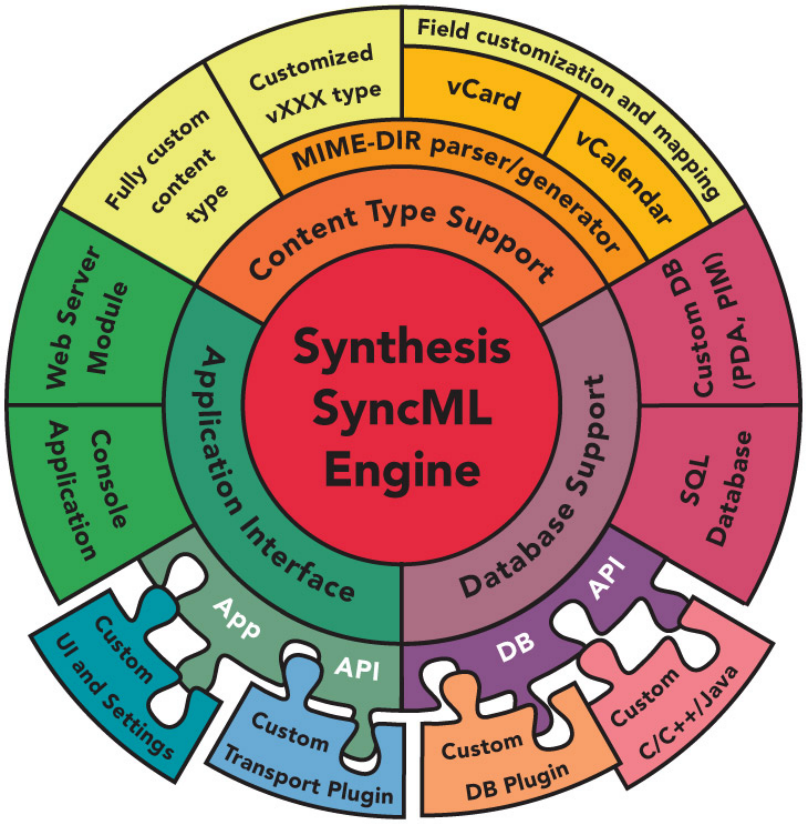
Sustainable Software Concepts

SYNTHESIS AG

www.synthesis.ch

Installation Manual for Synthesis SyncML Server 3.2

SyncML



This manual was written for Synthesis SyncML Server Version 3.2.0.11

Synthesis Sync Server 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

Use of the Synthesis Sync Server software and other software accompanying your license (the "Software") and its documentation is at your sole risk. The Software and its documentation (including this manual), and software maintainance by Synthesis AG, if applicable, are provided "AS IS" and without warranty of any kind and Synthesis AG EXPRESSLY DISCLAIM ALL WARRANTIES, EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE, TITLE, AND NON-INFRINGEMENT. IN NO EVENT SHALL SYNTHESIS AG BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

1. Introduction

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

If you have used earlier versions of the Synthesis SyncML server, please refer to Chapter 2 for a quick overview of the changes and especially the **Migration Guide** for updating existing 2.1 installations to 3.0, see 3.4.

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

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 Server – 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 Server (PRO version).

To allow using existing database layouts, the Synthesis Sync Server 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 Server comes with sample files for creating a standard layout in common SQL databases like MySQL and MSSQL. This allows you to set-up a working SyncML solution in a short time. It also provides a starting point for customizing the server to your specific needs.

Therefore, the setup guide in this manual describes how to use the standard layout (see 5.1). If you are new to Synthesis Sync Server, we recommend that you first install and test a standard layout before doing your own configuration.

Synthesis Sync Server - DEMO Version is a free demonstration version. It is available for free and provides full SyncML functionality, but stores data in TAB-separated text files or in SQLite3 databases. While this version is not indended (nor suitable) for productive use, it is installed in seconds and very useful to test SyncML clients or get an insight into Synthesis Sync Server'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 3.0 and STD vs PRO

The following matrix shows the most important changes between 2.1 and the new 3.0 version 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 required for newer devices.		➽		➽
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 server. See Migration Guide in 3.4.**		➽		➽
SyncML DS 1.2 filtering: Support for new SyncML DS 1.2 filter capabilities to restrict sync set to a subset based on a fil-				➽

ter expression.				
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, login procedures, data-comparison and merging algorithms 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.				➳
Full email implementation in sample config: The new sample config now includes a full-featured sample of a email sync datastore, including advanced functionality like Mark-For-Download, attachments, size limiting etc.				➳
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 server

- SyncML DS 1.2 only clients (like some newer Symbian UIQ smartphones) now work with the Synthesis SyncML Server. All SyncML 1.1.1, 1.1, 1.0 based clients are still supported.
- 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) clients, 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 server now supports email sync and vBookmark sync with devices that support these data types.

3.3 Changes in the new 3.x 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.x

3.4.1 Overview

What needs to be done to upgrade an existing 2.1.x based installation to 3.x?
There are three possibilities:

1. **Just use the new server 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 standalone server 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 server 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.x with full Suspend & Resume support

The following is a step by step guide how to update an existing Synthesis SyncML Server 2.1 installation to version 3.x 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 server 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', LAST-
SYNC=%L WHERE TARGET_KEY='%t'
UPDATE SYNC_TARGETS SET ANCHOR='%A', LASTSYNC=%L, LAST-
TOREMOTESYNC=%RL, RESUMEALERT=%SUA, LASTSUSPEND=%SU,
LISOURCE='%pSU', LITARGET='%pTU', LISTATUS=%pSt, PIS-
TATE=%pM, PITOTALSZ=%pTS, PIUNCONFSZ=%pUS, PIS-
TOREDSZ=%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 LO-
CALID=%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.x server, have a look at the new 3.x sample config and maybe use a difference viewer to compare your config to the new 3.x – 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 / IIS

