).

3. **Enhance your existing configuration with new features from the new 3.0 sample.** The new 3.0 client supports new things that were not possible with the 2.1. For example, support for pictures in contacts is now easily possible and also part of the new 3.0 config sample. If your application supports contact pictures, you should adapt your 2.1 config to support them with your application. The new sample config now also contains sample support for an email datastore and vBookmark.

3.4.2 Updating a 2.1 installation for 3.0 with full Suspend & Resume support

The following is a step by step guide how to update an existing Synthesis SyncML Client 2.1 installation to version 3.0 to gain full SyncML DS 1.2 Suspend & Resume support.

1. Backup your database and config (of course). Or work on a copy of the current database.

2. For suspend and resume to work, the SYNC_TARGETS and SYNC_MAPS tables need additional fields. In the *sample_config* directory of the 3.0 client distribution, there is a SQL script for each supported database called *XXXX _upgrade_2.1_to_3.0.sql* (XXX = database system). This script contains the necessary ALTER TABLE statements to add the required fields to SYNC_TARGETS and SYNC_MAPS.
   In addition, the size of the deviceID in SYNC_DEVICES is increased and two new fields are added to SYNC_LOGS. These two changes are not absolutely needed, but strongly recommended.
   Execute this script on your sync database using your favorite SQL tool to apply the changes. You can also apply them manually in a GUI table editor.

   To summarize: The new fields are:

   - **In SYNC_TARGETS:**

LASTSUSPEND	DATETIME	
RESUMEALERT	INTEGER	16 bits required
LISOURCE	VARCHAR(63)	
LITARGET	VARCHAR(63)	
LISTATUS	INTEGER	16 bits required
PISTATE	INTEGER	8 bits required
PITOTALSZ	INTEGER	32 bits required
PIUNCONFSZ	INTEGER	32 bits required
PISTOREDSZ	INTEGER	32 bits required
PIDATA	BLOB	(eventually called IMAGE or LARGEBLOB)

   - **In SYNC_MAPS:**

ENTRYTYPE	INTEGER	8 bits required – **Part of Primary Key!!!**
FLAGS	INTEGER	32 bits required

     Note that it is essential that ENTRYTYPE becomes part of the primary key of the map table (in addition to LOCALID and TARGETKEY).

   - **In SYNC_LOGS:**

LOC_ERRORS	INTEGER
REM_ERRORS	INTEGER

3. In your XML config file, immediately Before each occurrence of **<synctargetgetsql>** (there is one for each <datastore>), add the following two new tags (add the **bold red part**) - these enable Suspend and Resume functionality, for details see Config Reference Manual:

```
<resumesupport>yes</resumesupport>
<resumeitemsupport>yes</resumeitemsupport>

<synctargetsql>
...
```

4. Change all **<synctargetgetsql>** (there is one for each <datastore>, differing only in the *DSCODE="xx"* part – but that's essential, make sure you leave that "xx" as it is for each datastore!) in your XML config file as follows (add the **bold red part**):

```
SELECT TARGET_KEY, ANCHOR, LASTSYNC, LASTTOREMOTESYNC,
RESUMEALERT, LASTSUSPEND, LISOURCE, LITARGET, LISTATUS,
PISTATE, PITOTALSZ, PIUNCONFSZ, PISTOREDSZ, PIDATA FROM
SYNC_TARGETS WHERE DSCODE='xx' AND USERKEY=%u AND
FOLDERKEY=%f AND DEVICEKEY=%d AND DEVICEDBPATH='%P'
```

5. Change all **<synctargetupdatesql>** (one for each <datastore>, all identical) as follows: (add the **bold red part**):

```
UPDATE SYNC_CONTACTS_TARGETS SET ANCHOR='%A',
LASTSYNC=%L WHERE TARGET_KEY='%t'
UPDATE SYNC_TARGETS SET ANCHOR='%A', LASTSYNC=%L,
LASTTOREMOTESYNC=%RL, RESUMEALERT=%SUA,
LASTSUSPEND=%SU, LISOURCE='%pSU', LITARGET='%pTU',
LISTATUS=%pSt, PISTATE=%pM, PITOTALSZ=%pTS,
PIUNCONFSZ=%pUS, PISTOREDSZ=%pSS, PIDATA=%pDAT WHERE
TARGET_KEY=%t
```

6. Change all **<selectmapallsql>** (one for each <datastore>, all identical) as follows: (add the **bold red part**):

```
SELECT LOCALID, REMOTEID, ENTRYTYPE, FLAGS FROM
SYNC_MAPS WHERE TARGETKEY=%t
```

7. Change all **<insertmapsql>** (one for each <datastore>, all identical) as follows: (add the **bold red part**):

```
INSERT INTO SYNC_MAPS (LOCALID, REMOTEID, TARGETKEY,
ENTRYTYPE, FLAGS) VALUES (%k,'%r',%t, %e, %x)
```

8. Change all **<updatemapsql>** (one for each <datastore>, all identical) as follows: (add the **bold red part**):

```
UPDATE SYNC_MAPS SET REMOTEID='%r', FLAGS=%x WHERE
LOCALID=%k AND ENTRYTYPE=%e AND TARGETKEY=%t
```

9. Change all **<deletemapsql>** (one for each <datastore>, all identical) as follows: (add the **bold red part**):

```
DELETE FROM SYNC_MAPS WHERE LOCALID=%k AND ENTRYTYPE=%e
AND TARGETKEY=%t
```

With these steps, your installation is now enabled for SyncML DS 1.2 suspend and resume.

