

independIT Integrative Technologies GmbH
Bergstraße 6
D-86529 Schrobenhausen



BICsuite

Installation Guide Release 2.10

Dieter Stubler

Ronald Jeninga

March 6, 2024

Copyright © 2024 independIT GmbH

Legal notice

This work is copyright protected

Copyright © 2024 independIT Integrative Technologies GmbH

All rights reserved. No part of this work may be reproduced or transmitted in any form or by any means, electronically or mechanically, including photocopying, recording, or by any information storage or retrieval system, without the prior written permission of the copyright owner.

Contents

Table of contents	1
1 Requirements	3
BICsuite Server and Clients	3
Zope Application Server	4
Zope https	4
2 Installation in a Linux environment	5
Installing the BICsuite Server	5
Installing a BICsuite Client	8
Jobserver Sample Installation	10
Scenario	10
Requirements	10
Installation	11
Installation with Postgres	12
Introduction	12
Installation	13
Installation with MySQL	14
Introduction	14
Installation	14
Installation with Ingres	16
Introduction	16
Installation	16
Installation with DB2	17
Introduction	17
Installation	17
Installation with Oracle	19
Introduction	19
Installation	19
Configuration of the TLS/SSL connections	20
Introduction	20
TLS/SSL configuration	21
Installing the Zope server	24
Introduction	24
Installation (Zope2)	24
Installing the HTTPS extensions	29
Installation (Zope4+)	33

Migration of an existing BICsuite!Web Zope2 database to Zope4+ . . .	37
HTTPS using Zope behind an Apache Web Server	37
SSO for BICsuite with Zope	38
Introduction	38
Procedure	39
Kerberos installation and configuration	39
Apache web server and modules	42
Zope extension and configuration	45
Configuring the BICsuite server	54
Settings on the user side	55
Administration of the Zope server	55
3 Installation in a Windows environment	57
Installing the BICsuite Server	57
Introduction	57
Installation	57
Installation with Postgres	61
Introduction	61
Installation	61
Installation with MySQL	63
Introduction	63
Installation	63
Installation with Microsoft™ SQL Server	65
Introduction	65
Installation	65
Installation with Ingres	67
Introduction	67
Installation	67
Installation with DB2	69
Introduction	69
Installation	70
Installation with Oracle Express Edition	71
Introduction	71
Installation	71
Installing the Zope server	73
Introduction	73
Installation (Zope2)	73
Installation (Zope4+)	79

1 Requirements

BICsuite Server and Clients

The following software is required to install the BICsuite Server:

- Oracle(Sun) Java 1.8 SE jre
<http://www.oracle.com/technetwork/java/index.html>
- Microsoft Visual C++ 2010 Redistributable package (x86)
(only for an installation under Windows)
<http://www.microsoft.com>
- One of the following database systems:
 - PostgreSQL
<http://www.postgresql.org>
JDBC for PostgreSQL:
<http://jdbc.postgresql.org>
 - MySQL
<http://www.mysql.com>
MySQL (Connector/J) JDBC Driver
<http://www.mysql.com>
 - Ingres
<http://www.ingres.com>
 - Oracle
<http://www.oracle.com>
 - Microsoft SQL Server
<http://www.microsoft.com>
Microsoft SQL Server JDBC Driver
<http://www.microsoft.com>
- Java Native Access (JNA)
In order to avoid the use of a JNI library we use the JNA library from version 2.6 and later. This library is only required for the Java Jobexecutor.
<https://github.com/twall/jna>

Zope Application Server

The web front end is provided by the Zope Application Server. The following software is required to install the Zope server:

- Python 2.7
<http://www.python.org>
- python-virtualenv
<http://pypi.python.org>
- python-dev
<http://www.python.org>
- python setuptools (Windows only)
<http://pypi.python.org/pypi/setuptools>
- pywin32 (optional, Windows only)
<http://sourceforge.net/projects/pywin32>

Zope https

The following software is also required for using https in combination with Zope:

- SWIG
<http://www.swig.org>
- python M2Crypto package
<http://pypi.python.org>
- openSSL
www.openssl.org
(Or comparable)

2 Installation in a Linux environment

Installing the BICsuite Server

The installation of the BICsuite Scheduling Server is uncomplicated and only requires a few simple actions which are explained in the following.

Where (example) commands are shown, the prompt is usually indicated with `$`. These commands are then executed under the `bicsuite` account, which needs to be created. In some cases, the privileged `root` account is required. This is indicated by a `#` as prompt.

1. Create the user `bicsuite`

It is not necessary to name the user `bicsuite`. This means that the name can also be modified for any convention. In this document it is assumed that the user is called `bicsuite`.

Under the Ubuntu distribution of Linux, a user can be created as follows:

```
# useradd -d /home/bicsuite -m -s /bin/bash -U bicsuite
# passwd bicsuite
```

All the following operations are executed under the user `bicsuite` except where a different user is explicitly stated.

2. Download and install a database management system supported by BICsuite.

BICsuite for Linux currently supports the following systems:

- Postgres (page [12](#))
- MySQL (page [14](#))
- Ingres (page [16](#))
- Oracle (page [19](#))

Reference is made to the appropriate sections regarding the installation of the chosen database system as well as how to modify the configuration of the BICsuite Enterprise Scheduling System.

3. Unpack the software

Unpack the tar archive in the `bicsuite` home directory. For instance:

```
$ tar xvzf bicsuite-2.10.tgz
```

Create a symbolic link:

```
$ ln -s bicsuite-2.10 bicsuite
```

4. Create the configuration

a) User environment

The following variables have to be set to be able to work with the BIC-suite system:

```
BICSUITEHOME=/home/bicsuite/bicsuite
BICSUITECONFIG=/home/bicsuite/etc
PATH=$BICSUITEHOME/bin:$PATH
SWTJAR=/usr/lib/java/swt.jar
JNAJAR=/usr/share/java/jna.jar
```

It has proved to be good practice to save the system configuration in a different folder to the installation directory. This will make it substantially easier to upgrade the system later. Since the variables of all the system's users have to be set, it may be sensible to write the assignments (and exports) to a separate file and then to source this in `.profile` or `.bashrc`.

b) Software environment

Several templates for the configuration files that should be used as a basis for the system configuration can be found under `$BICSUITEHOME/etc`. These have to be copied without the `".template"` extension to the directory `$BICSUITECONFIG`.

For instance:

```
$ cd $BICSUITEHOME/etc; for fff in *.template; do
> TRG=`basename $fff .template`;
> cp $fff $BICSUITECONFIG/$TRG;
> done
```

Afterwards, the files obviously have to be modified to accommodate the environment.

The file `bicsuite.conf` configures some default settings and does not usually have to be modified. However, it may be worth considering running the system logging in a different folder to the installation directory. In this case it is only necessary to change the variable `BICSUITELOGDIR` accordingly. The directory set in `BICSUITELOGDIR` must exist.

The file `java.conf` describes the Java environment that is to be used. In particular, the path to the JDBC driver must be entered. The memory configuration of the server is also regulated here. Even in large-scale environments it is usually only necessary to adapt the variable `BICSUITEMEM`.

The file `server.conf` contains the server configuration. The settings for connecting the BICsuite Scheduling Server to its RDBMS repository have to be modified. More details about this can be found in the respective chapter on the RDBMS that is being used.

In this file it is also necessary to change the `hostname` property to the server's hostname or IP address.

The file `jobserver.conf` is not required here, but serves as a template for the jobserver configuration.

5. Set up the database

Follow the instructions for setting up the database system dependent upon which system you want to use.

For

- Ingres, see page [16](#),
- MySQL, see page [14](#),
- Oracle, see page [19](#), and for
- PostgreSQL, see page [12](#).

6. Start the server

The installation is now more or less complete. You just have to start the server and load the examples as required.

The server can be started with

```
$ server-start
```

7. Create the file `.sdmshrc`

The file `.sdmshrc`, if present, is read by all the BICsuite command line tools for populating the command line parameters. In the following it is assumed that this file exists and that the correct values have been set for the users, password, host and port. The file `.sdmshrc` is created in the Linux user's home directory.

Here is an example of the contents:

```
$ cat ~/.sdmshrc
User=SYSTEM
Password=G0H0ME
Host=localhost
Port=2506
Timeout=0
```

Important: Since the file contains the data for accessing the Scheduling Server, the file rights should be set so that only its owner can read the file.

```
$ chmod 600 ~/.sdmshrc
$ ls -lG ~/.sdmshrc
-rw----- 1 bicsuite 73 2011-11-09 09:28 /home/bicsuite/.sdmshrc
```

8. Install the convenience package

The convenience package installs a commonly used configuration for an exit state model.

```
$ sdmsh < $BICSUITEHOME/install/convenience.sdms
```

9. Install the examples (optionally)

The installation routine for the examples comprises two parts. First of all, three so-called jobserver that are required for the subsequent workflow definitions are created. Sample workflow definitions are then loaded onto the server.

a) Create the jobserver

It just takes one script to create the jobserver:

```
$ cd $BICSUITEHOME/install
$ setup_example_jobserver.sh
```

b) Load the workflow definitions

The following commands are entered to load the workflow definitions:

```
$ cd $BICSUITEHOME/install
$ sdmsh < setup_examples.sdms
```

Because the examples assume that the jobserver have already been created, the above sequence is mandatory.

Installing a BICsuite Client

The installation of a BICsuite scheduling client is easy and requires only a few simple actions which are explained in the following.

Where (example) commands are shown, the prompt is usually indicated with \$. These commands are then executed under the `bicsuite` account, which needs to be created. In some cases, the privileged `root` account is required. This is indicated by a # as prompt.

1. Creating the user `bicsuite`

It is not necessary to name the user `bicsuite`. This means that the name can also be modified for any convention. In this document it is assumed that the user is called `bicsuite`.

Under the Ubuntu distribution of Linux, a user can be created as follows:

```
# useradd -d /home/bicsuite -m -s /bin/bash -U bicsuite
# passwd bicsuite
```

All the following operations are executed under the user `bicsuite` except where a different user is explicitly stated.

2. Unpack the software

Unpack the tar archive in the `bicsuite` home directory. For instance:

```
$ tar xvzf bicsuite-2.10.tgz
```

Create a symbolic link:

```
$ ln -s bicsuite-2.10 bicsuite
```

3. Create the configuration

a) User environment

The following variables have to be set to be able to work with the BIC-suite system:

```
BICSUITEHOME=/home/bicsuite/bicsuite
BICSUITECONFIG=/home/bicsuite/etc
PATH=$BICSUITEHOME/bin:$PATH
SWTJAR=/usr/lib/java/swt.jar
JNAJAR=/usr/share/java/jna.jar
```

It has proved to be good practice to save the system configuration in a different folder to the installation directory. This will make it substantially easier to upgrade the system later. Since the variables of all the system's users have to be set, it may be sensible to write the assignments (and exports) to a separate file and then to source this in `.profile` or `.bashrc`.

b) Software environment

Several templates for the configuration files that should be used as a basis for the system configuration can be found under `$BICSUITEHOME/etc`.

For a client installation we need the files `bicsuite.conf` and `java.conf`. These have to be copied without the `".template"` extension to the directory `$BICSUITECONFIG`.

```
$ cp $BICSUITEHOME/etc/bicsuite.conf.template \
    $BICSUITECONFIG/bicsuite.conf
$ cp $BICSUITEHOME/etc/java.conf.template \
    $BICSUITECONFIG/java.conf
```

The file `bicsuite.conf` configures some default settings and does not usually have to be modified.

The file `java.conf` describes the Java environment that is to be used and usually does not require any changes.

4. Create the file `.sdmshrc`

The file `.sdmshrc`, if present, is read by all the BICsuite command line tools for populating the command line parameters. In the following it is assumed that this file exists and that the correct values have been set for the users, password, host and port. The file `.sdmshrc` is created in the Linux user's home directory.

Here is an example of the contents:

```
$ cat ~/.sdmshrc
User=SYSTEM
Password=G0H0ME
Host=localhost
Port=2506
Timeout=0
```

Important: Since the file contains the data for accessing the Scheduling Server, the file rights should be set so that only its owner can read the file.

```
$ chmod 600 ~/.sdmshrc
$ ls -lG ~/.sdmshrc
-rw----- 1 bicsuite 73 2011-11-09 09:28 /home/bicsuite/.sdmshrc
```

Jobserver Sample Installation

In the following a sample installation of a BICsuite jobserver will be shown.

Scenario

On the machine `machine_42` a jobserver shall execute processes as user `arthur`. The HOME directory of the user is `/home/arthur`. The BICsuite server is installed on the machine `scheduling_server` and is listening on port 2506. The system password is `G0H0ME`.

Requirements

On the machine `machine_42`, a BICsuite client was installed in the HOME directory `/home/bicsuite`.

The user `arthur` has the following access privileges on the client installation files:

- Read access on the files `java.conf` and `bicsuite.conf` in `$BICSUITECONFIG` and all files under `/home/bicsuite/lib`
- Read and execute privileges on all files in `/home/bicsuite/bin`

Installation

To enable a jobserver to connect to the BICsuite scheduling server, the jobserver has to be configured within BICsuite scheduling server. In the following we execute the necessary steps using the `sdmsh` command line utility. Alternatively this could also be done using the Web GUI.

1. Login as user `arthur` into the machine `machine_42`
2. Setting of the shell environment variables (`.bashrc`).

```
export BICSUITEHOME=/home/bicsuite/bicsuite
export BICSUITECONFIG=/home/bicsuite/etc
export PATH=$BICSUITEHOME/bin:$PATH
```

3. Testing the environment

```
sdmsh --host localhost --port 2506 --user SYSTEM --pass G0H0ME
```

A `SDMS>` prompt should appear (use `'exit'` to close `sdmsh`).

4. Create directories

```
cd $HOME
mkdir etc
mkdir taskfiles
mkdir work
mkdir log
```

5. Create a Scope for machine `machine_42` using `sdmsh`

```
SDMS> CREATE OR ALTER SCOPE GLOBAL.'MACHINE_42'
WITH
  CONFIG = (
    'JOBEXECUTOR' = '/home/bicsuite/bicsuite/bin/jobserver',
    'HTTPHOST' = 'machine_42'
  );
```

All directory paths must be full qualified. The use of shell environment variables is not supported.

6. Create a jobserver using `sdmsh`

```
SDMS> CREATE OR ALTER JOB SERVER GLOBAL.'MACHINE_42'.'ARTHUR'
WITH
  PASSWORD = 'dent',
  NODE = 'machine_42',
  CONFIG = (
```

```
'JOBFILEPREFIX' = '/home/arthur/taskfiles/',
'DEFAULTWORKDIR' = '/home/arthur/work',
'HTTPPORT' = '8905',
'NAME_PATTERN_LOGFILES' = '/home/arthur/work/.*\\.log'
);
```

The HTTPPORT is necessary for the display of job logs and can be selected freely but must be unique for all jobserver on the same machine. All directory paths must be full qualified. The use of shell environment variables is not supported.