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 SERVER
├──some read-me, license, disclaimer text files
├──docs
│       *.txt    (read-me, history information etc. in plain text format)
│       *.pdf    (documentation in Adobe PDF format)
├──bin_win32
│       sync.dll                        (same name for all versions)
│       syncserv_xpt_odbc.exe           (PRO version)
│       syncserv_xpt_odbc_std.exe       (STD version)
│
├──sample_config
│   ├──interbase_firebird
│   │       interbase.sql
│   │       interbase_upgrade_2.1_to_3.0.sql
│   │       syncserv_odbc.xml       (for PRO version)
│   │       syncserv_odbc_std.xml   (for STD version)
│   │
│   ├──mssql
│   │       mssql.sql
│   │       mssql_upgrade_2.1_to_3.0.sql
│   │       syncserv_odbc.xml       (for PRO version)
│   │       syncserv_odbc_std.xml   (for STD version)
│   │
│   └──mysql
│           mysql.sql
│           mysql_upgrade_2.1_to_3.0.sql
│           syncserv_odbc.xml       (for PRO version)
│           syncserv_odbc_std.xml   (for STD version)
│
└──unsupported
        admin.php
        config.php
        index.php
        read_me.txt
        logo.jpg
```

'docs' Directory

- 'SySync_Server_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

- 'sync.dll' : This is the ISAPI DLL. This must be installed in a IIS-accessible directory. Note that the name is the same for all versions (STD and PRO).
- 'syncserv_xpt_odbcXXX.exe' where XXX is "_std" for STD: This is a console-based, stand-alone version of the server. This version is intended to be used for testing a configuration, because it is easier to start up and shut down and displays informative messages on the console window. It is identical in functionality to the ISAPI version except that it can handle only a single connection at a time. **It is not indended for productive use, just for testing configurations and clients.**

'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 ISAPI and standalone versions of the Synthesis Sync Server. 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 - Apache (Linux x86)

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 SERVER
├──install.sh                          (shell script to install Apache 1.3)
├──install2.sh                         (shell script to install Apache 2.0)
├──some read-me, license, disclaimer text files
├──docs
│       *.txt    (read-me, history information etc. in plain text format)
```

```
|         *.pdf    (documentation in Adobe PDF format)
|---bin_linux_x86
|         sysync_session_odbc          (PRO version)
|         sysync_session_odbc_std      (STD version)
|         syncserv_xpt_odbc            (PRO version)
|         syncserv_xpt_odbc_std        (STD version)
|---src
|   \---mod_sysync
|             compile.sh           (shell script to compile for Apache1.3)
|             compile2.sh          (shell script to compile for Apache2.0)
|             *.c, *.h             (actual source files)
|
|---sample_config
|   |---interbase_firebird
|   |         interbase.sql
|   |         interbase_upgrade_2.1_to_3.0.sql
|   |         syncserv_odbc.xml      (for PRO version)
|   |         syncserv_odbc_std.xml  (for STD version)
|   |
|   |---mssql
|   |         mssql.sql
|   |         mssql_upgrade_2.1_to_3.0.sql
|   |         syncserv_odbc.xml      (for PRO version)
|   |         syncserv_odbc_std.xml  (for STD version)
|   |
|   \---mysql
|             mysql.sql
|             mysql_upgrade_2.1_to_3.0.sql
|             syncserv_odbc.xml      (for PRO version)
|             syncserv_odbc_std.xml  (for STD version)
|
|
\---unsupported
          admin.php
          config.php
          index.php
          read_me.txt
          logo.jpg
```

distribution directory root

- 'install.sh': A shell script for installing the SyncML server for some supported Linux distributions. See 5.1.4.1 for details.
- Read-me, disclaimer, license text files depending on version.

'docs' Directory

- 'SySync_Server_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

- 'sysync_session_odbc' for the PRO and 'sysync_session_odbc_std" for STD version: This is a special executable which contains the actual SyncML server engine. 'mod_sysync' (see 'src/mod_sysync' below) will call this program to perform SyncML sessions.
- 'syncserv_xpt_odbc' for the PRO and 'syncserv_xpt_odbc_std" for STD version: This is a console-based, standalone version of the server. This version is intended to be used for testing a configuration, because it is easier to start up and shut down and displays informative messages on the console window. It is identical in functionality to the Apache version except that it can handle only a single connection at a time. **It is not indended for productive use, just for testing configurations and clients.**

'src/mod_sysync' Directory

To run the SyncML server as a part of the Apache web server, the 'mod_sysync.so' Apache 1.3 plugin module or 'mod_sysync2.so' Apache 2.0 plugin module is required: This directory contains the source code required to compile the module for your target system (as the binary form is dependent on various target system factors, we do not provide a precompiled binary for it). It is identical for both STD and PRO versions of the server (they only differ in the "sysync_session_xxx", see above) and serves as the interface between the Apache HTTP server and the actual SyncML server session processes (see below).

- '*.c', '*.h' files: The actual source files
- 'compile.sh': a shell script, which compiles 'mod_sysyn.c' for Apache 1.3 (using the apache 'apxs' tool)
- 'compile2.sh': a shell script, which compiles 'mod_sysync2.c' for Apache 2.0 (using the apache 'apxs2' tool).

'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 Apache and standalone versions of the Synthesis Sync Server. 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 PRO-**

VIDED 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_Server_manual.pdf' : This file
- 'SySync_config_reference.pdf': This extra manual contains detailed information for writing config files.
- 'syncserv_demo.exe' : This is the demo sync server for Windows. It is a console-based, stand-alone version of the server using text files for storing data. It can handle only a single connection at a time.
- 'syncserv_demo' : For MacOSX and Linux, the executable file has no extension.
- 'syncserv_demo.xml' : Sample config file. This file contains everything you need to start up the server.
- 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 on Page 32 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). The ODBC server comes as an ISAPI DLL for productive use with IIS/Windows (see 5.1.3), or as an Apache module for productive use on Linux/x86 (see 5.1.4) and as a standalone application for testing and developing configurations (see 5.1.2). We recommend that if you are new to Synthesis SyncML Server, start testing with the standalone version as you have immediate visible feedback on the console screen. If everything is working fine, then you can easily switch to the ISAPI or Apache version.

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.

Synthesis Sync 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 with the server. In addition, the user record also contains a field named READONLY. If this field is non-zero, the user will not be able to write any data to the server via SyncML.
- Sync folder management: The standard layout provides multiple 'folders' (separate datastores). This allows different users to have different data on the same server. The table named 'SYNC_FOLDERS' contains a folder ID (name which is used to access the folder with SyncML) 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.
- Device Management. The table named 'SYNC_DEVICES' is maintained by the SyncML server and will get an entry for every device that successfully connects to the SyncML server. This table can be useful to retrieve additional information about the devices that use a SyncML server.
- Syncable database management: SYNC_TARGETS manages the so called Sync Targets.A Sync Target is a remote database (one remote device often has more than one database) which is performing synchronisation with a local database. The server 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-device basis and for

associating server and client records to each other. 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 server 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 Server 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 Server.