Still, for making full use of the numerous improvements and new features in the 3.0 client, have a look at the new 3.0 sample config and maybe use a difference viewer to compare your config to the new 3.0 – this way you'll probably see other changes that might be advantageous for your installation. The config reference manual explains all config features in detail.

4. Distribution Files

4.1 ODBC Version - Windows

The distribution media (normally a .ZIP archive) contains the following files (note that depending on the version you have, not all of the listed files will be included):

```
SYNTHESIS SYNC CLIENT
├───docs
│       *.pdf    (documentation in Adobe PDF format)
├───bin_win32
│       syncclient_odbc.exe           (PRO version)
│       syncclient_odbc_std.exe       (STD version)
│
├───sample_config
│   ├───interbase_firebird
│   │       interbase.sql
│   │       interbase_upgrade_2.1_to_3.0.sql
│   │       syncclient_odbc.xml       (for PRO version)
│   │       syncclient_odbc_std.xml   (for STD version)
│   │
│   ├───mssql
│   │       mssql.sql
│   │       mssql_upgrade_2.1_to_3.0.sql
│   │       syncclient_odbc.xml       (for PRO version)
│   │       syncclient_odbc_std.xml   (for STD version)
│   │
│   └───mysql
│           mysql.sql
│           mysql_upgrade_2.1_to_3.0.sql
│           syncclient_odbc.xml       (for PRO version)
│           syncclient_odbc_std.xml   (for STD version)
│
└───unsupported
        admin.php
        config.php
        index.php
        read_me.txt
        logo.jpg
```

'docs' Directory

- 'SySync_Client_manual.pdf' : This file
- 'SySync_config_reference.pdf' : Detailed description of XML configuration options

- Eventually some extra materials such as Synthesis SyncML product overviews etc.

'bin_win32' Directory

- 'syncclient_odbcXXX.exe' where XXX is "_std" for STD: This is the console-based, standalone version of the client.

'sample_config' Directory

- 'interbase_firebird' : This directory contains a SQL set-up script for Interbase 6 or Firebird 1 SQL servers and sample config files for the Synthesis Sync Client. Note that you need to use the config files ending with "_std" when you are using the STANDARD version.
- 'mysql' : This directory contains a SQL set-up script for MySQL server and sample config files. Note that you need to use the config files ending with "_std" when you are using the STANDARD version. **Please note that this sample is primarily intended for use with Linux and therefore sample file paths are in UNIX format** (however you can just change them to appropriate Windows paths)
- 'mssql' : This directory contains a SQL set-up script for MS SQL server and sample config files. Note that you need to use the config files ending with "_std" when you are using the STANDARD version.

'unsupported' Directory

- 'config.php', 'admin.php', 'index.php' and 'logo.jpg': These files provide a simple web-based interface to access the data in the standard database layouts. You need to have PHP4 installed on the web server which hosts these files (see. www.php.net). **THESE FILES ARE PROVIDED AS IS AS A FREE ADD-ON FOR TESTING PURPOSES. SYNTHESIS AG DOES NOT PROVIDE ANY SUPPORT FOR THESE FILES.**
- 'read-me.txt' : Short explanation how to use the PHP files.

4.2 ODBC Version - Linux

The distribution media (normally a gzipped tar archive - .tgz) contains the following files (note that depending on the version you have, not all of the listed files will be included):

```
SYNTHESIS SYNC CLIENT
├───some read-me, license, disclaimer text files
├───docs
│       *.pdf    (documentation in Adobe PDF format)
├───bin_linux_x86
│       syncclient_odbc                  (PRO version)
│       syncclient_odbc_std              (STD version)
│
├───sample_config
│   ├───interbase_firebird
│   │       interbase.sql
│   │       interbase_upgrade_2.1_to_3.0.sql
│   │       syncclient_odbc.xml          (for PRO version)
│   │       syncclient_odbc_std.xml      (for STD version)
│   │
│   ├───mssql
│   │       mssql.sql
│   │       mssql_upgrade_2.1_to_3.0.sql
│   │       syncclient_odbc.xml          (for PRO version)
│   │       syncclient_odbc_std.xml      (for STD version)
│   │
│   └───mysql
│           mysql.sql
│           mysql_upgrade_2.1_to_3.0.sql
│           syncclient_odbc.xml          (for PRO version)
│           syncclient_odbc_std.xml      (for STD version)
│
└───unsupported
        admin.php
        config.php
        index.php
        read_me.txt
        logo.jpg
```

distribution directory root

- Read-me, disclaimer, license text files depending on version.

'docs' Directory

- 'SySync_Client_manual.pdf' : This file

- 'SySync_config_reference.pdf' : Detailed description of XML configuration options
- Eventually some extra materials such as Synthesis SyncML product overviews etc.