7. Create the Named Resource for the jobserver using sdmsh

```
SDMS> CREATE OR ALTER NAMED RESOURCE RESOURCE.'JOBSERVERS'
      WITH USAGE = CATEGORY;
SDMS> CREATE NAMED RESOURCE RESOURCE.'JOBSERVERS'.'ARTHUR@MACHINE_42'
      WITH USAGE = STATIC;
```

8. Create an Environment for the jobserver using sdmsh

```
SDMS> CREATE ENVIRONMENT 'ARTHUR@MACHINE_42'
      WITH RESOURCE = (RESOURCE.'JOBSERVERS'.'ARTHUR@MACHINE_42');
```

9. Create the Resource in the jobserver using sdmsh

```
SDMS> CREATE RESOURCE RESOURCE.'JOBSERVERS'.'ARTHUR@MACHINE_42'
      IN GLOBAL.'MACHINE_42'.'ARTHUR' WITH ONLINE;
```

10. Create the configuration file `$HOME/etc/jobserver.conf` for the jobserver containing the following:

```
RepoHost= scheduling_server
RepoPort= 2506
RepoUser= "GLOBAL.'MACHINE_42'.'ARTHUR'"
RepoPass= dent
```

11. Start the jobserver

```
jobserver-run $HOME/etc/jobserver.conf $HOME/log/jobserver.out
```

If the jobserver should start automatically at boot time, the system administrator has to configure this accordingly.

Installation with Postgres

Introduction

This guide does not claim to precisely describe the installation of the database system. Details about this can be found in the Postgres documentation. Normally,

though, this guide should be adequate for performing a "standard" installation.

Installation

1. Download and install the latest version of Postgres

A Postgres package is normally provided for every Linux distribution. This package, as well as a package for the JDBC driver for Postgres, should usually be easy to install.

2. Configure the file `pg_hba.conf`

To enable the BICsuite Scheduling Server to identify itself to PostgreSQL with a user and password, the following line has to be added to the Postgres configuration file `pg_hba.conf`. This file is usually found under `/var/lib/pgsql/<version>/data:`

```
host          all          all          127.0.0.1/32          md5
```

PostgreSQL then has to be restarted.

3. Create the Postgres user `bicsuite`

Log on as the Postgres user and run the command `createuser` as shown in the example (version 8):

```
$ createuser -P bicsuite
Enter password for new role:
Enter it again:
Shall the new role be a superuser? (y/n): n
Shall the new role be allowed to create databases? (y/n): y
Shall the new role be allowed to create more new roles? (y/n): n
```

or, in the case of version 9:

```
$ createuser -P -d bicsuite
```

The entered password will be required again later.

4. Create the repository database `bicsuitedb`

When you are logged on as the user `bicsuite`, create the database for the repository as shown in the following example:

```
$ createdb bicsuitedb
```

5. Create and initialise the database tables

To create the required database schema, switch to the `bicsuite sql` directory and call the Postgres utility `psql` as shown in the following example:

```
$ cd $BICSUITEHOME/sql
$ psql -f pg/install.sql bicsuitedb
```

6. Configure the database connection in the BICsuite Server configuration file

`$BICSUITECONFIG/server.conf`

Change the following properties as shown here:

```
DbPasswd=bicsuite password
DbUrl=jdbc:postgresql:bicsuitedb
DbUser=bicsuite
JdbcDriver=org.postgresql.Driver
```

The `DbUrl` is to a certain degree dependent upon which version of PostgreSQL is installed. Under Version 8 this is

```
DbUrl=jdbc:postgresql:bicsuitedb
```

7. Configure the BICsuite Java Class Path for the Postgres JDBC

Now all you have to do is to append the path to the Postgres JDBC to `CLASSPATH` in the configuration file `$BICSUITECONFIG/java.conf`.

For instance:

```
BICSUITECLASSPATH=$BICSUITEJAR:/usr/share/java/postgresql-jdbc4-9.2.jar
```

Installation with MySQL

Introduction

This guide does not claim to precisely describe the installation of the database system. Details about this can be found in the MySQL documentation. Normally, though, this guide should be adequate for performing a "standard" installation.

Installation

1. Download and install the latest version of MySQL.

Ready-to-use MySQL packages are available for most Linux distributions. These can be simply installed using the appropriate tools.

During this installation, you will be prompted to enter a password for the MySQL root user (not to be confused with the Linux root user). This password is required again in the next step.

Because BICsuite sets up a JDBC connection to access the database, the MySQL JDBC driver has to be installed as well.

2. Create the MySQL user `bicsuite` and the database `bicsuitedb`

Start the Utility `mysql` utility and log on as the MySQL root user to create the user `bicsuite` and the database `bicsuitedb`:


```

$ mysql --user=root --password=mysql-root-password

Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 41
Server version: 5.1.54-1ubuntu4 (Ubuntu)

Copyright (c) 2000, 2010, Oracle and/or its affiliates. All rights reserved.
This software comes with ABSOLUTELY NO WARRANTY. This is free software,
and you are welcome to modify and redistribute it under the GPL v2 license

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql> create user bicsuite identified by 'bicsuite_password';
Query OK, 0 rows affected (0.01 sec)

mysql> create database bicsuitedb;
Query OK, 1 row affected (0.00 sec)

mysql> grant all on bicsuitedb.* to bicsuite;
Query OK, 0 rows affected (0.00 sec)

mysql> quit
Bye

```

3. Create and initialise the database tables

Run the following commands:

```

$ cd $BICSUITEHOME/sql
$ mysql --user=bicsuite --password=bicsuite_password
  --database=bicsuitedb --execute="source mysql/install.sql"

```

4. Configure the database connection in the \$BICSUITECONFIG/server.conf configuration file

Change the following properties as shown here:

```

DbPasswd=bicsuite_password
DbUrl=jdbc:mysql:///bicsuitedb
DbUser=bicsuite
JdbcDriver=com.mysql.jdbc.Driver

```

5. Configure the BICsuite Java Class Path for the MySQL JDBC

Now all you have to do is to append the path to the MySQL JDBC to CLASSPATH in the configuration file \$BICSUITECONFIG/java.conf.

For instance

```

BICSUITECLASSPATH=$BICSUITEJAR:/usr/share/java/mysql-connector-java.jar

```

Installation with Ingres

Introduction

This guide does not claim to precisely describe the installation of the database system. Details about this can be found in the Ingres documentation. Normally, though, this guide should be adequate for performing a "standard" installation.

Installation

1. Install Ingres

We assume that the Ingres system will be installed under the user `ingres`. The installation identifier is taken here as being `II`, which is the default value.

2. Create the user `bicsuite`

There are two ways of registering the user `bicsuite` in the Ingres system. The first method involves creating the user with the help of the tool `accessdb`. This method is not explained here any further.

The second method is to create the user using SQL commands. To do this, start the SQL Terminal Monitor as the user `ingres`:

```
$ su - ingres
Password:
ingres@cheetah:~$ sql iidbdb
INGRES TERMINAL MONITOR Copyright 2008 Ingres Corporation
Ingres Linux Version II 9.2.1 (a64.lnx/103)NPTL login
Mon Jun 13 10:05:19 2011
```

```
continue
* create user bicsuite with privileges = (createdb);
* \g
Executing . . .
```

```
continue
* commit;\g
Executing . . .
```

```
continue
* \q
Ingres Version II 9.2.1 (a64.lnx/103)NPTL logout
Mon Jun 13 10:07:58 2011
ingres@cheetah:~$
```

3. Create the repository database `bicsuitedb`

```
$ $II_SYSTEM/ingres/bin/createdb bicsuitedb
Creating database 'bicsuitedb' . . .
```

```
Creating DBMS System Catalogs . . .
Modifying DBMS System Catalogs . . .
Creating Standard Catalog Interface . . .
Creating Front-end System Catalogs . . .
```

Creation of database 'bicsuitedb' completed successfully.

4. Create and initialise the database tables

Run the following commands to create the required tables:

```
$ cd $BICSUITEHOME/sql
$ sql bicsuitedb < ing\install.sql
```

5. Configure the database connection in the BICsuite Server configuration file

`$BICSUITECONFIG/server.conf`

Change the following properties as shown here:

```
DbPasswd=<bicsuite OS password>
DbUrl=jdbc:ingres://localhost:II7/bicsuitedb;
DbUser=bicsuite
JdbcDriver=com.ingres.jdbc.IngresDriver
```

6. Configure the BICsuite Java Class Path for the Ingres JDBC

Now all you have to do is to append the path to the Ingres JDBC to `CLASSPATH` in the configuration file `$BICSUITECONFIG/java.conf`.

For instance:

```
BICSUITECLASSPATH=$BICSUITEJAR:$II_SYSTEM/ingres/lib/iijdbc.jar
```

Installation with DB2

Introduction

This guide does not claim to precisely describe the installation of the database system. Details about this can be found in the DB2 documentation. Normally, though, this guide should be adequate for performing a "standard" installation.

Installation

1. Installation of DB2

We assume that a DB2 instance is installed under user `db2inst`. The listen port is configured to be 50000. In case this is different in the target environment this should be considered in the configuration of the `DbUrl`.

2. Create the repository database bicsuite

The instance owner db2inst creates the repository database bicsuite and grants all required privileges to the user bicsuite. The database name can be any arbitrary name, not longer than 8 characters.

```
[db2inst@cheetah bin]$ db2
(c) Copyright IBM Corporation 1993,2007
Command Line Processor for DB2 Client 10.5.5
```

You can issue database manager commands and SQL statements from the command prompt. For example:

```
db2 => connect to sample
db2 => bind sample.bnd
```

For general help, type: ?.

For command help, type: ? command, where command can be the first few keywords of a database manager command. For example:

```
? CATALOG DATABASE for help on the CATALOG DATABASE command
? CATALOG           for help on all of the CATALOG commands.
```

To exit db2 interactive mode, type QUIT at the command prompt. Outside interactive mode, all commands must be prefixed with 'db2'.

To list the current command option settings, type LIST COMMAND OPTIONS.

For more detailed help, refer to the Online Reference Manual.

```
db2 => create database bicsuite restrictive
DB20000I The CREATE DATABASE command completed successfully.
db2 => connect to bicsuite
```

Database Connection Information

```
Database server      = DB2/LINUX8664 10.5.5
SQL authorization ID = DB2INST
Local database alias = BICSUITE
```

```
db2 => grant dbadm, dataaccess, accessctrl, secadm on database to
      user bicsuite
DB20000I The SQL command completed successfully.
```

3. Create and initialise the database tables

Run the following commands to create the required tables:

```
[bicsuite@ocelot ~]$ cd $BICSUITEHOME/sql
[bicsuite@ocelot sql]$ . /home/db2inst/.ibm/db2/desktop/env
[bicsuite@ocelot sql]$ export PATH=/home/db2inst/sqllib/bin:$PATH
[bicsuite@ocelot sql]$ clpplus -nw \
      bicsuite/'<bicsuite OS password>'@localhost/bicsuite \
      @db2/install.sql > /tmp/install.log
```

4. Configure the database connection in the BICsuite Server configuration file

\$BICSUITECONFIG/server.conf

Change the following properties as shown here:

```
DbPasswd=<bicsuite OS password>
DbUrl=jdbc:db2://localhost:50000/bicsuite
DbUser=bicsuite
JdbcDriver=com.ibm.db2.jcc.DB2Driver
```

5. Configure the BICSuite Java Class Path for the DB2 JDBC

In der Konfigurationsdatei `$BICSUITECONFIG/java.conf` muss nun nur noch der Pfad zum IBM DB2 JDBC-Treiber an dem `CLASSPATH` angehängt werden.

Now all you have to do is to append the path to the IBM DB2 JDBC driver to `CLASSPATH` in the configuration file `$BICSUITECONFIG/java.conf`.

For instance:

```
DB2JAVA=/home/db2inst/sqllib/java
BICSUITEJDBC=$DB2JAVA/db2jcc4.jar
BICSUITEJDBC=$BICSUITEJDBC:$DB2JAVA/db2jcc.jar
BICSUITEJDBC=$BICSUITEJDBC:$DB2JAVA/db2jcc_license_cu.jar
BICSUITECLASSPATH=$BICSUITEJAR:$BICSUITEJDBC
unset DB2JAVA
```

Installation with Oracle

Introduction

This guide does not claim to precisely describe the installation of the database system. Details about this can be found in the Oracle documentation. Normally, though, this guide should be adequate for performing a "standard" installation.

Installation

1. Download and install the latest version of Oracle Express Edition.

Remember your Oracle system password.

2. Create the Oracle user `bicsuite`

```
$ sqlplus SYSTEM/oracle_system_password
```

Create the user `bicsuite` with:

```
SQL> create user bicsuite identified by bicsuite_password;
```

Assign the required access privileges:

```
SQL> grant CONNECT, RESOURCE, CREATE VIEW, CREATE PROCEDURE
      TO bicsuite;
```

3. Create and initialise the database tables

```
$ cd $BICSUITEHOME/sql
$ sqlplus bicsuite/bicsuite_password @ora/install.sql
```

4. Configure the database connection in the BICsuite Server configuration file

Edit `$BICSUITECONFIG/server.conf` and modify the following properties:

```
DbPasswd=bicsuite_password
DbUrl=jdbc:oracle:thin:@hostname:1521:XE
DbUser=bicsuite
JdbcDriver=oracle.jdbc.OracleDriver
```

Here it is also necessary to change the hostname to the server's hostname or IP address.

5. Configure the BICsuite Java Class Path for the Oracle JDBC

Now all you have to do is to append the path to the Oracle JDBC driver to `CLASSPATH` in the configuration file `$BICSUITECONFIG/java.conf`.

For instance:

```
BICSUITECLASSPATH=$BICSUITEJAR:$ORACLE_HOME/jdbc/lib/ojdbc14.jar
```

Configuration of the TLS/SSL connections

Introduction

Configuring encrypted communication within the BICsuite system is relatively easy. However, the required time and effort will vary depending upon your requirements.

The following options are available:

1. BICsuite without SSL/TLS communication
2. BICsuite with and without SSL/TLS communication
3. BICsuite exclusively with SSL/TLS communication

In turn, there are two methods of communicating with TLS/SSL:

1. Server-side authentication only this means that clients can verify whether the server with which they are communicating really is trustworthy. This setting just requires one key pair for the server. All communication is encrypted.
2. Server-side and client-side authentication With this configuration, both parties verify whether the identity of the communication partner is known to them. This setting is extremely secure but also more time-consuming since a key pair has to be generated for each client and obviously for the server as well. All communication is naturally encrypted.

TLS/SSL configuration

Configuring the SSL/TLS communication essentially requires just a few steps.