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 Server 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 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).

The ODBC data source to be used by the Synthesis Sync Server must be installed as "System DSN", not "User DSN". User DSN are only accessible by the currently logged in user, and are therefore not suitable for use by a server.

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

1. Open the "ODBC Datasources" control panel.
2. Select the "System DSN" tab.
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 Server 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 Server on the Linux platform.However, MySQL is and MyODBC are available for Windows as well - installation procedure might differ slightly but using the SyncML server 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 Server 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 server 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 tables in a database.
2. Enter "use syncdb;", assuming the database you want to install the sync tables into is called "syncdb" (which is the case when you have created the database as in the example above).
3. Execute the "mysql.sql" script provided in the "sample_config/mysql" directory by entering "source path/to/evaluationpackage/sample_config/mysql/mysql.sql;"
4. MySQL creates the database tables required for the SyncML server
5. 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.
6. quit the mysql client program by entering "exit"

Now you have the standard database tables for Synthesis Sync Server 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 /etc (e.g. Debian) or /usr/local/etc (e.g. Redhat). To add a ODBC datasource for MySQL you need to add something like the following:

```
[mysql_sync_dsn]
Driver       = /usr/lib/odbc/libmyodbc.so
Description  = SyncML DNS with MyODBC 3.51
SERVER       = localhost
PORT         = 3306
USER         = syncml
Password     = mysecret
Database     = syncdb
OPTION       = 3
```

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").

It is important to note that the ODBC data source to be used by the Synthesis Sync Server must be installed as "System DSN", not "User DSN". User DSN are only accessible by the currently logged in user, and are therefore not suitable for use by a server.

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 "System DSN" tab.
- 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 server'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 server 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 Server on the Windows platform. Firebird itself is also available on Linux, so with appropriate third-party drivers (e.g. see www.ibdatabase.com) a SyncML server 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 Server. There are various ODBC drivers available for Interbase, but not all of the same quality. We have tested Synthesis Sync Server 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

7. Open "IBConsole"
8. Doubleclick and login to the server you want to create the database on.
9. Right-click "Databases" and select "Create Database..." from the context menu.
10. Under "Filename(s)", enter the FULL PATH for the database file into the first field (e.g. "C:\databases\syncdb.gdb").
11. Make sure the checkbox "Register Database" is checked.
12. Under "Alias", enter a descriptive name (e.g. "Sync DB")
13. Press "OK".
14. In the tree view, the new database is automatically opened.
15. 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 Server 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 Server 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.

It is however important to note that the ODBC data source to be used by the Synthesis Sync Server must be installed as "System DSN", not "User DSN". User DSN are only accessible by the currently logged in user, and are therefore not suitable for use by a server.

So 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 "System DSN" tab.
- 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 and Using the Standalone Console Application

5.1.2.1 Installing the Standalone Version

Installing the standalone version is simple:

1. Copy the file 'syncserv_xpt_odbc.exe' or 'syncserv_xpt_odbc_std.exe' (if you have the STANDARD 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 'syncserv_odbc.xml' (if you have the evaluation or PRO version) or 'syncserv_odbc_std.xml' (if you have the STANDARD version) from the appropriate distri-

bution directory (according to the database you are using) in 'sample_config' to the same directory where you put the '.exe' file.

5.1.2.2 Configuring the Standalone Version

1. Use a text editor of your choice (Windows Notepad is ok) to open the copied 'syncserv_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>
    <logpath platform="win32">D:\sync\logs</logpath>
    <logpath platform="linux">/var/log/syncml</logpath>
    <logpath platform="macosx">/var/log/syncml</logpath>

    (... debug log options …)

  </debug>

  <transport type="xpt">
    <protocol>HTTP</protocol>
    <httpport>80</httpport>
  </transport>

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

  <server 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 Server 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 Server uses this path to store diagnostic log files here.
   - <httpport>: If the machine you are using to run the standalone version has no web server running, you can leave it at 80. Otherwise, you need to specify a port that is not used by the web server.

- <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.

5.1.2.3 Running the Standalone version

To start the standalone version, just doubleclick the 'syncserv_xpt_odbc.exe' file.
Under Linux and Mac OS X, start 'syncserv_xpt_odbc' from the command line. See Chapter 6 for available command line options.

If there is an error condition (such as bad config or already used HTTP port) the server does not start up but only displays the error condition. If the error is "***** Terminated due to error: Error selecting protocol" this is normally because the selected port number is in use by another application (e.g. a web server). See Chapter 8 for explanation of error codes.

After successful startup, the console window looks like:

```
SySync Server Win32 Desktop ODBC+plugin PRO Version 3.1.6.6

- Evaluation Version Expiring after 2003-10-15

- Config file read from 'D:\sync\syncserv_odbc.xml'
- Debug log path: D:\sync\logs\ (SEPARATE log file)
- starting HTTP server on port 80
```

Now the Server is waiting for a connection.

5.1.2.4 Connecting to the Standalone SyncML Server

The SQL script used to set-up the database automatically installs a test user and a test data folder. So you can connect the SyncML server from any SyncML client with the following connection settings:

- Server URL: Of course, the machine you are running the SyncML server on must be accessible from the outside. Note that the standalone version does not examine the path in the URL, so only the server name and port number are important. For the standalone version, the following URLs are treated the same:
  - **http://www.yourserver.com:88/this/that/anything.dll**
  - **http://www.yourserver.com:88/**

  Both refer to the server which must (in this case) be running on port 88.
  Important Note: Some clients, such as the Nokia 9210 or 7650 devices, do not work correctly with SyncML Server URLs that do not contain a path (sync will abort after the second answering message is sent to the client). So we recommend always using a path (first example

above) to avoid these problems. Note also that some devices (e.g. Siemens S55) do not work properly with servers using a port other than 80 for HTTP.