'bin_linux_x86' Directory

- 'syncclient_odbc' for the PRO and 'syncclient_odbc_std" for STD version: This is the console-based, standalone version of the client.

'sample_config' Directory

- 'mysql' : This directory contains a SQL set-up script for MySQL server and sample config files. Note that you need to use the config files ending with "_std" when you are using the STANDARD version.
- 'interbase_firebird' : This directory contains a SQL set-up script for Interbase 6 or Firebird 1 SQL servers and sample config files for the Synthesis Sync Client. Note that you need to use the config files ending with "_std" when you are using the STANDARD version. **Please note that this sample is primarily intended for use with Windows and therefore sample file paths are in Windows format** (however you can just change them to appropriate Linux paths)
- 'mssql' : This directory contains a SQL set-up script for MS SQL server and sample config files. Note that you need to use the config files ending with "_std" when you are using the STANDARD version. **Please note that this sample is primarily intended for use with Windows and therefore sample file paths are in Windows format** (however you can just change them to appropriate Linux paths)

'unsupported' Directory

- 'config.php', 'admin.php', 'index.php' and 'logo.jpg': These files provide a simple web-based interface to access the data in the standard database layouts. You need to have PHP4 installed on the web server which hosts these files (see. www.php.net). **THESE FILES ARE PROVIDED AS IS AS A FREE ADD-ON FOR TESTING PURPOSES. SYNTHESIS AG DOES NOT PROVIDE ANY SUPPORT FOR THESE FILES.**
- 'read-me.txt' : Short explanation how to use the PHP files.

4.3 Demo (text file based) Version

The distribution media (normally a .ZIP archive) contains the following files:

- 'SySync_Client_manual.pdf' : This file
- 'SySync_config_reference.pdf': This extra manual contains detailed information for writing config files.
- 'syncclient_demo.exe' : This is the demo sync client. It is a console-based, standalone version of the client using text files for storing data.
- 'syncclient_demo' : For MacOSX and Linux, the executable file has no extension.
- 'syncclient_demo.xml' : Sample config file. This file contains everything you need to start up the client.
- Eventually some extra materials such as Synthesis SyncML product overviews etc.

5. Setup Guide

This setup guide consists of two main sections:

- Section 5.2 describes how to install the free textfile-based demo version.
- Section 5.1 describes the more complex set-up for the ODBC versions which store data in an SQL database. The setup guide contains step-by-step instructions to install one of the standard configurations (MS-SQL, MySQL or Interbase/Firebird).

5.1 ODBC Version with standard layouts

5.1.1 Database preparation

5.1.1.1 Standard Layout overview

You can skip reading this paragraph if you just want to install the standard layouts as quickly as possible.
Note: The Standard Layout is identical for both Synthesis SyncML Client and Server. It is therefore possible to connect a SyncML server to a database to make its data available for sync to remote devices, and in parallel use the Synthesis SyncML Client to keep the same database in sync with another SyncML server. This implies that the standard database layout contains all features (such as multi-user, multi-folder) needed for a SyncML server, even if these are not needed for a client in all cases.

Synthesis Sync client (and server) needs a database to read and write the synced objects (contact and calendar data in the standard layout). The standard layouts provided as samples include the following features:

- Sync user authentication: A table named 'SYNC_USERS' contains usernames and passwords for all users that are allowed to synchronize data in the database with a remote SyncML server (see <localdbuser> and <localdbpassword> configuration tags). Note that this has nothing to do with the user/password required to login into a remote SyncML server.
- Sync folder management: The standard layout provides multiple 'folders' (separate datastores). This allows different users to have different data sets in the same database. The table named 'SYNC_FOLDERS' contains a folder ID (name which is used to access the folder from the SyncML clients, see <localpathextension> configuration tag) and a decriptive text.
- Permission management. The table named 'SYNC_PERM' links between users and folders. A user can be granted permission to any folder by inserting a record in 'SYNC_PERM' containing the appropriate keys from 'SYNC_USERS' and 'SYNC_FOLDERS' resp.
- Syncable database management: SYNC_TARGETS manages the so called Sync Targets.A Sync Target is a remote database (one remote server often has more than one database) which is performing synchronisation with a local database. The client uses the SYNC_TARGETS records to remember the time of last sync and a few other items for every Sync Target.
- Change Log Management. The table named 'SYNC_MAP' contains mapping data. Mapping entries are needed to determine which items have been deleted on a per-server. All entries in SYNC_MAP are related to an entry in SYNC_TARGET.
- Sync Log. The table named 'SYNC_LOG' is used to log SyncML client activity. A new record is created for every started synchronisation of a Sync Target (so a device syncing contacts and events will create two entries, one for contacts, one for events). A log entry is created even if

synchronisation fails with a Sync Target. Note that no log entry is created if the sync session fails before any synchronisation is initialized (for example when a device tries to connect with invalid credentials).

- Data storage tables. These tables are named 'SYNC_CONTACTS' (for contact information) and 'SYNC_EVENTS' (for events), 'SYNC_TASKS' (for todo-list items) and SYNC_NOTES (for memos). While the layout of all the other tables will not need changes in most applications, the data tables can be totally different from this sample layout. A common customization will probably use all the other tables from the sample, but include customer-specific tables from an existing application for the data storage tables.