1. Generate the key pairs for the server

The utility `keytool`, a component of Java (SE), is used to generate the key pairs. More information about this utility can be found at

<http://download.oracle.com/javase/6/docs/technotes/tools/windows/keytool.html>

The following procedure should work (for non-specialists):

a) Create a directory, e.g.

```
$ mkdir $BICSUITECONFIG/certs
```

b) This directory will afterwards contain the private key for the server and should therefore be protected accordingly, e.g.

```
$ chmod 700 $BICSUITECONFIG/certs
```

c) Generate a key pair using `keytool`, e.g.

```
$ keytool -genkeypair -alias bicsuite -keypass secret \  
> -dname "cn=servername, ou=BICSuite, o=independIT, c=DE" \  
> -keystore $BICSUITECONFIG/certs/svrkeystore \  
> -storepass secret -validity 365
```

IMPORTANT: Both passwords (keypass and storepass) *must* be identical!

d) The server's public key must later be given to the clients. To do this, it first has to be extracted from the key file.

```
$ keytool -export -alias bicsuite -file svrkey \  
> -keystore $BICSUITECONFIG/certs/svrkeystore
```

The storepass (secret) is required here.

2. Modify the server configuration

The server configuration now has to be modified to notify the server about its keys. You also have to decide whether you want to use just server-side authentication or authentication by both the server and the client. A decision also has to be made with regard to non-encrypted communication.

```
#  
# SSLPort: port for encrypted communication  
# (inactive = 0)  
#  
SSLPort=2507  
#  
# KeyStore defines which key file is to be used  
# $BICSUITEHOME/etc/certs/svrkeystore
```

```

#
KeyStore=/home/bicsuite/etc/certs/avrkeystore
#
# The KeyStorePassword is required to read out the key store
#
KeyStorePassword=secret
#
# TrustStore defines which file contains the (public) keys of the
# valid communication partners
#
TrustStore=/home/bicsuite/etc/certs/avrkeystore
#
# The TrustStorePassword is required to read out the trust store
#
TrustStorePassword=secret
#
# ClientAuthentication defines whether clients have to authenticate
# themselves (true) or not (false)
#
ClientAuthentication=true
#
# Port defines the port for non-encrypted communication
# (inactive = 0)
# If all ports are configured as being inactive,
# port 2506 is opened for non-encrypted communication
#
Port=2506
#
# ServicePort defines the port for accessing the system in
# emergencies (inactive = 0)
#
ServicePort=2505

```

If Client Authentication=true, the server requires both the key store and the trust store. If client authentication is not necessary, only the key store is required.

3. Client configuration

If no client authentication is necessary, clients only need access to a trust store to be able to verify the server's identity. If the identity of the clients has to be verified as well, a key store will also need to be generated for each client. This is done analogue to the creation of the key store for the server.

4. Modify .sdmshrc

The use of an "ini" file is mandatory if sdmsh or the standard utilities are to communicate with the server via TLS/SSL. There are three possibilities here:

- a) \$BICSUITECONFIG/sdmshrc
- b) \$HOME/.sdmshrc
- c) A file specified when the utility is called

The information required for the secure connection is the decisive factor here.
For instance:

```
User=donald
Password=duck
Host=localhost
Port=2507
SSL=true
KeyStore=/home/bicsuite/etc/certs/clntkeystore
TrustStore=/home/bicsuite/etc/certs/clnttruststore
KeyStorePassword=secret
TrustStorePassword=secret
Timeout=0
```

This permits symmetrical authentication. If only server-side authentication is necessary, the lines concerning the key store can be deleted.

5. Jobserver configuration

The jobservers now have four new configuration parameters corresponding to the server parameters:

```
KEYSTORE
TRUSTSTORE
KEYSTOREPASSWORD
TRUSTSTOREPASSWORD
```

These are supplemented by another parameter which states whether encrypted communication is desired:

```
USE_SSL
```

If client authentication is to be used, a key pair has to be created for each jobserver. This is done in precisely the same way as has already been described for the BICsuite Server.

You must obviously make sure that the configuration parameter `RepoPort` in particular is set correctly.

6. Getting it all together

Last, but not least, the public keys now have to be swapped between the individual communication partners using `keytool`. To define the server's public key as being trustworthy, for instance, it is added to the client's trust store. For example:

```
$ keytool -import -keystore clntkeystore -alias bicsuite -file svrkey
Enter keystore password:
Owner: CN=servername, OU=BICsuite, O=independIT, C=DE
Issuer: CN=servername, OU=BICsuite, O=independIT, C=DE
Serial number: 4dc28814
Valid from: Thu May 05 13:20:52 2011 until: Fri May 04 13:20:52 2012
```

```
Certificate fingerprints:
  MD5:  D3:83:F2:2B:93:2B:65:7A:41:3E:CE:2E:C8:EC:40:62
  SHA1: AF:B1:18:95:B2:2A:BB:1D:08:BD:A6:87:68:64:6B:FC:0D:A8:30:DA
  Signature algorithm name: SHA1withDSA
  Version: 3
Trust this certificate? [no]: yes
Certificate was added to keystore
```

A storepass is naturally required to do this.

If client authentication is required as well, the client certificates will also need to be added to the server's trust store.

Installing the Zope server

Introduction

A Zope Application Server has to be installed before you can use the BICsuite!Web userinterface.

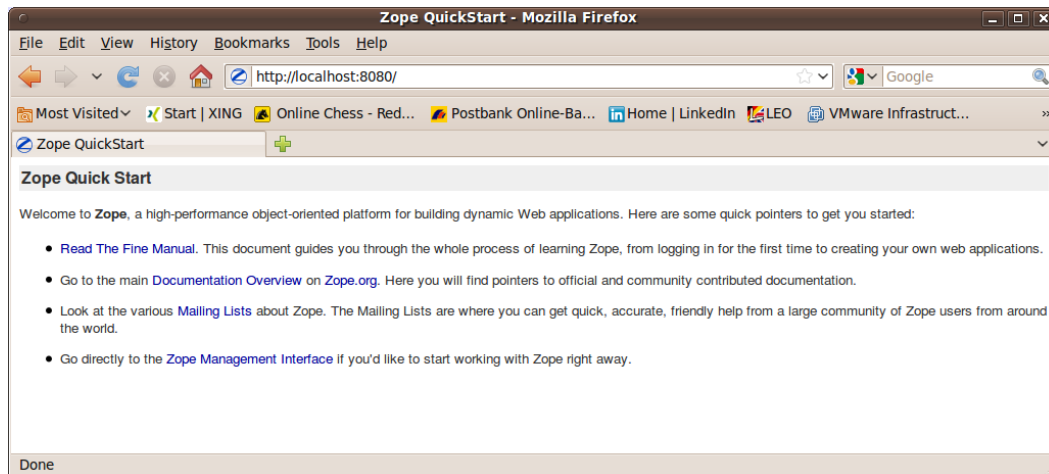


Figure 2.1: Zope Quick Start page

Installation (Zope2)

1. Install virtualenv

```
$ easy_install virtualenv
```

2. Create a virtual Python environment for the Zope installation

```
$ mkdir $HOME/software
$ cd $HOME/software
$ virtualenv --no-site-packages Zope
```

3. Install the Zope2 software

Install the latest release of Zope2. At the time this document was written, 2.13.29 was the most recent release.

```
$ cd $HOME/software/Zope
$ bin/pip install -r \
https://raw.githubusercontent.com/zoepfoundation/Zope/2.13.29/requirements.txt
```

If internet access is not available during the installation, Zope can also be installed offline. To do this, proceed as follows:

a) Download python packages

On an identical as possible system with internet access, execute the following commands:

```
$ wget \
https://raw.githubusercontent.com/zoepfoundation/Zope/2.13.29/requirements.txt
$ pip download -r requirements.txt -d packages
```

b) Transferring files to the target system

The file 'requirements.txt' and the directory 'packages' now have to be transferred to the target system without internet access. Place the files in the directory \$HOME/software.

c) Installation on the target system

The following command installs Zope from the downloaded files:

```
$ cd $HOME/software
$ Zope/bin/pip install --no-index --use-wheel \
--find-links=./packages -r requirements.txt
```

4. Create a Zope instance for BICsuite!Web user interface

```
$ cd $HOME/software/Zope
$ bin/mkzopeinstance -d $HOME/bicsuiteweb -u sdmsadm:sdmsadm_password
```

You can use any password of your choice. This will be required again later again. The username must be sdmsadm though.

The Zope server is started briefly for testing:

```
$ $HOME/bicsuiteweb/bin/zopectl start
```

In the web browser, the URL

```
http://localhost:8080
```

should now be displayed on the Zope QuickStart page as in Figure 2.1.

The Zope instance is now stopped again:

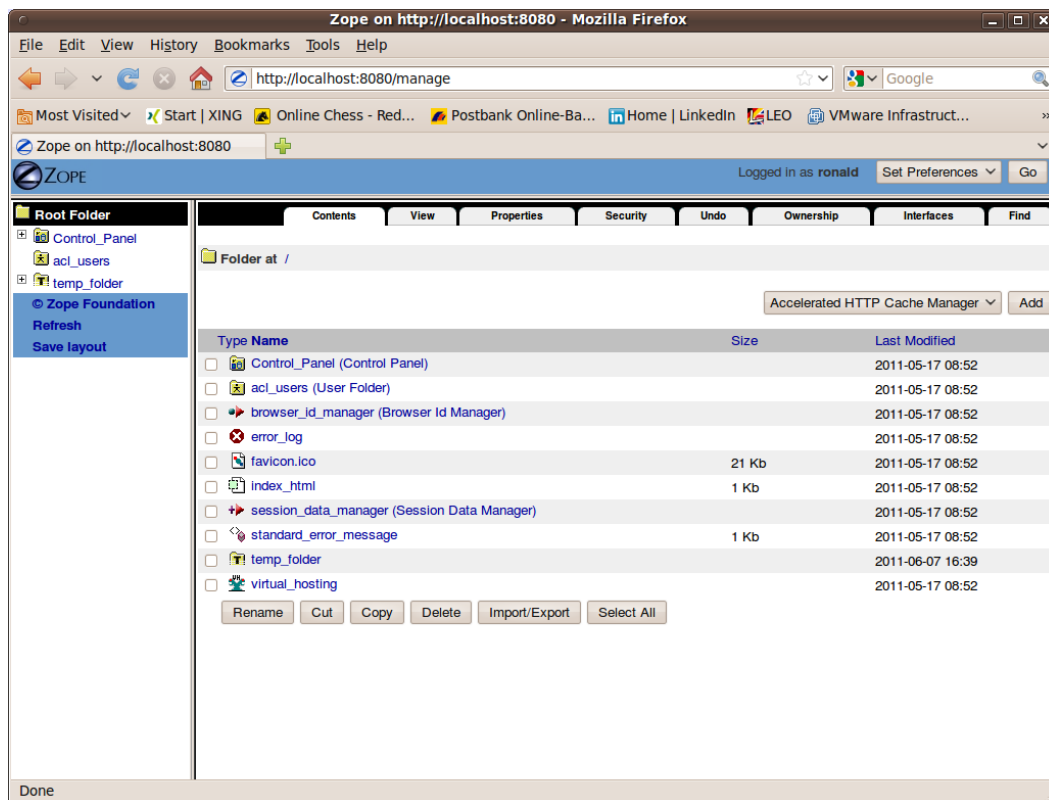


Figure 2.2: Zope Management screen

```
$ $HOME/bicsuiteweb/bin/zopectl stop
```

5. Install the BICsuite!Web components

The Zope installation has to be extended with several modules before the BICsuite!Web components can be installed.

```
$ cd $HOME/bicsuiteweb
$ mkdir Extensions
$ cd Extensions
$ ln -s $HOME/bicsuite/zope/*.py .
$ cd ../Products
$ ln -s $HOME/bicsuite/zope/BICsuiteSubmitMemory .
$ cd ../import
$ ln -s $HOME/bicsuite/zope/SDMS.zexp .
```

The Zope instance now has to be started again to register the changes on the Zope side.

```
$ $HOME/bicsuiteweb/bin/zopectl start
```

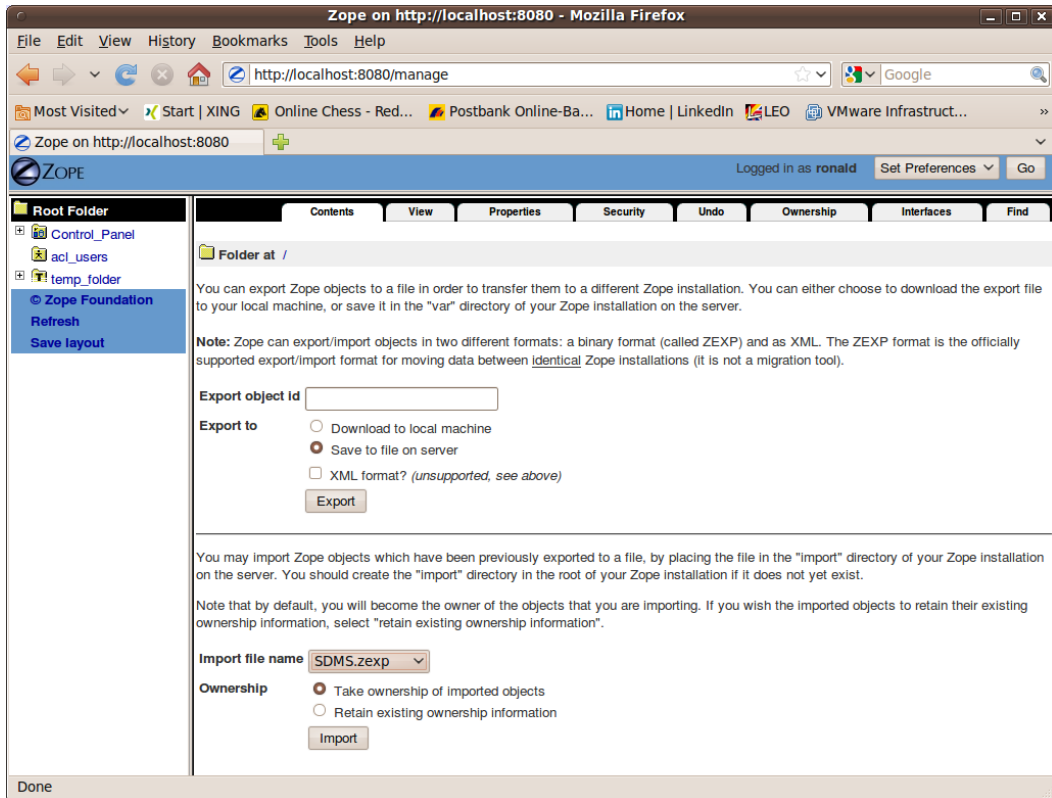


Figure 2.3: Zope Import dialogue

The Zope Management user interface is now opened at the address

`http://localhost:8080/manage`

in a browser (see also Figure 2.2). This is done with the user `sdmsadm` together with the password you have assigned.

The front end software is now loaded into Zope (Import button, see Figure 2.3):

- a) Import into the folder / `SDMS.zexp`.
- b) In the folder `/SDMS/Install`, mark and copy the folders `User` and `Custom`.
- c) Create the folders `User` and `Custom` in the folder / by pasting them.

If everything has been carried out without any errors, the interface will look like the screenshot in Figure 2.4.