- Server user name: **test**
- Server password: **test**
- Contacts database path: **contacts** (note that in the 1.0.5 samples, this was called *contact*, singular)
- Events database path (for clients such as T39m-T68i, S55): **events**
- Tasks database path (for clients such as T39m-T68i, S55): **tasks**
- Combined events/tasks (e.g. for 9210 or P800): **calendar**
- Notes database path: **notes**

With these settings, you should be able to perform a sync session. The console output for a successful sync session (using a Nokia 9210, synchronizing contacts) looks like the following:

```
Started new SyncML session (server
id=7291327798616013976)
> SyncML message #1 received from 'IMEI:123456789123'
- Remote alerts (200): two-way normal sync for
  './contacts' (source='./D\System\Data\pretest.cdb')
< SyncML message #1 sent to 'IMEI:123456789123'
> SyncML message #2 received from 'IMEI:123456789123'
- Starting Sync with Datastore './contacts', slow sync
  0 local items are new/changed/deleted for this sync
< SyncML message #2 sent to 'IMEI:123456789123'
> SyncML message #3 received from 'IMEI:123456789123'

- Sync Statistics for Datastore './contacts', normal sync
  ================================================
                               on Server   on Client
  Added:                               0           0
  Deleted:                             0           0
  Updated:                             0           0
  SlowSync Matches:                    0
  Server won Conflicts:                0
  Client won Conflicts:                0
  Conflicts with Duplication:          0

< SyncML message #3 sent to 'IMEI:123456789123'
Terminated SyncML session (server id=7291327798616013976)
```

5.1.3 Installing and Using the IIS ISAPI Version (Windows)

5.1.3.1 Installing the ISAPI Version

The ISAPI version of Synthesis Sync Server has to be installed like any other ISAPI extension:

1. Copy the file 'sync.dll' from the 'bin_win32' distribution directory to a directory which is published to the web via IIS. It is recommended to create a new directory for this purpose and publish it by creating a new "Virtual Directory" in IIS for it. You can also use an already existing published directory.
2. Configure IIS to allow execution of ISAPI extensions in that directory. This can be done in the "Properties" dialog of the virtual directory in IIS.
3. Copy the file 'syncserv_odbc.xml' (if you have the evaluation or PRO version) or 'syncserv_odbc_std.xml' (if you have the STANDARD version) from the appropriate distribution directory (according to the database you are using) in 'sample_config' to the windows system directory. Normally, this directory is 'C:\WINNT'. If you don't want to have the config file in the windows directory, see next paragraph "Multi-Server operation".
4. Make sure that the Web server uses ISAPI caching. Synthesis Sync Server will not work with ISAPI caching turned off (ASP applications won't work, either). This option can be found in the "Properties" dialog of the root for the website, on the "root directory" tab, after clicking the "Configuration" button, on the "application mappings" tab.

5.1.3.2 Multi-Server operation

This is a new feature of the 2.0 version. If you want to run more than one ISAPI-based SyncML server on the same machine, you will probably want to use two different configuration files. By default, the Synthesis SyncML server searches for the config file in the windows system directory, as described above. To use different config files, a new (optional) mechanism has been added as follows:

- If the SyncML server (ISAPI version only!) finds a file named 'syncconfig.txt' in the same directory where the 'sync.dll' is located, it will read this file and interpret the first non-empty line not starting with a # as a path name to the config file to be used. If the path name starts with a period, it is interpreted as a relative path (relative to the 'sync.dll' location), otherwise it must be a fully specified absolute path. **Please note that the config file should always be located outside the directory tree which is available via the web server as it contains security relevant data such as database passwords etc.**
- If no 'syncconfig.txt' is found, the SyncML searches for the default global config file in the windows system directory.
- If 'syncconfig.txt' is found, but does not contain a valid pathname, the SyncML server will **not** start and will not search for the default global config file.

Note that with this mechanism, it is also possible to give the config files custom names, such as 'syncml_config_server3.xml' etc. as long as they are correctly referred to from a 'syncconfig.txt' file.

5.1.3.3 Configuring the ISAPI Version

1. Use a text editor of your choice (Windows Notepad is ok) to open the copied 'syncserv_odbc.xml' file. The beginning looks approximately as follows:

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

  <licensename>licensee name</licensename>
```

```
<licensecode>XXXX-XXXX-XXXX-XXXX</licensecode>

<debug>
  <logpath platform="win32">D:\sync\logs</logpath>

  (... debug log options …)

</debug>

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

<server 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 Server uses this path to store diagnostic log files here. Note that the path must be writable for the user under which the server is running (normally called IUSR_xxx and/or IWAM_xxx).
   - <logfile>: Enter a full path to the desired log file and make sure the file can be created there. Sync Server 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.

5.1.3.4 Running and stopping the ISAPI version

The ISAPI version of Synthesis Sync Server is started by accessing it via IIS (see the following paragraph). It reads the config file when it is first accessed.

In order to apply changes made in the config file, you can either completely restart IIS (not recommended), or just "unload" the application. To unload, in IIS open the "Properties" dialog for the (virtual) directory where the "sync.dll" file is located. On the "(virtual) directory" tab, click the "Unload" button and confirm the unload operation. Note that the "unload" button might not be available in older versions of IIS.

Unload completely stops the Synthesis Sync Server. You can now for example replace the "sync.dll" file (in order to install a new version). When the next Sync request comes in, the config file is read again, and changes you have made in the meantime are applied.

In some cases, strict Windows file access permissions do not allow the SyncML server to either access the configuration file or write any logs. As the ISAPI version has no console to show what's wrong, it returns a HTML error page when accessed with a browser and either config is invalid or logfiles cannot be written. So, if you don't get any log files when accessing the Server with a SyncML client, open a Web Browser and enter the SyncML URL. If there is a problem with the configuration, a simple page titled "Synthesis SyncML Server Fatal Error" will show two kind of error codes:

- The first code is an internal error code. If it is 0, config was read ok. Possible error codes are: 20010=config parse error, 20011=config read error, 20012=config not sufficient, 20013=config file not found. See Chapter 8 for explanation of other error codes.
- The second code is the Win32 error code caused by opening the shown logfile path for append. If it is not 0, the server cannot write log files at all and you should try to find out why. Common error codes are: 5=no permission, 3=path not found (see Win32 reference for error code lists or use a tool like errlook.exe). You can also try to completely disable debug logs by specifying <disable option="all"> in the <debug> section and see if the server works this way.

As the ISAPI version has no console to show configuration error details, Synthesis Sync Server tries to create a log file in the windows system directory (the same directory where the config file is read from, usually "C:\WINNT"). The log file is named "sysync_isapi_odbc_cfgerr.log". It contains the same error messages that would appear on the console when using the standalone version, but prefixed with a timestamp.

All other log information is written to the path specified with <logpath>.