5.1.1.2 Database preparation for MS SQL Server

Note: This description is indended for using the Synthesis SyncML Client on the Windows platform. Using appropriate third-party drivers (such as those from www.openlinksw.com) MS SQL can also be accessed via ODBC from a Linux or MacOS X hosted SyncML Client.

We assume that you are familiar with using MS SQL Server. The following step by step description is written just as an orientation and not as an introduction to SQL Server.
The steps are described for SQL Server 7, but are similar in SQL Server 2000.

You can install the sync tables into an existing database or create a new one. Skip the following paragraph if you want to install into an existing database.

Creating a new Database

1. Open "SQL Server Enterprise Manager"
2. Select the server you want to create the database on.
3. Right-click "Databases" and select "New Database" from the context menu.
4. Give it a name (e.g. "SyncDB").
5. Press "OK".
6. Open the "Databases" folder in the tree: the new database should appear there.
7. Add database users required access the database (For a quick test, you can just use existing "dbo", but this is not a secure way to do it).

Creating the Sync Tables:

1. Open "SQL Query Analyzer".
2. Connect to the correct SQL Server.
3. In the query window, click the open button (folder icon).
4. Select the SQL script named "mssql.sql" provided in the Synthesis Sync Client distribution in the "mssql" directory in the "sample_config" directory.
5. Make sure to select the correct database name in the "DB" popup in the toolbar of the query window.
6. Execute the script (Press F5 or click the green right-pointing arrow)

Now you have the standard database tables for Synthesis Sync Client (and Server) installed in MS SQL Server.

Setting up the ODBC Datasource

ODBC Datasources are configured using the "ODBC Datasources" (or similar) control panel. It is hidden at different places on different versions of Windows. In older versions it can be found among the other Control Panels ("Start->Settings->Control Panel"), in Windows 2000 and newer it is under "Start->Applications->Administration").

The installation of a new datasource depends on the server used as well as on the ODBC driver used. So the following procedure applies to SQL Server only (altough other drivers might behave similar).

So installing the ODBC datasource for SQL Server consists of the following steps:

1. Open the "ODBC Datasources" control panel.
2. Select the "User DSN" or "System DSN" tab, depending on wether you want to make the datasoure available to the currently logged-in user only or all users and system processes on the machine.
3. Click the "Add..." button
4. From the list of available drivers select "SQL Server" for MS SQL (as if MS SQL was the only SQL server in the world...).
5. Click "Finish"
6. Now the driver-specific installation dialogs appear.
7. Enter a name for the datasource. This name will be used to identify the datasource in the Synthesis Sync Client config, so remember the name you use here.
8. Optionally enter a descriptive text for the datasource.
9. Enter (or select, if you have used it before) the server name.
10. Click "Next"
11. Select the appropriate authentification method. If you are not sure, use the second option and enter the database username and password in the text fields at the bottom of the dialog.
12. Make sure the "connect to SQL server..." check box is checked.
13. Click "Client Configuration" and enter server parameters (normally just select TCP/IP and set the computer name).
14. Click "Next".
15. Check "Change standard database..." checkbox.
16. Select the correct database from the popup list below.
17. Click "Next".
18. Click "Finish".
19. Click "Test Datasource..." to see if datasource works ok.
20. Click "OK" and close the "ODBC Datasources" control panel.

5.1.1.3 Database preparation for MySQL/MyODBC

Note: This description is primarily indended for using the Synthesis SyncML Client on the Linux platform.However, MySQL is and MyODBC are available for Windows as well - installation procedure might differ slightly but using the SyncML client with MySQL on Windows is no problem.

Some information about MySQL: MySQL is a very popular open source SQL database server which is widely used for web applications on many different platforms (Linux, MacOS X, Windows among them). MySQL sources and ready-to-install packages for most popular operating systems are available from www.mysql.com. The same site also provides an ODBC driver for MySQL named MyODBC.

Synthesis SyncML Client and the sample configuration delivered with it was tested with MySQL Version 4.0.16 and MyODBC 3.51.06, but is expected to run with earlier (and future, of course) versions of MySQL as well as the SyncML client does not make use of any special features of newer MySQL versions.

We assume that you are familiar with using MySQL on the target platform of your choice (Linux, Windows, Mac OS X). The following step by step description is written just as an orientation and not as an introduction to MySQL or MyODBC.

You can install the sync tables into an existing database or create a new one. Skip the following paragraph if you want to install into an existing database.

Creating a new Database

1. Start the mysql (mysql.exe under windows) client program with a username/password having enough privileges to create a new database.
2. Enter "create database syncdb;"
3. MySQL will create a new database
4. quit the mysql client program by entering "exit"

Creating the Sync Tables

1. Start the mysql (mysql.exe under windows) client program with a username/password having enough privileges to create a new database.
2. Execute the "mysql.sql" script provided in the "sample_config/mysql" directory by entering "source path/to/evaluationpackage/sample_config/mysql/mysql.sql;"
3. MySQL creates the database tables required for the SyncML client (and server)
4. Grant appropriate privileges to the user/password combination you want to use for accessing the database from the SyncML server. For example, executing "GRANT USAGE ON *.* TO syncml@localhost;" and then "GRANT SELECT,INSERT,UPDATE,DELETE ON syncdb.* TO syncml@localhost IDENTIFIED BY 'mysecret';" will allow access from the local machine with user/password = syncml/mysecret.
5. quit the mysql client program by entering "exit"

Now you have the standard database tables for Synthesis Sync Client installed MySQL Server.