6. Configure the server connections

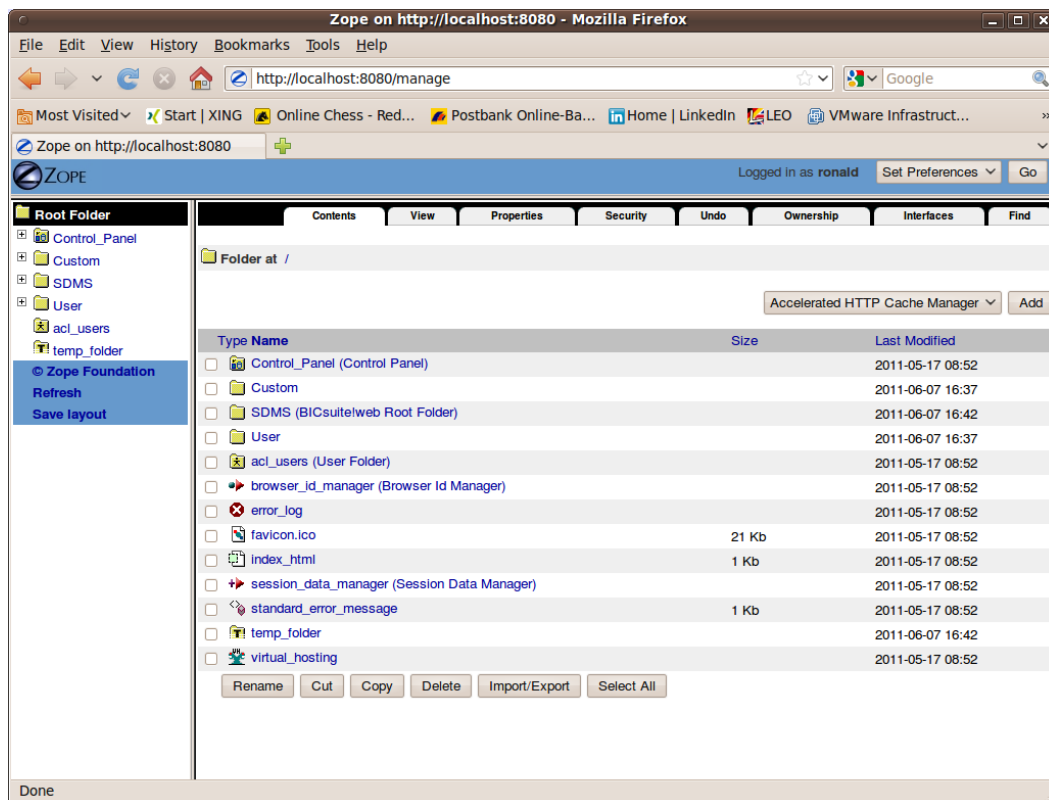


Figure 2.4: Zope result window

The server connections are also configured in the Zope Management user interface. To do this, log on as the user `sdmsadm`.

The `SDMSServers` Python script is edited in the folder `Custom`. This script delivers a dictionary which has to contain an entry for every BICSuite Server that is to be addressed by this BICSuite!Web installation. The entry looks like this:

```
# Server name that identifies the server in the BICSuite!Web
# user interface
'servername' : {
    # TCP/IP hostname or address at which the BICSuite Server is running
    'HOST'      : 'hostname',

    # Port at which the BICSuite Server is addressed
    'PORT'      : '2506',

    # BASIC, PROFESSIONAL, ENTERPRISE
    'VERSION'   : 'BASIC',
```

```

# Optional property stating whether BICsuite!Web should
# cache the server connections
'CACHE'      : 'Y'

# Optional property stating for how long cached BICsuite!Web
# server connections should continue to be valid
# Default setting is 60 seconds, only significant if 'CACHE' : 'Y'
'TIMEOUT'    : '60'
}

```

An entry with the name `DEFAULT` must exist for bootstrapping. This entry can be deleted after the user has been set up (who obviously should not then use this connection).

If a server is to be addressed via a secure SSL connection, the following additional properties have to be defined as well:

```

# Connection is set up via Secure Socket Layer
'SSL'        : 'true',

# If stated, the identity of the BICsuite! Server is verified
# The stated file must contain the BICsuite!Server server certificate
'TRUSTSTORE' : 'truststore.pem',

# If the BICsuite!Server requires client authentication,
# this property must be defined and the stated file
# must contain the client's certificate and private key.
# The certificate must be known to the server in its trust store.
'KEYSTORE'   : 'keystore.pem'

```

Note:

When using SSL, for performance reasons it is advisable to use cached server connections because setting up a secure connection is a resource-intensive operation.

7. Open the BICsuite!Web interface

The user interface is now available at the address

`http://localhost:8080/SDMS`

A logon prompt is displayed when this page is opened. When the user has logged on, the application is started by clicking the "Take Off" button.

How to work with the interface is described in the relevant documentation.

Installing the HTTPS extensions

Installing the Zope HTTPS extension will take some time. It is not particularly complicated, but it is important to carefully and precisely follow the instructions and ideally to understand them as well.

1. Download the M2Crypto module

The M2Crypto module can currently be found at

<http://pypi.python.org/packages/source/M/M2Crypto/M2Crypto-0.21.1.tar.gz>

Unpack it in \$HOME.

2. Install M2Crypto in your virtual environment

```
$ cd $HOME/M2Crypto-0.21.1
$ $HOME/software/Zope/bin/python setup.py install
```

To test whether the installation was completed successfully, you can simply try to load the module in Python:

```
$ $HOME/software/Zope/bin/python
Python 2.7.1+ (r271:86832, Apr 11 2011, 18:05:24)
[GCC 4.5.2] on linux2
Type "help", "copyright", "credits" or "license" for more information.
>>> import M2Crypto
>>>
```

If this triggers an error message, an `ldconfig` may be required or you may have to close and reopen the terminal.

3. Patch the Zope installation

The next step ensures that, depending upon the configuration, Zope will actually use the M2Crypto module that has now been installed.

```
$ $BICSUITEHOME/zope/https/patch.sh
```

4. Create the files required by Zope HTTPS

The files required by Zope can be saved to the "etc" directory of the Zope instance, although this is not mandatory. In this guide, it is assumed that this is indeed the case. If you want to adapt this procedure, it is assumed that you have a sound understanding of how both TLS/SSL and HTTPS function.

First of all, navigate to the "etc" directory:

```
$ cd $HOME/bicsuiteweb/etc
```

An SSL Certificate Authority is now created:

```
$ openssl req -new -x509 -newkey rsa:2048 -keyout cakey.pem \
> -out cacert.pem -days 3650
```

The following parameters can be entered as an example:

```
Enter PEM pass phrase: super_secret
Verifying - Enter PEM pass phrase: super_secret
Country Name (2 letter code) [AU]:DE
State or Province Name (full name) [Some-State]:Bavaria
```



```
Locality Name (eg, city) []:Schrobenhausen
Organization Name (eg, company) [Internet Widgits Ltd]:independIT
Organizational Unit Name (eg, section) []:Development
Common Name (eg, YOUR name) []:Dieter Stubler
Email Address []:dieter.stubler@independit.de
```

You can now create an SSL server certificate:

- Generate a key for the server certificate:

```
$ openssl genrsa -out serverkey.pem -aes128 2048 -days 3650
```

```
Enter pass phrase for serverkey.pem: dummy
```

```
Verifying - Enter pass phrase for serverkey.pem: dummy
```

- Remove the password from serverkey.pem:

```
$ openssl rsa -in serverkey.pem -out serverkey.pem
```

```
Enter pass phrase for serverkey.pem: dummy
```

- Generate a Certificate Signing Request:

```
$ openssl req -new -key serverkey.pem -out req.pem -nodes
```

The following parameters can be entered as an example:

```
Country Name (2 letter code) [AU]:DE
State or Province Name (full name) [Some-State]:Bavaria
Locality Name (eg, city) []:Schrobenhausen
Organization Name (eg, company) [Internet Widgits Ltd]:independIT
Organizational Unit Name (eg, section) []:Development
Common Name (eg, YOUR name) []:my_hostname
Email Address []:dieter.stubler@independit.de
A challenge password []:
An optional company name []:
```

Important !!!

`my_hostname` must be replaced with the name that is used by the browser to address the HTTPS host !!!

- Sign the certificate:

- a) Save openssl.cnf (usually requires root privileges)

```
# cp /etc/ssl/openssl.cnf /etc/ssl/openssl.cnf_save
```

- b) Edit openssl.cnf:

```
# sudo vi /etc/ssl/openssl.cnf
```

The following entries are modified:

```
dir                = .
private_key        = $dir/cakey.pem
RANDFILE           = $dir/.rand
default_days       = 3650
new_certs_dir      = $dir
```

c) Prepare files:

```
$ touch index.txt
$ echo 01 > serial
```

d) Sign:

```
$ openssl ca -in req.pem -notext -out servercert.pem
```

Example:

```
Enter pass phrase for ./cakey.pem: super_secret
Sign the certificate? [y/n]:y
1 out of 1 certificate requests certified, commit? [y/n]y
```

e) Restore the old openssl.cnf

```
# mv /etc/ssl/openssl.cnf_save /etc/ssl/openssl.cnf
```

- Write the SSL server certificate:

```
$ cat servercert.pem serverkey.pem > server_cert.pem
```

- Create SSL entropy pool :

```
$ dd if=/dev/random of=ssl_entropy_pool.dat bs=1024 count=1
```

- Create ssl_dh_init:

```
$ openssl dhparam -out ssl_dh_init.pem 1024
```

5. Configure Zope HTTPS

If https is to listen at port 8085 and

```
$HOME == /home/bicsuite
```

the following should be added to the file `$HOME/bicsuiteweb/etc/zope.conf` after the closing tag `</http-server>`:

```
<https-server>
# valid keys are "address", "force-connection-close",
# required keys are
#     "x509_remote_users",
#     "ssl_certificate_authority",
#     "ssl_server_certificate"
address 8085
x509_remote_users off
ssl_certificate_authority /home/bicsuite/bicsuiteweb/etc/cacert.pem
ssl_server_certificate /home/bicsuite/bicsuiteweb/etc/server_cert.pem
ssl_dh_init /home/bicsuite/bicsuiteweb/etc/ssl_dh_init.pem
ssl_entropy_pool /home/bicsuite/bicsuiteweb/etc/ssl_entropy_pool.dat
# force-connection-close off
</https-server>
```

If communication should only take place via https, the section `<http-server>` to `</http-server>` is to be commented out.

6. Restart Zope

```
$ $HOME/bicsuiteweb/bin/zopectl stop
```

We use runzope so that we see any errors straight away:

```
$ $HOME/bicsuiteweb/bin/runzope
```

`$HOME/bicsuiteweb/log/event.log` is also helpful for analysing problems.

7. Once it has functioned properly, cancel runzope (Ctrl+C) and restart Zope

```
$ $HOME/bicsuiteweb/bin/zopectl start
```

To start BICsuite!Web:

In the browser

`http://hostname:8080/SDMS`

or

`https://hostname:8085/SDMS`

The first time BICsuite!Web is started, the server certificate has to be confirmed as being trustworthy in the browser. This is all you need to do in Firefox. In Internet Explorer, however,

`/home/bicsuite/bicsuiteweb/etc/cacert.pem`

has to be imported under Internet Options → Content → Publishers... → Trusted Root Certification Authorities to prevent warnings from constantly being displayed.

Installation (Zope4+)

1. Requirements

To install Zope4 or Zope5, the following packages have to be installed:

- a) python3

```
$ sudo yum install python3
```

- b) python3 development headers

```
$ sudo yum install python3-devel
```

2. Create a python virtual environment for your Zope installation

```
$ export INSTALLDIR=$HOME/software
$ export ZOPE5VENV=Zope5
$ export ZOPE5DIR=$INSTALLDIR/$ZOPE5VENV
$ mkdir -p $INSTALLDIR
$ cd $INSTALLDIR
$ python3 -m venv $ZOPE5VENV
```

3. Installation of the Zope5 software

Install the latest stable release of Zope5. At the time of this writing this is Zope 5.1.2.

```
$ cd $Zope5DIR
$ bin/pip install -U pip wheel
$ bin/pip install Zope[wsgi]==5.0 \
-c https://zopefoundation.github.io/Zope/releases/5.1.2/constraints.txt
$ bin/pip install Products.ExternalMethod
$ bin/pip install Products.Sessions
$ bin/pip install Products.SiteErrorLog
$ bin/pip install Products.PythonScripts
```

If internet access is not available during the installation, Zope can also be installed offline. To do this, proceed as follows:

a) Download python packages

On an identical as possible system with internet access, execute the following commands:

```
$ wget \
https://raw.githubusercontent.com/zopefoundation/Zope/5.1.2/requirements.txt
$ pip download -r requirements.txt -d packages
```

b) Transferring files to the target system

The file 'requirements.txt' and the directory 'packages' now have to be transferred to the target system without internet access. Place the files in the directory \$HOME/software.

c) Installation on the target system

The following command installs Zope from the downloaded files:

```
$ cd $HOME/software
$ Zope/bin/pip install --no-index --use-wheel \
--find-links=./packages -r requirements.txt
```

4. Create a Zope instance for BICsuite!Web

```
$ cd $HOME
$ export Zope5INSTANCE=$HOME/Zope5Instance
$ $Zope5DIR/bin/mkwsgiinstance -d $Zope5INSTANCE \
-u sdmsadm:sdmsadm_password
```

The password can be chosen freely and is used later on, but the user name must be sdmsadm.

Test:

Start the Zope5 instance using the command:

```
$ $Zope5DIR/bin/runwsgi -v $Zope5INSTANCE/etc/zope.ini
```

The browser should show an 'Zope Auto-generated default page' for the Url <http://localhost:8080>.

Pressing Strg-C or clsing the terminal window will stop the insance again.

5. Installation of the BICsuite!Web components

To install the BICsuite!Web components , the Zope installation has to be extended by some modules:

```
$ cd $ZOE5INSTANCE
$ mkdir Extensions
$ cd Extensions
$ ln -s $BICSUITEHOME/zope4/Extensions/*.py .
$ cd ..
$ ln -s $BICSUITEHOME/zope4/BICsuiteSubmitMemory \
$ZOE5DIR/lib64/python3*/site-packages/Products
$ ln -s $BICSUITEHOME/zope4/StringFixer \
$ZOE5DIR/lib64/python3*/site-packages/Products
$ mkdir import
$ cd import
$ ln -s $BICSUITEHOME/zope4/import/SDMS.zexp .
```

Now the Zope instance has to be started again to enable access to these extensions.

```
$ $ZOE5DIR/bin/runwsgi -v $ZOE5INSTANCE/etc/zope.ini
```

The Zope Management user interface is now opened at the address

`http://localhost:8080/manage`

in a browser. This is done with the user `sdmsadm` together with the password you have assigned.

The front end software is now loaded into Zope (Import button):

- a) Import into the folder / SDMS.zexp.
- b) In the folder /SDMS/Install, mark and copy the folders User and Custom.
- c) Create the folders User and Custom in the folder / by pasting them.

6. Configure the server connections

The server connections are also configured in the Zope Management user interface. To do this, log on as the user `sdmsadm`.