5.1.3.5 Connecting to the ISAPI based SyncML Server

The SQL script used to set-up the database automatically installs a test user and a test data folder. So you can connect the SyncML server from any SyncML client with the following connection settings:

Server URL: Of course, the machine you are running the SyncML server on must be accessible from the outside.
Important Note: Some clients (e.g. Siemens S55) do not work properly with servers using a port other than 80 for HTTP.

- Server user name: **test**
- Server password: **test**
- Contacts database path: **contacts** (note that in the 1.0.5 samples, this was called *contact*, singular)
- Events database path (for clients such as T39m-T68i, S55): **events**
- Tasks database path (for clients such as T39m-T68i, S55): **tasks**

- Combined events/tasks (e.g. for 9210 or P800): **calendar**
- Notes database path: **notes**

With these settings, you should be able to perform a sync session. To see error information, look at the log files created (see 5.3).

5.1.4 Installing and Using the Apache Module Version

Note: This description is primarily indended for using the Synthesis SyncML Server on the Linux x86 platform.However, the Apache version is also available for other platforms - but installation might slightly differ.

5.1.4.1 Installing the Apache module Version

Important: Note that installing the Apache module Version of the SyncML server requires that the 'mod_sysync' (Apache 1.3 module) or 'mod_sysync2' (Apache 2.0 module) is compiled for your Apache server. Therefore the Apache headers, the 'apxs' tool ('apxs2' for Apache 2.0) and a suitable C compiler must be installed on your system. Please make sure you have the appropriate packages installed before proceeding with the installation.

The Apache module version comes with a script called 'install.sh'. Use 'install2.sh' when working with Apache2. This script knows a few Linux distributions (Debian, SuSE, RedHat) and Apple's MacOSX and its preferred paths to install the SyncML server. It also automatically compiles the mod_sysync for you, if needed. If your distribution is supported, using 'install.sh' is the easiest way to install. Simply type execute the script from a terminal window. If the script tells you that your system is not supported, you need to do a manual install - please refer to 5.1.4.2. ‚install.sh' and ‚install2.sh' decide for each distribution, whether the older gcc 2.95 or the newer gcc 3.3 will be chosen. Apple's MacOSX up to 10.4 (Tiger) is based on Apache 1.3, 10.5 (Leopard) is based on Apache 2.

After running the script, the following things need to be done manually:

- Your httpd.conf must be modified to include a LoadModule directive for the SyncML server plugin 'mod_sysync' (for Apache 1.3) or 'mod_sysync2' (for Apache 2.0) . The 'install.sh' script (or 'install2.sh' for Apache2) outputs the information how to do that. Using the apxs tool, this can be done automatically. However depending on how your httpd.conf is set up (with Ifdefs etc.), manual edits might still be required. See Apache docs for details.
- Your syncserv_odbc.xml (or syncserv_odbc_std.xml) file must be edited to make sure the <logpath> and <logfile> directives point to the the log directory created by 'install.sh'. In addition, you need to configure the SyncML server for accessing the correct database etc. - see 5.1.4.4.
- You need to add the appropriate Apache directives to have the SyncML server plugin actually serve your SyncML URL. This can be done by adding a <Location> or <Directory> or <Files> directive into your httpd.conf or even with an appropriate .htaccess file. See Apache docs for details.
  The 'install.sh' script produces a sample <Location> directive in the config directory in the file 'sample_directives_for_httpd.conf'. You can simply copy and paste the contents of this file to your httpd.conf to have the SyncML server activated at '/sync' on your web server. See Chapter 7 for detailed descriptions for the mod_sysync / mod_sysync2 directive

5.1.4.2 Manually installing the Apache module Version

The Apache version of Synthesis Sync Server consists of two parts:

- an Apache plugin module 'mod_sysync.so' (for Apache 1.3) or 'mod_sysync2.so' (for Apache 2.0) which serves as a communication interface between the Apache httpd and the Synthesis SyncML sessions (which run as separate processes, see below). This module must be compiled from source as its binary form depends on many parameters. Source is included in the distribution in the 'src/mod_sysync' directory.

- the actual server program ('sysync_session_odbc') which contains the actual SyncML engine. A server programm is automatically started by the 'mod_sysync' / 'mod_sysync2' Apache module for each SyncML session. This server program is provided in binary form in the 'bin_linux_x86' or 'bin_macosx' directory.
  Note: the server program is the same for use with Apache 1.3 or Apache 2.0.

Technical Note: The probably uncommon sounding idea behind having a separate process for each SyncML session is due to fundamental differences in ressource usage between normal HTTP serving and running a SyncML session. Normal HTTP serving requires large amounts of data to be delivered quickly in in high frequency, while the session context information is usually small. SyncML sessions transfer small amounts of data in low frequency, but the session context can be very large.
With server version 3.0.2.0, the server process can still be used as before (one server process per sync session), but in addition it also **allows a multithreaded operation, which runs multiple sessions within the same server process**. This can significantly increase efficiency for example when using Java based plugins, as the Java VM must be instantiated once per process, but can be shared among multiple threads in the same process. Please refer to the <transfer type="pipe"> description in the config reference manual (SySync_config_reference.pdf) for description of the relevant <maxthreads> and <maxsessionruns> config options.

For the following examples, we assume that the Apache HTTP server is installed at "**/usr/local/httpd**" (as the case in a standard RedHat 9 installation for example). Note that the following is only an example for installing the SyncML server - you are free to choose different locations to store files according to your needs. The example follows the strategy to have all files (executables, config, logs...) belonging to one service or application within subdirectories of a single common directory; however, some Linux distributions (e.g. Debian) use an entirely different strategy by keeping all the executables, all logfiles, all config files together in common places.

To install the Apache 1.3 or Apache 2.0 module version:

1. Compile the mod_sysync module for your Apache server. Provided that C compiler, Apache headers and the apxs tool are properly installed, you can simply execute the script "src/mod_sysync/compile.sh". The output file will be generated in "src/mod_sysync".
   (use "src/mod_sysync/compile2.sh" for use with Apache 2.0)

2. Copy the file "mod_sysync.so" from the "src/mod_sysync" directory (where it should have been created by compiling in step 1 above) to the 'libexec' directory of your Apache server (e.g. "/usr/local/httpd/libexec/mod_sysync.so")
   (the module is called "mod_sysync2.so" for use with Apache 2.0)

3. Make sure the file has sufficient execution permission to be executed by the httpd server user (e.g. "nobody" or "wwwrun").