Linux/Unix: Installing ODBC

Unlike Windows, under Linux and other UNIX flavors ODBC is not a standard component of the operating system. There are two well-known implementations of ODBC under non-Windows platforms, one is called iODBC and the other is called UnixODBC. In order for ODBC to work, one of them must be installed. Most Linux distributions contain one of them as ready-to-install packages.

Setting up the ODBC Datasource

Under Linux, there might be no graphical interface to the ODBC configuration in all cases (but iODBC features a graphical "iODBC Administrator"). In this case, you need to edit the odbc.ini file with a text editor. This file is usually located at /usr/local/etc. To add a ODBC datasource for MySQL you need to add something like the following:

```
 [mysql_sync_dsn]
Driver       = /usr/local/lib/libmyodbc3.so
Description  = MySQL ODBC 3.51 Driver DSN
SERVER       = localhost
PORT         = 3306
USER         = syncml
```

```
Password      = syncml
Database      = syncdb
OPTION        = 3
SOCKET        = /tmp/mysql.sock
```

Note that SOCKET is important only if SERVER is "localhost", as in this case clients connect directly to the mysql.sock. If server is an IP address/name, TCP/IP is used on the PORT specified. **Important: It seems that there is a problem in MyODBC 3.51 which causes that neither "database" nor "Socket" nor "Server" settings are properly read from odbc.ini. This is why we recommed including them in the connection string as well (see below).**

Under Windows, ODBC Datasources are configured using the "ODBC Datasources" (or similar) control panel. It is hidden at different places on different versions of Windows. In older versions it can be found among the other Control Panels ("Start->Settings->Control Panel"), in Windows 2000 and newer it is under "Start->Applications->Administration").

Installing the ODBC datasource for MySQL/MyODBC consists of the following steps on a Windows system:

- Install the MyODBC driver.
- Open the "ODBC Datasources" control panel.
- Select the "User DSN" or "System DSN" tab, depending on wether you want to make the datasoure available to the currently logged-in user only or all users and system processes on the machine.Click the "Add..." button
- From the list of available drivers select "MySQL ODBC 3.51" (or newer version).
- Click "Finish"
- The MySQL-specific set-up panel appears
- Enter a name for the datasource, such as "syncml" - this will be used to refer to the datasource from within the SyncML client's config file.
- You may also enter a description for the datasource
- Now you can enter the host name (usually localhost if MySQL and the SyncML client are running on the same machine)
- To test the connection, you can enter username/password as well (would be syncml/mysecret if database was set-up as shown in the example above) and click the "Test Data Source" button to see if the datasource is defined ok.
- If the test is ok, click the "OK" button to confirm the new datasource
- Close the ODBC control panel.

5.1.1.4 Database preparation for Interbase/Firebird SQL Server

Note: This description is indended for using the Synthesis SyncML Client on the Windows platform. Firebird itself is also available on Linux, so with appropriate third-party drivers (see www.ibdatabase.com or ofbodbc.sourceforge.net) a SyncML client using Firebird can also run on Linux.

Some information about Interbase/Firebird: Originally developed by Borland, Interbase 6 was made Open Source mid 2000 (available in under www.borland.com/devsupport/interbase/opensource). Further development of the open source version are now named Firebird. Firebird is available for download under firebird.sourceforge.net. Note that the GUI-Tool IBConsole used in the following description is not included in the Firebird distribution, so you need to download it separately from www.borland.com/devsupport/interbase/opensource.

Note that in order to use Interbase / Firebird, you need to have an ODBC driver for this server before you can use it with Synthesis Sync Client. There are various ODBC drivers available for Interbase, but not all of the same quality. We have tested Synthesis Sync Client successfully with "Gemini InterBase ODBC Driver", which is a commercial product (see www.ibdatabase.com)

We assume that you are familiar with using Interbase 6 or Firebird. The following step by step description is written just as an orientation and not as an introduction to Interbase or Firebird.

You can install the sync tables into an existing database or create a new one. Skip the following paragraph if you want to install into an existing database.

Creating a new Database

6. Open "IBConsole"
7. Doubleclick and login to the server you want to create the database on.
8. Right-click "Databases" and select "Create Database..." from the context menu.
9. Under "Filename(s)", enter the FULL PATH for the database file into the first field (e.g. "C:\databases\syncdb.gdb").
10. Make sure the checkbox "Register Database" is checked.
11. Under "Alias", enter a descriptive name (e.g. "Sync DB")
12. Press "OK".
13. In the tree view, the new database is automatically opened.
14. Add database users required access the database (For a quick test, you can just use existing "sysdba", but this is not a secure way to do it).