The `SDMSServers` Python script is edited in the folder Custom. This script delivers a dictionary which has to contain an entry for every BICsuite Server that is to be addressed by this BICsuite!Web installation. The entry looks like this:

```
# Server name that identifies the server in the BICsuite!Web
# user interface
'servername' : {
    # TCP/IP hostname or address at which the BICsuite Server is running
    'HOST'      : 'hostname',
```

```

# Port at which the BICsuite Server is addressed
'PORT'      : '2506',

# BASIC, PROFESSIONAL, ENTERPRISE
'VERSION'   : 'BASIC',

# Optional property stating whether BICsuite!Web should
# cache the server connections
'CACHE'     : 'Y'

# Optional property stating for how long cached BICsuite!Web
# server connections should continue to be valid
# Default setting is 60 seconds, only significant if 'CACHE' : 'Y'
'TIMEOUT'   : '60'
}

```

An entry with the name `DEFAULT` must exist for bootstrapping. This entry can be deleted after the user has been set up (who obviously should not then use this connection).

If a server is to be addressed via a secure SSL connection, the following additional properties have to be defined as well:

```

# Connection is set up via Secure Socket Layer
'SSL'       : 'true',

# If stated, the identity of the BICsuite! Server is verified
# The stated file must contain the BICsuite!Server server certificate
'TRUSTSTORE' : 'truststore.pem',

# If the BICsuite!Server requires client authentication,
# this property must be defined and the stated file
# must contain the client's certificate and private key.
# The certificate must be known to the server in its trust store.
'KEYSTORE'   : 'keystore.pem'

```

Note:

When using SSL, for performance reasons it is advisable to use cached server connections because setting up a secure connection is a resource-intensive operation.

7. Open the BICsuite!Web interface

The user interface is now available at the address

```
http://localhost:8080/SDMS
```

A logon prompt is displayed when this page is opened. When the user has logged on, the application is started by clicking the "Take Off" button.

How to work with the interface is described in the relevant documentation.

Migration of an existing BICsuite!Web Zope2 database to Zope4+

To migrate the database on an existing BICsuite!Web Zope2 installation into a new BICsuite!Web Zope4+ installation, the following actions have to be taken:

1. Execute all steps of "Installation (Zope4+)", as described above.
2. Stop your BICsuite!web Zope4+ instance if running
3. Install zodbupdate

For the migration of an existing BICsuite!web Zope2 database, the tool zodbupdate has to be installed.

```
$ cd $ZPE5DIR
$ bin/pip install zodbupdate
```

Stop your BICsuite!web Zope2 instance if running

Copy your Data.fs of your BICsuite!web Zope2 instance into your Zope4+ instance
Set the environment variables ZPE5DIR, ZPE2INSTANCE and ZPE5INSTANCE to fit your environment.

```
$ rm $ZPE5INSTANCE/var/Data.fs*
$ cp $ZPE2INSTANCE/var/Data.fs $ZPE5INSTANCE/var
$ $ZPE5DIR/bin/zodbupdate -v -f var/Data.fs --convert-py3 --encoding utf8
```

Start your BICsuite!web Zope4+ instance

Replace the SDMS Folder

1. Open the management interface of your Zope4+ instance
2. Remove the temp_folder from your Zope Root(/) Folder
3. Navigate to the Folder containing the Folder SDMS
4. Remove the Folder SDMS
5. Import the Zope4+ SDMS.zexp

HTTPS using Zope behind an Apache Web Server

The simplest way to enable HTTPS access to the BICsuite!Web Frontend is to use an Apache web server as a proxy in front of Zope.

This section will not go deep into the configuration of apache but describes how to achieve HTTPS communication in an easy way.

Of course the Apache web server and its mod_ssl extension has to be installed. The exact process differs slightly for different operating systems and linux distributions.

In a RHEL or CentOS Environment the only thing to do here is:

```
# yum install httpd mod_ssl
```

In and Ubuntu Environment only a

```
# apt install apache2
```

```
# a2enmod ssl
```

is necessary.

Now the ssl Engine of Apache has to be configured. Often the package manager will do a lot of the job and will create a self-signed certificate which can be used or replaced by an official certificate.

Further on, a proxy rule has to be defined in the ssl configuration for the virtual host used. In principle, this looks like the following:

```
#
# Redirect to Zope
#
<IfModule mod_proxy.c>
    ProxyRequests Off
    ProxyPreserveHost On
    ProxyPass / http://127.0.0.1:8080/VirtualHostBase/\
                https/myvirtualhost:443/VirtualHostRoot/
    ProxyPassReverse / http://127.0.0.1:8080/VirtualHostBase/\
                      http/https/myvirtualhost:443/VirtualHostRoot/
</IfModule>

<IfModule mod_headers.c>
    RequestHeader set X-Forwarded-Proto "https"
</IfModule>
```

Please note: For layout reasons some lines have been wrapped which is indicated by a backslash. The actual configuration should contain unwrapped lines without the backslash.

Since the Zope Server is running on the same host as the apache web server, the request will be forwarded to 127.0.0.1:8080. As own hostname myvirtualhost is used in this example.

In request header the communication protocol will be set to https.

The only thing left to do, is to redirect any attempt to access the server using http to https. This can be done in the global configuration by adding the following line to the configuration of your virtual host:

```
Redirect permanent / https://myvirtualhost/
```

SSO for BICsuite with Zope

Introduction

In small environments, the user management integrated in BICsuite is practical and allows systems to be easily kept separate. This changes dramatically, how-

ever, when the environments become larger. Centralised management is essential to keep track of the different systems, their users and the associated rights.

In many cases, such a management capability is implemented using Microsoft's Active Directory. One interesting function is the single sign-on (SSO) principle. Here, users are authenticated when they log on to their workstations. They do not have to enter their password again when accessing a system that supports SSO. Instead, a token is used to establish that the authentication has already taken place. Apart from the fact that this makes things more convenient for the user, no sensitive data is exchanged in the process either, which makes for significantly greater security.

Using a model environment, this chapter explains how to integrate BICsuite with Active Directory. It will make it easy for you to reproduce this in your environment. In our model environment, the BICsuite and Zope servers are installed on a CentOS 7 Linux system. The computer is called `centos7sso!`. The network also contains an Active Directory server called `adserver`, as well as a Windows client. Both Windows machines are in the Windows domain `INDEPENDIT.DE`.

Procedure

To begin with, an Apache web server is installed together with the required modules and configured. The aim is to achieve not only the SSO functionality, but also to enable communication by https. The Apache server will communicate with the Zope server via a `proxy_html` interface. Since the Scheduling Server also performs user authentication, it likewise has to be made aware of the SSO situation.

Kerberos installation and configuration

The SSO protocol internally relies on Kerberos. Therefore the required software has to be installed and configured. The installation is easy. In a RHEL or CentOS environment a mere

```
yum install krb5-libs
yum install krb5-workstation
yum install sssd-krb5
yum install sssd-krb5-common
```

will do the job. The file `/etc/krb5.conf` contains the configuration of the Kerberos installation. In our model environment it looks like:

```
includedir /etc/krb5.conf.d/

[logging]
default = FILE:/var/log/krb5libs.log
kdc = FILE:/var/log/krb5kdc.log
admin_server = FILE:/var/log/kadmind.log

[libdefaults]
```

```

dns_lookup_realm = false
default_keytab_name = /etc/httpd/krb5.keytab
ticket_lifetime = 24h
renew_lifetime = 7d
forwardable = true
rdns = false
pkinit_anchors = FILE:/etc/pki/tls/certs/ca-bundle.crt
default_realm = INDEPENDIT.DE
default_ccache_name = KEYRING:persistent:%{uid}

[realms]
INDEPENDIT.DE = {
    kdc = ADSERVER.INDEPENDIT.DE
    master_kdc = ADSERVER.INDEPENDIT.DE
    admin_server = ADSERVER.INDEPENDIT.DE
    default_domain = INDEPENDIT.DE
}

[login]
krb4_convert = true
krb4_get_tickets = false

[domain_realm]
.independit.de = INDEPENDIT.DE
independit.de = INDEPENDIT.DE

```

The mentioned file `/etc/httpd/krb5.keytab` can be created on the Active Directory server and deleted after copying it over:

```

ktpass -princ HOST/centos7sso.independit.de@INDEPENDIT.DE -mapuser \
    bicsuite@INDEPENDIT.DE -crypto RC4-HMAC-NT \
    -ptype KRB5_NT_PRINCIPAL -pass "VerySecret" \
    -out c:\temp\krb5.keytab
ktpass -princ HTTP/centos7sso.independit.de@INDEPENDIT.DE -mapuser \
    bicsuite@INDEPENDIT.DE -crypto RC4-HMAC-NT \
    -ptype KRB5_NT_PRINCIPAL -pass "VerySecret" \
    -out c:\temp\krb5.keytab -in c:\temp\krb5.keytab

```

It is possible, that instead of RC4-HMAC-NT another encryption algorithm, like e.g. AES256-SHA1 has to be chosen. From the security perspective, it is recommended to use modern encryption algorithms.

The Apache server requires access to the Active Directory server to be able to perform the authorisation. In our example, we used `bicsuite@INDEPENDIT.DE`, an account without any special privileges. The name itself is insignificant. `VerySecret` was set as the password. A password that is easy to guess, even if it's "VerySecret". In order to test the Kerberos configuration the following statement will help:

```

KRB5_TRACE=/dev/stdout kinit -k -t krb5.keytab \
    -p HTTP/centos7sso.independit.de

```

The expected output looks like:

```

[root@centos7sso httpd]# KRB5_TRACE=/dev/stdout kinit -k \
    -t krb5.keytab -p HTTP/centos7sso.independit.de
....: Getting initial credentials for \
    HTTP/centos7sso.independit.de@INDEPENDIT.DE
....: Looked up etypes in keytab: aes256-cts
....: Sending unauthenticated request
....: Sending request (225 bytes) to INDEPENDIT.DE
....: Resolving hostname ADSERVER.INDEPENDIT.DE
....: Sending initial UDP request to dgram 192.168.123.45:88
....: Received answer (204 bytes) from dgram 192.168.123.45:88
....: Response was from master KDC
....: Received error from KDC: -1765328359/Additional \
    pre-authentication required
....: Preauthenticating using KDC method data
....: Processing preauth types: PA-PK-AS-REQ (16), \
    PA-PK-AS-REP_OLD (15), PA-ETYPE-INFO2 (19), \
    PA-ENC-TIMESTAMP (2)
....: Selected etype info: etype aes256-cts, salt \
    "INDEPENDIT.DEHTTPcentos8sso.independit.de", params ""
....: Retrieving HTTP/centos8sso.independit.de@INDEPENDIT.DE \
    from FILE:krb5.keytab (vno 0, enctype aes256-cts) with \
    result: 0/Success
....: AS key obtained for encrypted timestamp: aes256-cts/8EB8
....: Encrypted timestamp (for 1619424732.825845): plain ...
....: Preauth module encrypted_timestamp (2) (real) returned: \
    0/Success
....: Produced preauth for next request: PA-ENC-TIMESTAMP (2)
....: Sending request (305 bytes) to INDEPENDIT.DE
....: Resolving hostname ADSERVER.INDEPENDIT.DE
....: Sending initial UDP request to dgram 192.168.123.45:88
....: Received answer (98 bytes) from dgram 192.168.123.45:88
....: Response was from master KDC
....: Received error from KDC: -1765328332/Response too big for \
    UDP, retry with TCP
....: Request or response is too big for UDP; retrying with TCP
....: Sending request (305 bytes) to INDEPENDIT.DE (tcp only)
....: Resolving hostname ADSERVER.INDEPENDIT.DE
....: Initiating TCP connection to stream 192.168.123.45:88
....: Sending TCP request to stream 192.168.25.3:88
....: Received answer (1706 bytes) from stream 192.168.123.45:88
....: Terminating TCP connection to stream 192.168.123.45:88
....: Response was from master KDC
....: Processing preauth types: PA-ETYPE-INFO2 (19)
....: Selected etype info: etype aes256-cts, salt \
    "INDEPENDIT.DEHTTPcentos8sso.independit.de", params ""
....: Produced preauth for next request: (empty)
....: AS key determined by preauth: aes256-cts/8EB8
....: Decrypted AS reply; session key is: aes256-cts/9218
....: FAST negotiation: unavailable
....: Initializing KCM:0:99729 with default princ \
    HTTP/centos8sso.independit.de@INDEPENDIT.DE
....: Storing HTTP/centos8sso.independit.de@INDEPENDIT.DE -> \
    krbtgt/INDEPENDIT.DE@INDEPENDIT.DE in KCM:0:99729

```

```

...: Storing config in KCM:0:99729 for \
    krbtgt/INDEPENDIT.DE@INDEPENDIT.DE: pa_type: 2
...: Storing HTTP/centos8sso.independit.de@INDEPENDIT.DE -> \
    krb5_ccache_conf_data/pa_type/krbtgt\INDEPENDIT.DE\
    @INDEPENDIT.DE@X-CACHECONF: in KCM:0:99729

```

(For layout reasons the output has been shortened and reformatted).

Apache web server and modules

Installation

The next step is to install Apache (httpd) and several required modules. In case of RHEL/CentOS 7 there is a choice between `mod_auth_kerb` and `mod_auth_gssapi`. From RHEL/CentOS 8 on, the Kerberos module isn't supported any longer. Hence the `gssapi` module is the only option left. This means either

```

yum install httpd
yum install mod_ssl
yum install mod_ldap
yum install mod_proxy_html
yum install mod_auth_kerb

```

to obtain the Kerberos module, or

```

yum install httpd
yum install mod_ssl
yum install mod_ldap
yum install mod_proxy_html
yum install mod_auth_gssapi

```

for the `gssapi` module.

Configuration

On Red Hat-based systems, Apache is configured under `/etc/httpd` and in some subdirectories. This may be different for other distributions, but the principle remains the same.

First of all, it is ensured that communication with the Apache web server only takes place via https. To do this, some parameters need to be entered in the file `/etc/httpd/conf.d/ssl.conf`. In our model environment these are:

```

ServerName centos7sso.independit.de:443
SSLCertificateFile /etc/pki/tls/certs/centos7sso.crt
SSLCertificateKeyFile /etc/pki/tls/private/centos7sso.key

```

Depending on the environment, the intermediate certificates may naturally also play a role.