4. Create a new subdirectory "syncml" in "/usr/local/httpd". You can choose any other location of course. Make sure the subdirectory has sufficient permissions for the httpd server user to access it for read and write (needed for log files)

5. Create the following subdirectories in "syncml": "bin", "conf", "pipes", "logs".

6. Make sure that "bin" can be accessed for execute, and "pipes" and "logs" can be accessed for read and write by the httpd server user.

7. Copy the file "sysync_session_odbc" (or "sysync_session_odbc_std", if you are using the STD version) from the "bin_linux_x86" or "bin_macosx" distribution directory to "/usr/local/httpd/syncml/bin". Take the files from „bin_linux_x86_gcc33“, if you want to use the newer GCC 3.3 compiled binaries.

8. Make sure the file has sufficient execution permission to be executed by the httpd server user.
   NOTE: An activated **SELinux** (Security-Enhanced Linux) as in Fedora Core 3 distribution does not allow creation of pipes for Apache2 by default, so either "setenforce 0" must be called (to switch off SELinux) or the context for the pipes directory (usually the directory "/usr/local/httpd/sysync/pipes") must be set accordingly (with the utility "chcon"). For details see the Apache2 and the SELinux documentation.

9. Copy the appropriate config file "syncserv_odbc.xml" (or "syncserv_odbc_std.xml") from the "sample_config" distribution directory to "/usr/local/httpd/syncml/conf".

10. Now edit Apache's main configuration file 'httpd.conf" (usually found at "/usr/local/httpd/conf/httpd.conf" or "/etc/httpd/httpd.conf") to include the "mod_sysync" module: Search for "LoadModule" and append one line after the "LoadModule" directives that are already there:

    ```
    LoadModule sysync_module /usr/local/httpd/libexec/mod_sysync.so
    ```

    NOTE: Take "mod_sysync2.so" when using Apache 2.0.

    Eventually, if your httpd.conf uses ClearModuleList/AddModule, you also need to add a Addmodule directive as well - we also recommed adding after the "AddModule"s already there:

    ```
    AddModule mod_sysync.c
    ```

    NOTE: Take "mod_sysync2.c" when using Apache 2.0.

    Alternatively, you can also have the apache module installer tool apxs to (try to) do these changes automatically - depending on how much the httpd.conf is already "IfDef"fed, this might not always work as expected:

    ```
    $ apxs -i -a /usr/local/httpd/libexec/mod_sysync.so
    ```

    NOTE: Take "apxs2" when using Apache 2.0, under Fedora Core it is still called "apxs".

11. Now Apache should be able to load the mod_sysync.so / mod_sysync2.so when restarted.

This is the basic configuration needed to integrate the SyncML server into Apache httpd. To actually create an URL on your server that serves SyncML, you must create a directory or file and add the appropriate Apache configuration directives to have it handled by mod_sysync.so. There are many ways to do this. The following describes one possibility which is using the <Location> apache directive:

1. Edit your httpd.conf and insert the following directives for mod_sysync. These lines should be inserted into your main server config section, or if you are using virtual servers, into the appropriate virtual server section:

```
<Location /sync>
SetHandler sysync-handler
#SetEnv LD_ASSUME_KERNEL 2.4.1
sysync_Debug Off
sysync_SessionProcess /usr/local/httpd/syncml/bin/sysync_session_odbc
sysync_ConfigFile /usr/local/httpd/syncml/conf/syncserv_odbc.xml
sysync_PipeDir /usr/local/httpd/syncml/pipes
sysync_browserPage /how_to_use_sync_service.html
</Location>
```

   Please refer to the mod_sysync directives reference in chapter 7 on page 43 for details about the directives.
   The above directives will make the SyncML server available under an URL like 'http://your.server.com/sync' (no slash at end!). If accessed with a browser, the SyncML server will internally redirect to the URL specified with 'sysync_browserPage', so the file 'how_to_use_sync_service.html' should exist at the server's root to make this work (if it doesn't SyncML server will still work, but users accessing it with a browser will get an error message).

   Important: The 'SetEnv LD_ASSUME_KERNEL 2.4.1' might be required for older MyODBC drivers. Without this setenv, MyODBC might not find some of the required libraries. So if you encounter problems, try uncommenting the SetEnv directive above.

   Important: Some SuSE distribution contain an error in the apxs2 script: 'APXS=apxs' must be replaced by 'APXS=apxs2'.

2. For first tests, you might want to set sysync_Debug to On, as this will show much diagnostic information in the Apache error log. Please make sure you turn sysync_Debug Off before going productive.

Now Synthesis SyncML Server is ready to work with your Apache server.

5.1.4.3 Multi-Server operation

Running multiple SyncML servers with different configuration files on the same server is as easy as defining different URLs which access the SyncML server with a different 'sysync_ConfigFile' directive in the <Location> directive.

5.1.4.4 Configuring the Apache Version

1. Use a text editor of your choice to open the copied 'syncserv_odbc.xml' file (if you followed the example above, it's full path is '/usr/local/httpd/syncml/conf/syncserv_odbc.xml'). The beginning looks approximately as follows:

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

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

```
<debug>
  <logpath platform="linux">/var/log/sync</logpath>

  (... debug log options …)

</debug>

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

<server type="odbc">

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

  <logfile>/var/log/synclog.txt</logfile>

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

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

  <!-- for MySQL:
  <dbconnectionstring>
    DSN=DataSourceName; UID=dbuser;
    DATABASE=yourdatabase; SOCKET=/tmp/mysql.sock;
  </dbconnectionstring>
  <dbpass>dbpassword</dbpass>
  -->
```

1. Edit the config XML file and adjust the logfile paths: If you created the directories above as recommended, <logpath> will be '/usr/local/httpd/syncml/logs' and <logfile> will be '/usr/local/httpd/syncml/ logs/synclog.txt'.

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 Server uses this path to store diagnostic log files here. Note that the path must be writable for the user under which the server is running (normally called "nobody" or "wwwrun", see 'user' and 'group' directives in 'httpd.conf').
   - <logfile>: Enter a full path to the desired log file and make sure the file can be created there. Sync Server 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.
   - **Important Note:** some MySQL/MyODBC installations do not work correctly when only given the datasource name - for some reason the driver will neither use the database name configured in the datasource, nor eventual SOCKET, SERVER or PORT parameters you might need to connect. Therefore the database name (and probably the SOCKET, SERVER and PORT parameters) must be specified explicitly when connect-

ing the datasource. In this case, you should comment out <datasource> and <dbuser> and use the <dbconnectionstring> tag instead. This is described in detail in the configuration reference manual.