Creating the Sync Tables

1. Open "IBConsole".
2. Doubleclick and login to the server the database is on (you are already there if you just have created the database).
3. Doubleclick the database you want to use to connect to it.
4. From the menu, select "Tools"->"Interactive SQL".
5. In the Interactive SQL window, select "Query"->"Load Script" from the menu.
6. Select the SQL script named "interbase.sql" provided in the Synthesis Sync Client distribution in the "interbase_firebird" directory in the "sample_config" directory.
7. Execute the script (Press Ctrl-E or click the yellow lightning icon)

Now you have the standard database tables for Synthesis Sync Client installed in Interbase 6 or Firebird 1 Server.

Setting up the ODBC Datasource

ODBC Datasources are configured using the "ODBC Datasources" (or similar) control panel. It is hidden at different places on different versions of Windows. In older versions it can be found among the other Control Panels ("Start->Settings->Control Panel"), in Windows 2000 and newer it is under "Start->Applications->Administration").

The installation of a new datasource depends on ODBC driver used, so we cannot provide details here.

Installing the ODBC datasource consists roughly of the following steps:

- Install the Interbase/Firebird ODBC driver of your choice.
- Open the "ODBC Datasources" control panel.
- Select the "User DSN" or "System DSN" tab, depending on wether you want to make the datasoure available to the currently logged-in user only or all users and system processes on the machine. Click the "Add..." button
- From the list of available drivers select an ODBC driver for Interbase / Firebird.
- Click "Finish"
- Now the driver-specific installation dialogs appear. Usually, the only data that MUST be entered is the name of the datasource (you will need it for Synthesis Sync Server config) and the specification of the database to be accessed (a protocol selection, normally TCP/IP, a server name and the full path to the database file, like "C:\databases\syncdb.gdb" from the database creation example above). Additionally, you need to enter a valid user/password to connect to the database. Most drivers offer a "Test" button to test the connection and make sure the ODBC Datasource will work ok. **Important: Make sure that the same character set is chosen in the ODBC settings for the datasource as defined when creating the database. Otherwise, using non-ASCII-chars will cause database acesses to fail.**
- Confirm the settings and close the control panel.

5.1.2 Installing the ODBC SyncML client

Installing the client is simple:

1. Copy the file 'syncclient_odbc.exe' or 'syncclient_odbc_std.exe' (if you have the STD version from the 'bin_win32', 'bin_linux_x86' or 'bin_macosx' distribution directory to a directory of your choice. Note: on Linux and Mac OS X these files have no '.exe' extension.
2. Copy the file 'syncclient_odbc.xml' (if you have the evaluation or PRO version) or 'syncclient_odbc_std.xml' (if you have the STD version) from the appropriate distribution directory (according to the database you are using) in 'sample_config' to the same directory where you put the executable file.

5.1.3 Configuring the ODBC SyncML client

1. Use a text editor of your choice (Windows Notepad is ok) to open the copied 'syncclient_odbc.xml' file. The beginning looks similar to the following:

```
<?xml version="1.0"?>
<sysync_config version="1.0">

  <licensename>licensee name</licensename>
  <licensecode>XXXX-XXXX-XXXX-XXXX</licensecode>

  <debug>
    <enable option="all"/>
    <logpath>D:\sync\logs</logpath>
    <msgdump>no</msgdump>
  </debug>

  (... a lot of XML follows here, scroll down until you reach the <client> tag)
```

```
<client type="odbc">

  (... some more XML follows here, please scroll down)

  <logfile D:\sync\logs\synclog.txt</logfile>

  (... some more XML follows here, please scroll down)

  <datasource>DataSourceName</datasource>
  <dbuser>dbuser</dbuser>
  <dbpass>dbpassword</dbpass>
```

2. Modify the highlighted settings:
   - <licensename>, <licensecode>: Enter the licensing information you have received from Synthesis AG for your purchased product or evaluation package here. Note that demo versions do not need any licensing information.
   - <logpath>: Enter a full path that exists and is writable here. Sync Client uses this path to store diagnostic log files here.
   - <logfile>: Enter a full path to the desired log file and make sure the file can be created there. Sync Client uses this path to store diagnostic log files here.
   - <datasource>: This is the name of the datasource you have created above
   - <dbuser>/<dbpassword>: User and password used to access the database.
   - **Note:** some MySQL/MyODBC installations do not work correctly when only given the datasource name - for some reason the driver will not use the database name configured in the datasource, but expects that the database name is specified explicitly when connecting the datasource. In this case, you should use the <dbconnectionstring> tag instead of <datasource> and <dbuser>. This is described in detail in the configuration reference manual. For making it work quickly, just delete the <datasource> and <dbuser> tags from the config file and insert something like:

     ```
     <dbconnectionstring>DSN=syncml;UID=syncml;DATABASE=syncdb;</dbconnectionstring>
     ```

3. Save the config file.

5.2 Installing and configuring the DEMO SyncML client

5.2.1 Installing the DEMO SyncML client

Simply copy the executable ('syncserv_demo.exe' for Windows, 'syncserv_demo' for Linux and MacOS X) and 'syncserv_demo.xml' file into a directory where you have write access.