In the file `etc/httpd/conf/httpd.conf`, it is then ensured that any requests sent to the standard http port are redirected to the https port:

```

<VirtualHost _default_:80>
    ServerName centos7sso.independit.de:80
    #
    # force the use of https
    #
    Redirect permanent / https://centos7sso.independit.de/
</VirtualHost>

```

The firewall should obviously also be informed as well, for example like this:

```

firewall-cmd --zone=public --add-service=https
firewall-cmd --zone=public --permanent --add-service=https

```

The user authentication looks a bit more complicated and needs to be supplemented with information from the environment. Depending on the module that is used, one of both similar albeit different configurations apply. In case of `mod_auth_kerb` the following configuration is used in our model environment:

```

# If the AD users belong to multiple groups, the
# FieldSize can quickly become exhausted and you run into errors
# For this reason it is enlarged here
LimitRequestFieldSize 32768
<Location "/bicsuite">
    AuthType          Kerberos
    KrbAuthRealms     INDEPENDIT.DE
    KrbServiceName    HTTP
    Krb5Keytab        /etc/httpd/krb5.keytab
    KrbMethodNegotiate On
    KrbMethodK5Passwd Off
    require valid-user

    RewriteEngine on
    RewriteCond %{REMOTE_USER} (.*?)
    RewriteRule .* - [E=X_REMOTE_USER:%1]
    RequestHeader set REMOTE_USER %{X_REMOTE_USER}e
    RequestHeader set X-Remote-User %{REMOTE_USER}s

    SetHandler "proxy:http://127.0.0.1:8080"
    SetEnvIfNoCase ^Authorization$ "(.+)" HTTP_AUTHORIZATION=$1
</Location>

```

The configuration in case of `mod_auth_gssapi` looks similar, even a bit simpler:

```

<Location "/bicsuite">
    AuthType          GSSAPI
    AuthName          "INDEPENDIT.DE"
    GssapiCredStore    keytab:/etc/httpd/krb5.keytab
    GssapiSSLOnly      On
    GssapiLocalname    Off
    Require valid-user

    RewriteEngine on

```

```

RewriteCond %{REMOTE_USER} (.* )
RewriteRule .* - [E=X_REMOTE_USER:%1]
RequestHeader set REMOTE_USER %{X_REMOTE_USER}e
RequestHeader set X-Remote-User %{REMOTE_USER}s

SetHandler "proxy:http://127.0.0.1:8080/"
SetEnvIfNoCase ^Authorization$ "(.+)" HTTP_AUTHORIZATION=$1
</Location>

```

If the web user requests the location `bicsuite`, or naturally a resource below the `bicsuite` folder, Kerberos checks whether the user is authorised to do so. This requires the domain (in our case `INDEPENDIT.DE`) as well as a file `krb5.keytab` which was stored in the Apache configuration directory.

If the authentication is successful, the request is forwarded to the Zope server using `http`. Any authorisation headers are sent as well.

SELinux

If SELinux is activated, the Apache server is prohibited from opening socket connections itself. However, since this is required for the communication between Apache and Zope, it must be allowed:

```

/usr/sbin/setsebool -P httpd_can_network_connect 1

```

Results test

Whether everything has gone well so far can be tested. To do this, a directory (e.g. `ssotest`) is created in the `DocumentRoot`, in this case `/var/www/html`. A file `index.html` is created in the directory with content such as this:

```

<DOCTYPE html>
<html>
  <body>

    <h1>My First Heading</h1>

    <p>Hello World</p>

  </body>
</html>

```

The Apache configuration now has to be temporarily modified for the test. This is done by editing the `location` section as follows:

```

<Location "/ssotest">
  AuthType      Kerberos
  KrbAuthRealms INDEPENDIT.DE
  KrbServiceName HTTP
  Krb5Keytab    /etc/httpd/krb5.keytab

```

```

KrbMethodNegotiate On
KrbMethodK5Passwd Off
require valid-user
    # SetHandler "proxy:http://127.0.0.1:8080"
    # SetEnvIfNoCase ^Authorization$ "(.+)" HTTP_AUTHORIZATION=$1
</Location>

```

If `https://centos7sso/ssotest` is now accessed from a Windows workstation after the Apache server has been restarted, "Hello World" should be displayed. Calling `wget` from the server itself, such as with `wget https://localhost/ssotest`, should trigger a "401 Unauthorized" page.

If the test was successful, the changes should now be undone again.

Zope extension and configuration

The second player is the Zope server, which should naturally learn about its good fortune as well. Zope must likewise be able to talk to the Active Directory server. To do this, it requires the `python_ldap` package, whose installation in turn requires the `openldap-devel` package:

```

yum install openldap-devel
cd $BICSUITEHOME/./software/Zope
bin/pip install python-ldap

```

It is advisable to prepare Zope for using SSO before importing `SDMS.zexp`. Although SSO can also be configured later, this requires a few extra steps. In particular, it must be noted that the URL used to address the GUI is different compared to a normal installation.

Product installation

A user ID is required for Zope to be able to execute requests in the right context. Without SSO, the login takes place in Zope, which means that the information is available. With SSO, Zope is sent the user information from the Apache server. An extension is required so that Zope can handle the information.

The installation is easy:

```

cd $BICSUITEHOME/./bicsuiteweb/Products
ln -s $BICSUITEHOME/bicsuite/zope/RemoteUserFolder .

```

It is important to make the port accessible to Apache, but not externally. With the aid of `firewalld`, that can look like this:

```

firewall-cmd --permanent --zone=public --add-rich-rule='
    rule family="ipv4"
    source address="127.0.0.1/32"
    port protocol="tcp" port="8080" accept '

```

Installing SDMS.zexp

The GUI application is installed in a similar way to the installation without SSO. What is important to note here, however, is that the installation takes place in a subfolder instead of the root folder. The name of the folder is not so important as long you always use the correct name.

For technical reasons, two empty folders and another folder for installing the software in the Zope root folder are required. To do this, the Zope management interface is opened by bypassing the Apache server. In our model environment, the URL

```
\verb!http://centos7sso.independit.de:8080/manage!
```

is entered in the browser. If a login prompt is displayed, the user credentials that were entered when installing Zope have to be used. This is typically the user `sdmsadm`.

First of all, the folders `bicsuite`, `web` and `GUI` are now created. A property `SDMSROOT` with the value `/bicsuite/GUI` is also created. Later, the URL

```
\verb!https://centos7sso.independit.de/bicsuite/GUI/SDMS!
```

is used to open the GUI.

Now all the steps follow as with the normal installation, except that `SDMS.zexp` is imported into the folder `GUI`!. The folders `Custom` and `User` are also created in the folder `GUI`.

Im `Custom` Folder gibt es ein Konfigurationsscript namens `SDMSServers`, in dem festgehalten wird, welche Scheduling Servers von der Zope Instanz aus bedient werden können. Zope muss wissen, dass SSO benutzt werden soll: Within the `Custom` folder there is a configuration script called `SDMSServers`, which is used to define which scheduling servers can be accessed by this Zope instance. It is crucial to tell Zope that it should use the SSO protocol.

```
#
# define all accessible SDMS Servers here
#
return {
  'DEFAULT' : {
    'HOST'    : 'localhost',
    'PORT'    : '2506',
    'VERSION' : 'BASIC',
    'CACHE'   : 'Y',
    'TIMEOUT' : '60',
    'SSO'     : True, # <-- Use SSO
    'SSL'     : 'N',
    #
    # Edit following Options if SSL set to 'Y'
    #
    'TRUSTSTORE' : 'truststore.pem',
    'KEYSTORE'   : 'keystore.pem'
  }
}
```


Finally, an object of the type `RemoteUserFolder` is created in the GUI folder.

Configuring the SDMS application

Under `$BICSUITEHOME/etc` is a file called `ZopeSSO.conf.template`. This is copied to the configuration directory, which is usually `$BICSUITEHOME/./etc`, and renamed into `ZopeSSO.conf`.

This file is pretty self-explanatory. It begins with general settings, which can then be overwritten for each domain or server.

In our model environment, the following settings were used:

```
# =====
# ZopeSSO.conf.template
#
# Copy this file to the BICSUITECONFIG directory and edit it according
# to your needs
# At least all properties set to <TO_BE_CONFIGURED> have to be set.
# =====
# Configurations for handling SSO for the BICsuite web frontend
#
# WARNING:
# This file contains credentials for LDAP and BICsuite ADMIN access.
# Make this file only readable for the user running the Zope
# application server
{
    #=====
    # General configurations which can be (partially) overridden by
    # domain-specific or server-specific configurations
    #-----
    # Defaults for domain specific settings if not set in the DOMAINS
    # section
    # - - - - -
    # WebNameCase defines how names for Zope authenticated user names
    # are converted
    # 'UPPER' convert to upper case
    # 'LOWER' convert to lower case
    # 'MIXED' no conversion (default)
    'WebNameCase' : 'MIXED',
    # - - - - -
    # WebAutoCreateUsers indicates if AD users should be created
    # automatically as BICsuite frontend users. If WebUseLdapGroups,
    # is set to True, only AD users who are members of the UserGroup
    # and/or ManagerGroup below will be allowed.
    # True
    # False (default)
    'WebAutoCreateUsers' : True,
    # - - - - -
    # WebUseLdapGroups indicates if ldap groups should be used to
    # detect whether an AD user is allowed to log in to the BICsuite
    # web frontend
    # True
    # False (default)
```

```

# 'WebUseLdapGroups' : True,
#-----
# WebIncludeDomainNames indicates if Domain Names should be part
# of web user identifiers
# True or False
# defaults to False
'WebIncludeDomainNames' : False,
#-----
# WebUserGroup allowed to log in in via SSO
# defaults to 'BICSUITE_WEB_USER'
'WebUserGroup' : 'bicsuite',
#-----
# manager group granting manage privilege on Zope website
# defaults to 'BICSUITE_WEB_MANAGER'
'WebManagerGroup' : 'bicsuite_admin',
#-----
# WebGroupCheckIntervall is the time in minutes after which ldap
# group assignments for a BICsuite web server are checked again
# defaults to 60 (1 hour)
# 'WebGroupCheckIntervall' : 60
#-----
# Defaults for server-specific settings if not set in the SERVERS
# section
#-----
# ServerIncludeUserDomainNames indicates if domain names should be
# part of user identifiers
# True or False
# defaults to False
'ServerIncludeUserDomainNames' : False,
#-----
# ServerIncludeGroupDomainNames indicates if domain names should
# part of group identifiers
# True or False
# defaults to False
'ServerIncludeGroupDomainNames' : False,
#-----
# ServerUserNameCase defines how names for BICsuite are converted
# UPPER case
# LOWER case
# MIXED case (don't convert them)
# defaults to 'UPPER'
'ServerUserNameCase' : 'UPPER',
#-----
# ServerGroupNameCase defines how names for BICsuite groups are
# converted
# UPPER case
# LOWER case
# MIXED case (don't convert them)
# defaults to 'UPPER'
'ServerGroupNameCase' : 'UPPER',
#-----
# ServerAutoCreateUsers indicates if AD users should be created
# automatically. If ServerUseLdapGroups is True, only AD users

```

```

# who are a member of any AD group named
# <ServerBicsuitePrefix>_<ServerName>_<groupname> are allowed
# True or False
# defaults to False
'ServerAutoCreateUsers' : True,
#-----
# ServerUseLdapGroups indicates if AD groups should be used
# True or False
# defaults to False
'ServerUseLdapGroups' : True,
#-----
# ServerBicsuitePrefix is the prefix used for AD groups. Groups
# called other than <ServerBicsuitePrefix>_<ServerName>_<groupname>
# are ignored
# defaults to 'BICSUITE'
'ServerBicsuitePrefix' : 'bicsuite',
#-----
# ServerName is the server name used for AD groups. Groups called
# other than <ServerBicsuitePrefix>_<ServerName>_<groupname>
# are ignored
# defaults to 'DEFAULT'
'ServerName' : 'centos7sso',
#-----
# ServerDefaultGroupSuffix
# Suffix used to decide whether a AD group should be the default
# group defaults to '_ISDEFAULT'
'ServerDefaultGroupSuffix' : '_ISDEFAULT',
#=====
# Domain-specific configurations independent of the BICsuite
# server. Accessing users from domains not configured here will not
# be able to log on to the BICsuite web frontend via SSO
#-----
'DOMAINS' : {
    # domain name as in <DOMAIN_NAME>\UserName
    'INDEPENDIT.DE' : {
        #-----
        # Domain-specific settings or one BICsuite server
        #-----
        # ldap server and base to get group membership from
        # Example:
        # 'LdapServer' : 'ldap://192.168.0.1',
        'LdapServer' : 'ldap://adserver.independit.de',
        # Example:
        # 'LdapBaseDn' : 'DC=INDEPENDIT,DC=dieter,DC=de',
        'LdapBaseDn' : 'DC=INDEPENDIT,DC=de',
        #-----
        # ldap credentials to use for group membership retrieval
        # Example
        # 'LdapUsername' : 'Administrator@INDEPENDIT.DIETER.DE',
        'LdapUsername' : 'bicsuite_admin',
        'LdapPassword' : 'G0H0ME-123',
        #-----
        # WebNameCase defines if names for Zope authenticated user

```

```

# names have to be converted to
# 'UPPER' convert to upper case (default)
# 'LOWER' convert to lower case
# 'MIXED' no conversion
'WebNameCase' : 'UPPER',
#-----
# WebAutoCreateUsers indicates if AD users should be created
# automatically as BICSuite frontend users. If
# UseLdapWebGroups is set to True, only AD users who are
# members of the UserGroup and/or ManagerGroup below will
# be allowed.
# True
# False (default)
'WebAutoCreateUsers' : True,
#-----
# WebUseLdapGroups indicates if Ldap groups should be used
# to detect whether AD user is allowed to log in to the
# BICSuite web frontend
# True
# False (default)
# 'WebUseWebGroups' : False,
#-----
# WebIncludeDomainNames indicates if Domain Names should be
# part of web user identifiers
# True or False
# defaults to False
'WebIncludeDomainNames' : False,
#-----
# WebUserGroup allowed to log in in via SSO
# defaults to 'BICSUITE_WEB_USER'
'WebUserGroup' : 'bicsuite',
#-----
# manager group granting manage privilege on Zope website
# defaults to 'BICSUITE_WEB_MANAGER'
'WebManagerGroup' : 'bicsuite_admin',
#-----
# WebGroupCheckIntervall is the time in minutes after which
# ldap group assignments for a BICSuite web server are
# checked again
# defaults to 60 (1 hour)
'WebGroupCheckIntervall' : 60
#-----
}
},
#=====
# BICSuite server-specific configurations
#-----
'SERVERS' : {
#-----
# For every BICSuite server to be accessed via SSO, the following
# section must be created with hostname:port
# Example: localhost:2506
'localhost:2506' : {

```

```

#-----
# General configuration for a BICsuite server independent
# of the login domain
#-----
# login credentials for a BICsuite admin user who is allowed
# to manage users and group. Used also to connect to BICsuite
# before sending the 'alter session set user' command when
# executing statements vis-a-vis BICsuite for a user
'AdminUser'      : 'SYSTEM',
'AdminPassword'  : 'GOHOME',
#-----
# Defaults for server-specific settings if not set in the
# DOMAINS section
#-----
# ServerIncludeUserDomainNames indicates if domain names
# should be part of user identifiers
# True or False
# defaults to False
'ServerIncludeUserDomainNames' : False,
#-----
# ServerIncludeGroupDomainNames indicates if domain names
# should be part of group identifiers
# True or False
# defaults to False
'ServerIncludeGroupDomainNames' : False,
#-----
# ServerUserNameCase defines how names for BICsuite users
# are converted
# UPPER case
# LOWER case
# MIXED case (don't convert them)
# defaults to 'UPPER'
'ServerUserNameCase' : 'UPPER',
#-----
# ServerGroupNameCase defines how names for BICsuite groups
# are converted
# UPPER case
# LOWER case
# MIXED case (don't convert them)
# defaults to 'UPPER'
'ServerGroupNameCase' : 'UPPER',
#-----
# ServerAutoCreateUsers indicates if AD users should be
# created automatically. If ServerUseLdapGroups is True
# only AD users who are a member of any AD group named
# <ServerBicsuitePrefix>_<ServerName>_<groupname> are allowed
# True or False
# defaults to False
'ServerAutoCreateUsers' : True,
#-----
# ServerUseLdapGroups indicates if AD groups should be used
# True or False
# defaults to False