3. Save the config file.

5.1.4.5 Running and stopping the Apache version

Please note that if your Linux distribution is not explicitly supported by us, there might be loading problems on certain distributions due to missing libraries (these can be fixed sometimes by installing appropriate file links into /usr/lib).If the SyncML server does not work on your desired target platform, please include the exact name and version of the Linux distribution used, and if possible, the version of libc/glibc. This helps us to resolve problems.

The Apache version of Synthesis Sync Server is ready whenever Apache httpd is ready.

In **non-multithreaded mode** (<maxthreads> set to 0), a separate process will be started for each sync session, and config will be read for each session. Therefore, changes made in the config file while Apache server is running will affect new SyncML sessions started, but will not affect already started sessions.

In **multithreaded modes** (<maxthreads> set to >0), multiple sessions will be run as threads in the same process, and the process will not terminate when a session finishes. The config file is only read once when the process starts, and will not be reloaded between sessions. Therefore, to apply changes made in the config file, all syncml server processes must be stopped to force reloading the config. To stop a server process, simply terminate it using "kill" shell command.

To make sure your XML configuration file is valid, you can start the 'sysync_session_odbc' with the '-t' special option to test config files:

```
$ /usr/local/httpd/syncml/sysync_session_odbc -t
/usr/local/httpd/syncml/conf/syncserv_odbc.xml

Configuration file syntax is OK.
```

If this tests returns an error code (See Chapter 8 for explanation of error codes), there is a problem in your config file (such as missing license code).

If you still encounter problems running the SyncML server, please consult the Apache server's error log (in our example probably: /usr/local/httpd/logs/error_log). Most common problems are access permission issues.

The httpd server user (normally called "nobody" or "wwwrun", see 'user' and 'group' directives in your 'httpd.conf') must be able to:

- read the config file ('/usr/local/httpd/syncml/conf/syncserv_odbc.xml' in the example above
- read and write to the directory configured with 'sysync_PipeDir' ('/usr/local/httpd/syncml/pipes' in the example above)
- write to the log directories specified with <logpath> and <logfile> in the xml configuration file ('/usr/local/httpd/syncml/logs/' in the example above)
- Access the ODBC libraries (which must therefore be somewhere in the PATH for the httpd server user).

To track down problems, you can also set ' sysync_Debug On' in the '.htaccess' file, this will write a lot of additional diagnostic information into the Apache error_log.

Once the SyncML server is basically running, there might still be configuration related problems (e.g. problems accessing the ODBC database) - in this case please consult the log files in the path specified with <logpath> and <logfile> in the XML config file.

Connecting to the Apache based SyncML Server

The SQL script used to set-up the database automatically installs a test user and a test data folder. So you can connect the SyncML server from any SyncML client with the following connection settings:

Server URL: Of course, the machine you are running the SyncML server on must be accessible from the outside.
Important Note: Some clients (e.g. Siemens S55) do not work properly with servers using a port other than 80 for HTTP.

- Server user name: **test**
- Server password: **test**
- Contacts database path: **contacts** (note that in the 1.0.5 samples, this was called *contact*, singular)
- Events database path (for clients such as T39m-T68i, S55): **events**
- Tasks database path (for clients such as T39m-T68i, S55): **tasks**
- Combined events/tasks (e.g. for 9210 or P800): **calendar**
- Notes database path: **notes**

With these settings, you should be able to perform a sync session. To see error information, look at the log files created (see 5.3).

5.2 Demo Version - Windows, Linux, MacOS X

5.2.1 Installing the demo version

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 Running the demo version

Make sure that no web server is running on the machine you are running the demo version (if there is one, you need to change the port number for Synthesis Sync Server, see below). See Chapter 6 for available command line options.

Then, start the executable:

- in Windows: just doubleclick the 'syncserv_demo.exe' file.
- in Linux: in a shell, change to the directory of the executable and type './syncserv_demo' Call './syncserv_demo_gcc33' to run the newer GCC 3.3 compiled version.
- in Mac OS X: open a terminal window (Applications->Utilities->Terminal), change to the directory of the executable and type './syncserv_demo'

After successful startup, the console window looks like:

```
Synthesis Sync Server 2.1.1.5 (standalone, textdb)

- Demo Version Expiring after 2005-12-31
- Config file read from 'D:\sync\syncserv_demo.xml'
- Debug log path: D:\sync\ (SEPARATE log file)
- starting HTTP server on port 80
```

Now you have a working SyncML server which can synchronize contact and calendar data with any SyncML compliant client device.

If there is an error condition (such as bad config or already used HTTP port) the server does not start up but only displays the error condition. If the error is "***** Terminated due to error: Error selecting protocol" this is normally because the selected port number is in use by another application (e.g. a web server) or because you don't have the privileges to install a TCP/IP server socket (in this case, use sudo to start the server under Linux/Mac OS X). See Chapter 8 for explanation of error codes.

5.2.3 Connecting to the Demo SyncML Server

The sample config file provided with the demo version provides a standard user named "test" having the password "test".
The connection settings in your SyncML client will be:

- Server URL: Of course, the machine you are running the SyncML server on must be accessible from the outside. Note that the demo version does not examine the path in the URL, so only the server name and port number are important. For the demo version, the following URLs are treated the same:

  - **http://www.yourserver.com/this/that/anything.dll**
  - **http://www.yourserver.com/**

  Both refer to the SyncML server which must (in this case) be running on port 80.
  Important Note: Some clients, such as the Nokia 9210 or 7650 devices, do not work correctly with SyncML Server URLs that do not contain a path (sync will abort after the second answering message is sent to the client). So we recommend always using a path (first example above) to avoid these problems. Note also that some devices (e.g. Siemens S55) do not work properly with servers using a port other than 80 for HTTP.
- Server user name: **test**
- Server password: **test**
- Contacts database path: **contacts** (note that in the 1.0.5 samples, this was called *contact*, singular)
- Events database path (for clients such as T39m-T68i, S55): **events**
- Tasks database path (for clients such as T39m-T68i, S55): **tasks**
- Combined events/tasks (e.g. for 9210 or P800): **calendar**
- Notes database path: **notes**