5.2.2 Configuring the DEMO SyncML client

There is basic configuration needed for the demo client. Of course, there are various options (such as logfile paths etc.) that can be configured, but for a basic operation, this is not required. Please refer to the SySync_Config_Reference manual for details.

6. Using the SyncML client

6.1 Specifying the SyncML server and databases to synchronize with

Before actually starting a synchronisation, you need to specify the server to synchronize with and some information what data you want to synchronize.
The related information is also contained in the XML config file.

1. Use a text editor to open the 'syncclient_odbc.xml' (ODBC version) the 'syncclient_demo.xml' (Demo version)or file. Scroll down to the comment that reads "<!-- Configuration of Sync Requests -->":

```
    <!-- Configuration of Sync Requests -->
    <!-- ============================== -->

    <localdbuser>test</localdbuser>
    <localdbpassword>test</localdbpassword>

    <defaultsyncmlversion>1.1</defaultsyncmlversion>

    <!-- choose encoding for SyncML messages: xml or wbxml -->
    <syncmlencoding>wbxml</syncmlencoding>

    <serverurl>http://yourserver.com/sync/sync.dll</serverurl>

    <!-- Enter user and password to be used to login to
         the remote server -->
    <serveruser>myuser</serveruser>
    <serverpassword>mypassword</serverpassword>


    <syncrequest datastore="contacts">
      <localpathextension></localpathextension>

      <dbpath>contacts</dbpath>

      <syncmode>twoway</syncmode>
      <slowsync>false</slowsync>
    </syncrequest>
```

2. Modify the highlighted settings:
   - < localdbuser >, < localdbpassword >: Enter the user name and password for the **local** database. This allows for having multiple user's data in the same database, but in case you only have a single user's data in the database, you can just use the predefined user/password **test/test**.
   - <syncmlencoding>: You can specify here what format the client uses to communicate with the server. Usually, this is the more efficient **wbxml**, but for debugging purposes you might want to set it to the more readable plain **xml** format.
   - <serverurl>: Enter the full URL of the SyncML server you want to sync with here.
   - <serveruser>/<serverpassword>: Enter the username and password required to log into the SyncML server you have specified with <serverurl>.

- <syncrequest>: The sample config contains 4 <syncrequest>s, but all of them are initially commented out. First, remove the comment bracket (<!-- -->) from the sync request for the data you want to sync with (**contacts, events, tasks, notes**). Then, edit the <dbpath> to match the server's name for that data type. The default value contained in the sample is compatible with Synthesis' SyncML server defaults, so if you are syncing with a Synthesis SyncML based server, you normally don't need to edit <dbpath> at all. Optionally, you can also set a different <syncmode> (**twoway, fromserver, fromclient**) and force a slow sync (<slowsync>true</slowsync>). For a normal, two-way sync, no changes are needed.

6.2 Running the SyncML client

To start the client, just doubleclick the 'syncserv_xpt_odbc_ev.exe' file (Under Linux and Mac OS X, start 'syncserv_xpt_odbc' from the command line.

Note that there are some command line options that can be used when staring the client in a console/terminal window (see Chapter 6.4 for details).

If there is an error condition (such as bad config or connection problem) the client does not start the sync session but displays the error condition. See Chapter 7 for explanation of error codes.

After successful startup, the console window will show some progress of the sync session like this:

```
SySync Client Win32 Demo Version 3.1.6.6

Reading config from : syncclient_odbc.xml
- Config file read from 'syncclient_odbc.xml'
- Debug log path: D:\synclogs\ (SEPARATE log file)
- starting client session with 'http://myserver/sync'
< SyncML message #1 sent to 'http://myserver/sync'
> SyncML message #1 received from 'http://myserver/sync'
< SyncML message #1 sent to 'http:// myserver/sync'
> SyncML message #1 received from 'http://myserver/sync'
- Remote alerts (201): two-way slow sync for './contacts'
  (source='contacts')
< SyncML message #2 sent to 'http://myserver/sync?sessionid=123'
> SyncML message #2 received from 'http://myserver/sync'
< SyncML message #3 sent to 'http://myserver/sync?sessionid=123'
> SyncML message #3 received from 'http://myserver/sync'

- Sync Statistics for 'contacts' (./contacts), slow sync
  ==================================================
                               on Client   on Server
  Added:                               1           0
  Deleted:                             0           0
  Updated:                             0           3

Client Session successfully terminated
```

6.3 Debug Log Files

Synthesis Sync Client can create a configurable amount of log information to help tracking down problems. The log information is controlled by the <debug> section in the config file:

```
<debug>
  <enable option="all"/>
  <logpath>D:\sync\logs</logpath>
  <msgdump>no</msgdump>
</debug>
```

There are many options available (see description of the "<debug>" tag in the config reference manual for details). For evaluation, we recommend to leave the settings as they are (full log information).

In addition, if you have problems with a server you want to report to Synthesis AG, you can set <msgdump> to "yes". This will generate a separate file for every message sent or received by the SyncML client.

You can also switch off debug information totally by replacing the "<enable option="all"/>" with "<disable option="all"/>".
To only log error conditions, use "<enable option="error"/>".