```

```

'ServerUseLdapGroups' : True,
#-----
# ServerBicsuitePrefix is the prefix used for AD groups.
# Groups called other than
# <ServerBicsuitePrefix>_<ServerName>_<groupname> are ignored
# defaults to 'BICSUITE'
'ServerBicsuitePrefix' : 'BICSUITE',
#-----
# ServerName is the server name used for AD groups. Groups
# called other than
# <ServerBicsuitePrefix>_<ServerName>_<groupname> are ignored
# defaults to 'DEFAULT'
'ServerName' : 'centos7sso',
#-----
# ServerDefaultGroupSuffix
# Suffix used to decide whether a AD group should be the
# default group
# defaults to '_ISDEFAULT'
# 'ServerDefaultGroupSuffix' : '_ISDEFAULT',
#-----
# Domain-specific configurations for this BICsuite server
#-----
'DOMAINS' : {
    # optional server and domain-specific configuration
    # overriding server and base defaults
    'INDEPENDIT.DE' : {
        #-----
        # Server-specific settings for this domain
        #-----
        # ServerIncludeUserDomainNames indicates if
        # domain names should be part of user
        # identifiers
        # True or False
        # defaults to False
        # 'ServerIncludeUserDomainNames' : False,
        #-----
        # ServerIncludeGroupDomainNames indicates if
        # domain names should be part of group
        # identifiers
        # True or False
        # defaults to False
        # 'ServerIncludeGroupDomainNames' : False,
        #-----
        # ServerUserNameCase defines how names for
        # BICsuite users
        # are converted
        # UPPER case
        # LOWER case
        # MIXED case (don't convert them)
        # defaults to 'UPPER'
        # 'ServerUserNameCase' : 'UPPER',
        #-----
        # ServerGroupNameCase defines how names for

```


Configuring the BICsuite server

The BICsuite server also needs some configuration so that it knows how to respond to SSO logins. What is important is that the settings match the settings for the Zope server. In our model environment, the relevant parameters were set as follows:

```
#
# SSOincludeDomainNames indicates if Domain Names should
# be part of group/user identifiers
# default = false
#
SSOincludeDomainNames=false

#
# SSOautoCreateUsers indicates if AD users should be
# created automatically
# default = false
#
SSOautoCreateUsers=true

#
# SSOautoCreateGroups indicates if AD groups should be
# created automatically
# default = false
#
SSOautoCreateGroups=true

#
# SSOuseADGroups indicates if AD groups should be used
# or not
# default = false
#
SSOuseADGroups=true

#
# SSOserverName is the name of the server (within AD);
# this is used to filter out groups
#
SSOserverName=centos7sso

#
# SSObicsuitePrefix is the prefix used for AD groups.
# Groups called other than
# <SSObicsuitePrefix>_<SSOserverName>_<groupname>
# are ignored
#
SSObicsuitePrefix=bicsuite

#
# SSOnameCase defines if names have to be converted to
# UPPER case (= default value; make identifier case
# insensitive if they adhere to BICsuite
# naming standards)
```



```
# LOWER case
# MIXED case (= don't convert them)
#
SSOnameCase=UPPER
```

Settings on the user side

For SSO to function, the browser must know that it should report the credentials to the Apache server. The configuration varies slightly depending on the browser. Which settings need to be configured for some commonly used browsers is explained in the next sections. The order is purely alphabetical.

Chrome and Microsoft Edge

The procedure is the same for both Chrome and Edge. Select the local intranet in the Windows Control Panel under → Network and Internet → Internet Options → Security tab. After clicking the sites and the Advanced button, enter the host name of the Apache server.

Firefox

In Firefox, enter the URL `about:config`. Now search for the word `negotiate`. One of the search results should be

```
network.negotiate-auth.trusted-uris
```

Double-click to enter a value. This is the host name of the Apache server. In our model environment this is `centos7sso.independit.de`.

After saving the setting and restarting the browser, the GUI should be accessible without having to enter a password.

Administration of the Zope server

Access separate from the SSO logic is required for administering the Zope server. This is due to the fact that users who are given access via SSO are treated differently from "normal" Zope users.

Nevertheless, the administration should preferably be handled via a secure connection as well. This is possible without any problems, albeit with a few restrictions. Basically, as already explained, the Apache server is configured as a reverse proxy. However, requests which reference `/bicsuite` must not be translated since the SSO logic is supposed to take effect here.

This results in the following addition to the Apache configuration:

```
ProxyRequests Off
ProxyPreserveHost On
ProxyPass / http://127.0.0.1:8080/VirtualHostBase/https/\
centos7sso.independit.de:443/VirtualHostRoot/
ProxyPassReverse / http://127.0.0.1:8080/VirtualHostBase/\
http/https/centos7sso.independit.de:443/VirtualHostRoot/
RequestHeader set X-Forwarded-Proto "https"
```

Please note: A line break has been entered for presentation reasons. As usual, this is indicated by a backslash. In the actual configuration, the two lines should be concatenated again without a backslash and whitespace.

This addition can be entered directly after the previously added instructions in `ssl.conf`.

It is advisable to use a different browser from the standard browser for administrative tasks because otherwise some undesirable side effects will occur. This is because you cannot be simultaneously logged in to a Zope instance as two different users in one browser.

3 Installation in a Windows environment

Installing the BICsuite Server

Introduction

This guide assumes that you have unpacked the BICsuite distribution to a local directory. We recommend using `C:\Program Files`. The unpacked distribution should now be in the directory `ic:\Program Files\bicsuite`.

The command shell is required for some of the installation steps. Many of the instructions or examples in this guide assume that you are working in the command line window. In this case, in the examples `C:>` is shown as the prompt. This naturally does not have to be entered. Many operations can often also be (more easily) performed using the Windows interface. You can obviously choose the method that is easiest for you.

You should also note that for typesetting reasons some lines in the examples are shown with line breaks. This is evident where the line has an indent and there is no prompt at the beginning of the line. When copying the commands, you should *not* use a return at the end of the line.

Installation

1. Create the installation directory

Create a folder for installing the BICsuite software. In the following we refer to this directory as `INSTALLDIR`.

Example:

```
C:> cd \Program Files
C:> mkdir bicsuite
```

2. Unpack the BICsuite distribution in `INSTALLDIR`

`INSTALLDIR` should now have a subdirectory called `bicsuite-release`.

Example:

```
C:\Program Files\bicsuite\bicsuite-\versionnumber
```

3. Set the `BICSUITEHOME` environment variable

Close all the Windows command line prompts. Under Control Panel → System → Advanced → Environment Variables, set the variable `BICSUITEHOME` as the system variable for the BICsuite distribution directory.

Example:

```
BICSUITEHOME=C:\Program Files\bicsuite\bicsuite-\versionnumber
```

4. Set the optional environment variables BICSUITECONFIG

We advise you not to save the BICsuite configuration files under %BICSUITEHOME%, but to use a separate directory.

Close all the Windows command line prompts. Under Control Panel → System → Advanced → Environment Variables, set the variable BICSUITECONFIG as the system variable for the BICsuite configuration directory.

Example:

```
BICSUITECONFIG=C:\Program Files\bicsuite\etc
```

This will make it easier to later upgrade to a newer release of BICsuite.

5. Create the configuration folder

If the configuration files are not to be saved under %BICSUITEHOME%, you will have to create the configuration folder. For instance:

```
C:> mkdir "%BICSUITECONFIG%"
```

Configuration files are required that can be created from the provided templates regardless of whether a separate folder is to be used for the configuration or not. These will then be modified at a later stage of the installation.

Open a Windows command line window. If you have not set the environment variable %BICSUITECONFIG%, use %BICSUITEHOME%\etc instead.

Simply copying the Configuration template is adequate for the time being:

```
C:> cd /d "%BICSUITEHOME%\etc"
C:> copy server_template.conf "%BICSUITECONFIG%\server.conf"
C:> copy BICSUITE_CONF_TEMPLATE.BAT "%BICSUITECONFIG%\BICSUITE_CONF.BAT"
C:> copy JAVA_CONF_TEMPLATE.BAT "%BICSUITECONFIG%\JAVA_CONF.BAT"
```

In the file BICSUITE_CONF.BAT, the value of the variable BICSUITELOGDIR can be modified as required so that the system logging takes place outside of the installation directory. In this case, you will obviously have to make sure that this directory actually exists.

If the Java executable is not in the path given here, the variable BICSUITEJAVA has to be set to the qualified path in the file JAVA JAVA_CONF.BAT. For example:

```
SET BICSUITEJAVA=C:\Program Files (x86)\Java\bin\java
```

6. Download and install a database management system supported by BICsuite.

BICsuite for Windows currently supports the following systems:

- Postgres (page [61](#))

- MySQL (page 63)
- Microsoft™ SQL Server (page 65)
- Oracle (page 71)
- Ingres (page 67)

Reference is made to the appropriate sections regarding the installation of the chosen database system as well as modifying the configuration of the BICsuite Enterprise Scheduling System.

7. Modify the PATH environment variable

Close all the Windows command line prompts. Navigate to Control Panel → System → Advanced → Environment Variables and add the following string to the Path system variable:

```
%BICSUITEHOME%\bin
```

Example:

```
PATH= ... ;C:\Program Files\bicsuite\bicsuite-2.10\bin
```

8. An up-to-date Java Runtime Environment has to be installed if this has not already been done

A JRE can be downloaded from <http://www.oracle.com/technetwork/java/index.html> and installed.

9. Start BICsuite Server

Open a Windows command line window if you have not already done so.

Run the BICsuite command line utility RUN_SERVER.

```
C:> RUN_SERVER
```

Leave this window open for the next steps.

10. Create the optional file .sdmshrc in the Windows user's home directory

When this file is created, it will be read by most of the BICsuite command line utilities to supplement any missing parameters.

This means that when you call `sdmsh`, for example, you won't have to enter the host port user and password each time.

Sample file:

```
User=system
Password=G0H0ME
Host=localhost
Port=2506
```

The important thing to note here is that this file may contain the password for accessing the Scheduling System. It should therefore be protected against access (including read access) by other users.

11. Test

Open a Windows command line window.

Run the BICsuite command line utility `sdmsh`:

```
C:> sdmsh
```

Or, if you haven't created an `.sdmsrc` file:

```
C:> sdmsh -u SYSTEM -w G0H0ME -h localhost -p 2506
```

A corresponding prompt is displayed once you have successfully logged on to the server. You can now send commands to the Scheduling Server.

```
[SYSTEM@localhost:2506] SDMS> show system;
```

System information about the running BICsuite Server should now be displayed.

The utility can be exited again as follows:

```
[SYSTEM@localhost:2506] SDMS> exit
```

12. Install the convenience package

The convenience package installs a commonly used configuration for an exit state model. In the following command line it is assumed that the `.sdmsrc` file exists. Should this not be the case, the command line has to be modified analogue to the previous step.

```
C:> type "%BICSUITEHOME%\install\convenience.sdms" | sdmsh
```

13. Install the examples

Installing the provided sample workflow definitions requires just a few commands:

```
C:> cd /d "%BICSUITEHOME%\install
C:> SETUP_EXAMPLE_JOBSERVERS.BAT
C:> type setup_examples.sdms | sdmsh
```

14. Automatically start the BICsuite Server

To start the BICsuite Server automatically when you log on, you just have to save one shortcut to the start script in the startup folder.

The start script is called:

```
%BICSUITEHOME%\bin\RUN_SERVER.BAT
```

15. Automatically start the jobserver

To start the jobserver automatically when you log on, you just have to save one shortcut to the start scripts in the startup folder.

The scripts are:

```
%BICSUITEHOME%\bin\RUN_JOBSERVER_LOCALHOST.BAT
%BICSUITEHOME%\bin\RUN_JOBSERVER_HOST_1.BAT
%BICSUITEHOME%\bin\RUN_JOBSERVER_HOST_2.BAT
```

Installation with Postgres

Introduction

This guide does not claim to precisely describe the installation of the database system. Details about this can be found in the Postgres documentation. Normally, though, this guide should be adequate for performing a "standard" installation. For typesetting reasons some lines in the examples are shown with line breaks. This is evident where the line has an indent and there is no prompt at the beginning of the line. When copying the commands, you should *not* use a return at the end of the line.

Installation

1. Download and install the latest version of Postgres

At the end of the installation, use stackbuilder to install the Postgres JDBC Driver (pg JDBC) as well.

The Postgres password you chose during the installation will be required again later.

Note:

If the Postgres service does not start, it may be blocked by your firewall or antivirus software. Running the Postgres service as an administrator may remedy this problem.

2. Create the Postgres user `bicsuite`

Open a Windows command line window and navigate to your Postgres installation directory.

Example:

```
C:> cd /d C:\Program Files\PostgreSQL\9.2\bin
```

Run the postgres createuser command to create the user bicsuite who is also permitted to create databases:

Example:

```
C:> cd /d C:\Program Files\PostgreSQL\9.2\bin
C:> .\createuser -U postgres -W -P -d bicsuite
```

Use the Postgres password (see 1.).

Remember the bicsuite password you have entered here.

3. Create the repository database bicsuitedb

If you have not already done so, open a Windows command line window and navigate to your Postgres installation directory. Run the postgres createdb command to create a database under the name bicsuitedb.

Example:

```
C:> cd /d C:\Program Files\PostgreSQL\9.2\bin
C:> .\createdb -U bicsuite -W -O bicsuite bicsuitedb
```

Use the assigned bicsuite password (see 5.1.2.).

4. Create and initialise the database tables

Open a Windows command line window if you have not already done so. Navigate to the BICSuite sql directory.

Example:

```
C:> cd /d "%BICSUITEHOME%\sql
```

Run the postgres psql command to initialise the BICSuite repository.

Example:

```
C:> C:\Program Files\PostgreSQL\9.2\bin\psql -U bicsuite
-f pg\install.sql bicsuitedb
```

Use the assigned bicsuite password (see 5.1.2.).

5. Configure the database connection in the BICSuite Server configuration file

Open a Windows command line window if you have not already done so. If you have not set the environment variable %BICSUITECONFIG%, use the location %BICSUITEHOME%\etc instead.

Edit %BICSUITECONFIG%\server.conf and modify the following properties:

```
DbPasswd=bicsuite password
DbUrl=jdbc:postgresql:bicsuitedb
DbUser=bicsuite
JdbcDriver=org.postgresql.Driver
```

The hostname must be the same as the computer's hostname:

Entering `echo %USERDOMAIN%` in a Windows command line window will return this name.