With these settings, you should be able to perform a sync session. The console output for a successful sync session (using a Nokia 9210, contacts) looks like the following:

```
Started new SyncML session (server
id=7291327798616013976)
> SyncML message #1 received from 'IMEI:123456789123'
- Remote alerts (200): two-way normal sync for
  './contacts' (source='./D\System\Data\pretest.cdb')
< SyncML message #1 sent to 'IMEI:123456789123'
> SyncML message #2 received from 'IMEI:123456789123'
- Starting Sync with Datastore './contacts', normal sync
  0 local items are new/changed/deleted for this sync
< SyncML message #2 sent to 'IMEI:123456789123'
> SyncML message #3 received from 'IMEI:123456789123'

- Sync Statistics for Datastore './contact', normal sync
  ================================================
                               on Server   on Client
  Added:                               0           0
  Deleted:                             0           0
  Updated:                             0           0
  SlowSync Matches:                    0
  Server won Conflicts:                0
  Client won Conflicts:                0
  Conflicts with Duplication:          0

< SyncML message #3 sent to 'IMEI:123456789123'
Terminated SyncML session (server id=7291327798616013976)
```

If you want to have a look at the data files produced by the demo server, please refer to the description of <mapfilepath> and <datafilepath> in the config reference manual for details about how data is stored.

5.3 Debug Log Files

Synthesis Sync Server 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>
  <logpath>D:\sync\logs</logpath>
  <enable option="extended"/>
  <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 (extended log information).

In addition, if you have problems with a certain client 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 server.

The files generated are (where "xxxx" is "isapi_odbc" for the ISAPI ODBC version, "xpt_odbc" for the ODBC standalone version, "pipe_odbc" for the Apache version and "demo" for the Demo version).

- **sysync_xxxx.log**: This is a non-debug log file. It logs error conditions that cannot be reported to the SyncML client.
- **xxxx.log**: This is the main debug log file. It contains all the messages that cannot be related to a certain SyncML session.
- **xxxx_sYYYYYYYYY.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.
- **xxxx_reqNNN_incoming.sml** and **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 Server, 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. Standalone SyncML server command line options

The standalone SyncML server 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
-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)
-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.
-h show a short help text for the command line options.

7. mod_sysync configuration directives reference

Installing mod_sysync / mod_sysync2 makes a few extra configuration directives available for use in apache config files (normally ".htaccess" and "httpd.conf"). See 5.1.4.2 for a sample configuration snippet that can be inserted in the a "httpd.conf" file.

All mod_sysync / mod_sysync2 specific directives begin with "sysync_". They are effective only for locations and directories which are handled by "sysync-handler" (by including a "SetHandler sysync-handler" in a <Directory>, <Files> or <Location> apache section)

7.1 sysync_Debug - enable debugging

Syntax: sysync_Debug On|Off
Context: server config, virtual host, directory, .htaccess
Override: Options
Module: mod_sysync

This directive is used to enable or disable debugging. Note that this does not affect debug logging options of the SyncML server itself, but only enables mod_sysync to put some extra information about communication with the session processes into the apache error log file. This might help to track down problems if the SyncML server itself does not write any logfiles (probably because mod_sysync could not start it due to permission problems). Make sure this is set to "Off" (the default) in productive environments.

7.2 sysync_SessionProcess - Path to Session Process Executable

Syntax: sysync_ SessionProcess *path_to_sysync_session_odbc*
Context: server config, virtual host, directory, .htaccess
Override: Options
Module: mod_sysync / mod_sysync2

This directive is used to specify where the SyncML server process executable (which is started by mod_sysync for each sync session in progress) is stored.
NOTE: The module is called "mod_sysync2", when used with Apache2.

Example:

```
sysync_SessionProcess /usr/local/httpd/syncml/bin/sysync_session_odbc
```

7.3 sysync_ConfigFile - Path to XML Server Config File

Syntax: sysync_ ConfigFile *path_to_config_file.xml*
Context: server config, virtual host, directory, .htaccess
Override: Options
Module: mod_sysync / mod_sysync2

This directive is used to specify where the SyncML server reads its configuration file (XML format, see "SySync_config_reference.pdf" file for Details).

Example:

```
sysync_ConfigFile /usr/local/httpd/syncml/conf/syncserv_odbc.xml
```

7.4 sysync_ConfigVars - Define config $(x) variable(s)

Syntax: sysync_ ConfigVars {<*var*>[=<*value*>]}
Context: server config, virtual host, directory, .htaccess
Override: Options
Module: mod_sysync / mod_sysync2

The SyncML server's config file can contain $(x) variables, which can be resolved with this directive. They will be passed directly with option –D to the SyncML server. See "Configuration variables and conditional configuration" in the Synthesis SyncML Config Reference manual for more information.

Example:

```
sysync_ConfigVars logpath=/var/log/sysync x=yyy
```

7.5 sysync_PipeDir - Path to Pipe Directory

Syntax: sysync_ PipeDir *path_to_pipe_directory*
Context: server config, virtual host, directory, .htaccess
Override: Options
Module: mod_sysync / mod_sysync2

The SyncML server needs a directory where it can create pipe files for communication between mod_sysync and the session processes. This directive specifies the location of this directory. Note that the directory must be both writable and readable for the httpd server user (normally "nobody" or "wwwrun").
NOTE: When running under SELinux (Security-Enhanced Linux), e.g. Fedora Core 3, Apache2 is not permitted to create pipes. Either switch off SELinux or set the pipe directories' context attributes accordingly.

Example:

```
sysync_PipeDir /usr/local/httpd/syncml/pipes
```

7.6 sysync_BrowserPage - Page for Browser (GET) accesses

Syntax: sysync_ BrowserPage *server_relative_URL_for_GET_accesses*
Context: server config, virtual host, directory, .htaccess
Override: Options
Module: mod_sysync / mod_sysync2

SyncML requests are always HTTP POST request. If someone uses a web browser to browse the SyncML server URL, this will generate a GET request.

This optional directive can be used to redirect GET requests to a page that will be displayed in the browser (e.g. some help about how to set-up SyncML).

The page should be specified as a location on the same server. Please make sure that you do not redirect to a location that is already served by mod_sysync.

Example:

```
sysync_BrowserPage /help/how_to_use_syncml.html
```

8. Error codes

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

8.1 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

8.2 Internal Error Codes

0	No error
10000..10999	These have the same meaning as the SyncML Status Codes (see 8.1), 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