The files generated are (where "xxxx" is " odbc" for the ODBC version and "demo" for the Demo version).

- **client_xxxx.log**: This is the main debug log file. It contains all the messages that cannot be related to a certain SyncML session.
- **client_xxxx_sYYYYYYYYYY.log**: These are session-specific log files, where "YYYY" is the internal session ID. To see what happened in a Sync Session, look at these files.
- **client_xxxx_reqNNN_incoming.sml** and **client_xxxx_reqNNN_outgoing.sml**: These are 1:1 dumps of the incoming and outgoing SyncML messages (when <msgdump> is set to "yes", see above). "NNN" is a three digit request number, that starts counting at 1 when the server is started. Note that these normally contain binary (WBXML) data, and cannot be viewed with a text editor.

In case of reporting problems with Synthesis Sync Client, it helps a lot if you include the appropriate log files (main debug log file, log file for the session in question, and message dumps if possible.

In case you want to do performance testing, please switch off debug logging completely (or reduce it to error reporting) because log writing is done in a safe way (log file opened and closed for every log message) which may degrade performance a little. So use <disable option="all"/> in the config to make performance tests.

6.4 SyncML client command line options

The SyncML client, when started from a command line, has several command line options:

-n	do not wait for keypress before exit (=Linux/MacOS X default)
-w	wait for keypress before exit (=Windows default)
-v	just show version info
-t	only test syntax of configuration file, but do not perform a sync session
-s	singlestep messages (for debug). If specified, the client will prompt for continuation or abort after every SyncML message.
-r	interactively ask for message retry (for debug). This can be used to force a client to re-send the previous message to test a server's handling for re-sent messages.
-f <configpath>	specify config file path and name. By default, the config file is read from the current directory and is named as described in this manual (see above for details)
-h	show a short help text for the command line options.
-D <configvar>[=<value>]	Define configuration variables (which can be used to insert variable parts into config files, see "Configuration variables and conditional configuration" in the Synthesis SyncML Config Reference manual.

7. Error codes

This section lists the error codes that can occur (normally visible in the logs or on the console).

7.1 SyncML Status Codes

These codes are defined by the SyncML standard. For details, see http://www.openmobilealliance.org/release_program/ds_v12.html. Note that this list is not complete, but only contains the codes that are important for the SyncML engine.

101	Server is busy (licensed number of connection limit reached)
200	OK, successful operation
201	Item added
207	Conflict resolved with merge
208	Conflict resolved - client wins
209	Conflict resolved by duplicating item
210	Deleted without archive
211	Item not deleted
212	Authentication accepted for entire session
213	Chunked item accepted and buffered (this status is sent for each non-final part of a data item that has been split across multiple SyncML messages)
400	Bad request
401	Unauthorized (bad credentials)
403	Forbidden (e.g. attempt to write to a read-only database)
404	Object not found
405	Command not allowed
406	Optional feature not supported
407	Authentication required (no credentials found)
408	Timeout
409	Conflict, operation failed
410	Gone, requested object not here any more
412	Incomplete command
415	Unsupported media type or format
418	Object already exists
419	Conflict resolved with server data
420	Device full
500	Command failed
501	Command not implemented
503	Service unavailable
505	DTD version not supported
508	Slow sync required
509	Authentication required
510	Database error
511	Server error
512	Synchronisation failed
513	SyncML Version not supported

514	Cancelled

7.2 Internal Error Codes

0	No error
10000..10999	These have the same meaning as the SyncML Status Codes (see 7.1), but they are offset by 10000 to make clear that they were generated internally, and not sent or received via SyncML.
20001	Bad or unknown transport protocol
20002	Fatal problem with SyncML encoder/decoder
20003	Cannot open communication
20004	Cannot send data
20005	Cannot receive data
20006	Bad content type (message received with an unknown MIME-type)
20007	Error processing incoming SyncML message (for example invalid XML or WBXML formatting)
20008	Cannot close communication
20009	Transport layer authorisation (e.g. HTTP auth) failed
20010	Error parsing XML config file
20011	Error reading config file
20012	No configuration found at all, or not enough for requested operation (client)
20013	Config file could not be found
20014	License expired or no license found
20015	Internal fatal error
20016	Bad handle
20017	Session aborted by user
20018	Invalid license
20019	Limited trial version
20020	Connection timeout
20021	Connection SSL certificate expired
20022	Connection SSL certificate invalid
20023	incomplete sync session (some datastores failed, some completed)
20025	Out of memory
20026	Connection impossible (e.g. no network available)
20027	Establishing connection failed (e.g. network layer login failure)
20028	element is already installed
20029	this build is too new for this license (need upgrading license)
20030	function not implemented
20031	this license code is valid, but not for this product (e.g. STD license used in PRO product, or client license in server product)
20033	this build is too old for this SDK/plugin
20034	unknown subsystem
20500..20599	These represent SIG_xxx codes in Linux versions of the server. Unexpected SIG_xxx will generate a error code of 20500+signal_code.

20998	Internal unkown exception
20999	Unknown error
21000…21999	Database plugin module specific error codes