Hostname=<enter the hostname here>

6. Configure the BICsuite Java Class Path for the Postgres JDBC

Open a Windows command line window if you have not already done so. If you have not set the environment variable BICSUITECONFIG, use the location %BICSUITEHOME%\etc instead of %BICSUITECONFIG%.

Edit %BICSUITECONFIG%\JAVA_CONF.BAT so that the environment variable BICSUITEJDBC is set to postgres JDBC jar.

Example:

```
SET DBMSHOME=C:\Program Files\PostgreSQL
SET BICSUITEJDBC=%DBMSHOME%\pgJDBC\postgresql-9.2-1004.jdbc4.jar
```

Installation with MySQL

Introduction

This guide does not claim to precisely describe the installation of the database system. Details about this can be found in the MySQL documentation. Normally, though, this guide should be adequate for performing a "standard" installation. For typesetting reasons some lines in the examples are shown with line breaks. This is evident where the line has an indent and there is no prompt at the beginning of the line. When copying the commands, you should *not* use a return at the end of the line.

Installation

1. Download and install the latest version of MySQL Community Version

The Instance Configuration Wizard is launched after the software has been downloaded.

Select:

- a) Detailed Configuration
→ Next
- b) Developer Machine
→ Next
- c) Transactional Database Only
→ Next
- d) Retain InnoDB tablespace Settings
→ Next
- e) Manual Setting (10 connections are more than adequate)
→ Next

- f) Add firewall exception for port
Port 3306, retain Enable TCP and Enable Strict Mode
→ Next
 - g) Retain Standard Character set
→ Next
 - h) Retain Install AS Windows Service
Set Include bin directory in Windows path
→ Next
 - i) enter root_password (remember this!)
→ Next
→ Execute
2. Create the MySQL user `bicsuite` and the database `bicsuitedb`
- Open a Windows command line window and run the MySQL command line utility.
- ```
C:> mysql --user=root --password=root_password
```
- Create the user `bicsuite` and make a note of the password.
- ```
mysql> create user bicsuite identified by 'bicsuite_password';
```
- Create the database.
- ```
mysql> create database bicsuitedb;
```
- Grant the user `bicsuite` all privileges to this database.
- ```
mysql> grant all on bicsuitedb.* to bicsuite;
```
3. Create and initialise the database tables
- Open a Windows command line window if you have not already done so.
- Run the following commands:
- ```
C:> cd /d %BICSUITEHOME%/sql
C:> mysql --user=bicsuite --password=bicsuite_password
--database=bicsuitedb --execute="source mysql\install.sql"
```
4. Download and unpack the MySQL (Connector/J) JDBC Driver
- e.g. under `C:\Program Files\MySQL`
- The MySQL JDBC jar archive can then be found e.g. under:
- ```
C:\Program Files\MySQL\mysql-connector-java-5.1.16
```
5. Configure the database connection in the BICsuite Server configuration file
- Open a Windows command line window if you have not already done so. If you have not set the environment variable `BICSUITECONFIG`, use the location `%BICSUITEHOME%\etc` instead of `%BICSUITECONFIG%`.

Edit %BICSUITECONFIG%\server.conf and modify the following properties:

```
DbPasswd=bicsuite_password
DbUrl=jdbc:mysql:///bicsuitedb
DbUser=bicsuite
JdbcDriver=com.mysql.jdbc.Driver
```

The hostname must be the same as the computer's hostname:

Entering `echo %USERDOMAIN%` in a Windows command line window will return this name.

Hostname=<enter the hostname here>

6. Configure the BICsuite Java Class Path for the MySQL JDBC

Open a Windows command line window if you have not already done so. If you have not set the environment variable BICSUITECONFIG, use the location %BICSUITEHOME%\etc instead of %BICSUITECONFIG%.

Edit %BICSUITECONFIG%\JAVA_CONF.BAT so that the environment variable BICSUITEJDBC is set to MySQL JDBC jar.

Example:

```
SET JARHOME=C:\Program Files\MySQL\mysql-connector-java-5.1.16
SET BICSUITEJDBC=%JARHOME%\mysql-connector-java-5.1.16-bin.jar
```

Installation with Microsoft™ SQL Server

Introduction

This guide does not claim to precisely describe the installation of the database system. Refer to the SQL Server documentation for more information. Normally, though, this guide should be adequate for performing a "standard" installation. For typesetting reasons some lines in the examples are shown with line breaks. This is evident where the line has an indent and there is no prompt at the beginning of the line. When copying the commands, you should *not* use a return at the end of the line.

Installation

1. Download and install the latest version of SQL Server Express

During the installation, configure the combined (SQL and Windows) authentication.

2. Create the MySQL user bicsuite and the database bicsuitedb

Open a Windows command line window and run the SQL Server command line utility.

```
C:> sqlcmd -S %USERDOMAIN%\SQLEXPRESS
```

Create the user bicsuite with:

```
1> sp_addlogin 'bicsuite','bicsuite_password'
2> go
1> EXEC master..sp_addsrvrolemember @loginame = N'bicsuite',
    @rolename = N'dbcreator'
2> go
```

Close the SQL Server command line utility:

```
1> exit
```

Start the SQL Server command line utility under the user bicsuite and create the database:

```
C:> sqlcmd -S %USERDOMAIN%\SQLEXPRESS -U bicsuite -P bicsuite_password
1> create database bicsuitedb
2> go
```

Close the SQL Server command line utility:

```
1> exit
```

3. Create and initialise the database tables

Open a Windows command line window if you have not already done so. Run the following commands:

```
C:> cd /d %BICSUITEHOME%\sql
C:> sqlcmd -S %USERDOMAIN%\SQLEXPRESS -U bicsuite -P bicsuite_password
    -d bicsuitedb -i mssql\install.sql
```

4. Download and install the JDBC driver for SQL Server

Install it under C:\Program Files as recommended by Microsoft.

5. Configure the database connection in the BICsuite Server configuration file

Open a Windows command line window if you have not already done so. If you have not set the environment variable BICSUITECONFIG, use the location %BICSUITEHOME%\etc instead of %BICSUITECONFIG%.

Edit %BICSUITECONFIG%\server.conf and modify the following properties:

```
DbPasswd=bicsuite_password
DbUrl=jdbc:sqlserver://localhost;database=bicsuitedb;instance=SQLEXPRESS
DbUser=bicsuite
JdbcDriver=com.microsoft.sqlserver.jdbc.SQLServerDriver
```

The hostname must be the same as the computer's hostname:

Entering `echo %USERDOMAIN%` in a Windows command line window will return this name.

```
Hostname=<enter the hostname here>
```

6. Configure the BICsuite Java Class Path for the SQL Server JDBC

Open a Windows command line window if you have not already done so. If you have not set the environment variable BICSUITECONFIG, use the location %BICSUITEHOME%\etc instead of %BICSUITECONFIG%.

Edit %BICSUITECONFIG%\JAVA_CONF.BAT so that the environment variable BICSUITEJDBC is set to SQL Server JDBC jar.

Example:

```
SET DBMSHOME=C:\Programme\Microsoft SQL Server JDBC Driver 3.0
SET BICSUITEJDBC=%DBMSHOME%\sqljdbc_3.0\enu\sqljdbc4.jar
```

7. Configure the TCP/IP access to the SQL Server

Open the SQL Server Configuration Manager from the start menu:

Start → All Programs → Microsoft SQL Server 2008 R2 → Configuration Tools
→ SQL Server Configuration Manager

Now perform the following steps:

- a) Under SQL Server Network Configuration, select Protocols for SQLEXPRESS
- b) Double-click TCP/IP to edit the TCP/IP connection properties
- c) Set Activated to "Yes"
- d) Switch to the IP Address tab
- e) Under IPALL, clear the Dynamic TCP Ports box
- f) Under IPALL, set the TCP Port box to 1433

Installation with Ingres

Introduction

This guide does not claim to precisely describe the installation of the database system. Details about this can be found in the Ingres documentation. Normally, though, this guide should be adequate for performing a "standard" installation. For typesetting reasons some lines in the examples are shown with line breaks. This is evident where the line has an indent and there is no prompt at the beginning of the line *not* use a return at the end of the line.

Installation

1. Download and install the latest version of Ingres Community Edition

In the Ingres setup wizard, select:

- a) Retain Transactional System
→ Next
- b) Retain Express
→ Next
- c) Retain Typical Server
→ Next
→ Install

2. Create the user `bicsuite`

There are several ways of registering the user `bicsuite` in the Ingres system. The first method involves creating the user with the help of the tool `accessdb`. This method is not explained here any further.

Another method is to create the user using SQL commands. To do this, start the SQL Terminal Monitor as the user `ingres`:

```
C:> sql iidbdb
INGRES TERMINAL MONITOR Copyright 2008 Ingres Corporation
Ingres Linux Version II 9.2.1 (a64.lnx/103)NPTL login
Mon Jun 13 10:05:19 2011

continue
* create user bicsuite with privileges = (createdb);
* \g
Executing . . .

continue
* commit;\g
Executing . . .

continue
* \q
Ingres Version II 9.2.1 (a64.lnx/103)NPTL logout
Mon Jun 13 10:07:58 2011
```

3. Create the repository database `bicsuitedb`

If you have not already done so, open a Windows command line window and create the repository database as the user `bicsuite`.

```
C:> createdb bicsuitedb
```

4. Create and initialise the database tables

Open a Windows command line window if you have not already done so. Navigate to the BICsuite `sql` directory.

```
C:> cd /d "%BICSUITEHOME%\sql
```

Run the `ingres sql` command to initialise the BICsuite repository.

```
C:> sql bicsuitedb < ing\install.sql
```

5. Configure the database connection in the BICsuite Server configuration file

Open a Windows command line window if you have not already done so. If you have not set the environment variable BICSUITECONFIG, use the location %BICSUITEHOME%\etc instead of %BICSUITECONFIG%.

A password must be assigned for your Windows user!

Edit %BICSUITECONFIG%\server.conf and modify the following properties:

```
DbPasswd=<windows password>
DbUrl=jdbc:ingres://localhost:II7/bicsuitedb;
DbUser=bicsuite
JdbcDriver=com.ingres.jdbc.IngresDriver
```

Replace <windows password> with the Windows password for bicsuite.

The hostname must be the same as the computer's hostname:

Entering `echo %USERDOMAIN%` in a Windows command line window will return this name.

```
Hostname=<enter the hostname here>
```

6. Configure the BICsuite Java Class Path for the Ingres JDBC

Open a Windows command line window if you have not already done so. If you have not set the environment variable BICSUITECONFIG, use the location %BICSUITEHOME%\etc instead of %BICSUITECONFIG%.

Edit %BICSUITECONFIG%\JAVA_CONF.BAT so that the environment variable BICSUITEJDBC is set to Ingres JDBC jar.

Example:

```
SET BICSUITEJDBC=%II_SYSTEM%\ingres\lib\iijdbc.jar
```

Installation with DB2

Introduction

This guide does not claim to precisely describe the installation of the database system. Details about this can be found in the DB2 documentation. Normally, though, this guide should be adequate for performing a "standard" installation.

For typesetting reasons some lines in the examples are shown with line breaks. This is evident where the line has an indent and there is no prompt at the beginning of the line. When copying the commands, you should not use a return at the end of the line.

Installation

1. Download and installation of the actual IBM DB2 Express-C version

2. des bicsuite Users `bicsuite`

DB2 hat keine eigene Benutzerverwaltung. Aus diesem Grund ist es nötig einen Betriebssystem- Benutzer anzulegen, welcher zur Authentifizierung bei Datenbank-Connects dient. In diesem Beispiel wurde der Windows Benutzer `bicsuite` mit Passwort `bicsuitepw` verwendet.

3. Anlegen der Datenbank `bicsuitedb`

Öffnen Sie eine Windows-Eingabeaufforderung und "öffnen Sie eine Subshell für den Setup der nötigen Umgebung auf

```
C:>db2cmd -i -w db2clpsetcp
```

Rufen Sie das DB2 Command Line Utility auf.

```
C:> db2
```

Führen Sie folgende Kommandos aus:

```
db2> create database bicsuite restrictive
db2> connect to bicsuite
db2> grant dbadm, dataaccess, accessctrl, secadm on database
      to user bicsuite
db2> terminate
```

Schließen Sie die mit DB2cmd geöffnete Subshell.

```
C:>exit
```

Anmerkungen:

Das Anlegen der Datenbank dauert sehr lange, also haben Sie etwas Geduld.

Der Datenbankname darf maximal 8 Zeichen lang sein.

4. Anlegen und Initialisierung der Datenbanktabellen

Falls noch nicht geschehen, öffnen Sie eine Windows-Eingabeaufforderung. Führen Sie folgende Kommandos aus:

```
C:> cd /d %BICSUITEHOME%\sql
C:> clpplus -nw bicsuite/bicsuitepw@localhost/bicsuite
      @db2/install.sql > install.log
```

Überprüfen Sie das `install.log`, ob Fehler aufgetreten sind.

5. Konfigurieren der Datenbankverbindung in der BICsuite Server Konfigurationsdatei

Falls noch nicht geschehen, öffnen Sie eine Windows-Eingabeaufforderung. Falls Sie keine Umgebungsvariable BICSUITECONFIG gesetzt haben verwenden Sie statt %BICSUITECONFIG% %BICSUITEHOME%\etc

Editieren Sie %BICSUITECONFIG%\server.conf und ändern Sie folgende Properties wie folgt:

```
DbPasswd=bicsuitepw
DbUrl=jdbc:db2://localhost:50000/bicsuite
DbUser=bicsuite
JdbcDriver=com.ibm.db2.jcc.DB2Driver
```

Hostname muss auf Hostnamen des Rechners gesetzt werden: In der Windows-Eingabeaufforderung liefert das Kommando hostname diesen Namen.

```
Hostname=<hier den hostname einsetzen>
```

6. Konfigurieren Sie den BICsuite Java Class Path für IBM DB2 JDBC

Falls noch nicht geschehen, öffnen Sie eine Windows-Eingabeaufforderung. Falls Sie keine Umgebungsvariable BICSUITECONFIG gesetzt haben verwenden Sie statt %BICSUITECONFIG% %BICSUITEHOME%\etc.

Bearbeiten Sie %BICSUITECONFIG%\JAVA_CONF.BAT so, dass die Umgebungsvariable BICSUITEJDBC auf das IBM DB2 JDBC jar gesetzt wird.

Beispiel:

```
SET BICSUITEJDBC=C:\Programme\IBM\SQLLIB\java\db2jcc4.jar
```

Installation with Oracle Express Edition

Introduction

This guide does not claim to precisely describe the installation of the database system. Details about this can be found in the Ingres documentation. Normally, though, this guide should be adequate for performing a "standard" installation. For typesetting reasons some lines in the examples are shown with line breaks. This is evident where the line has an indent and there is no prompt at the beginning of the line *not* use a return at the end of the line.

Installation

1. Download and install the latest version of Oracle Express Edition

Make a note of your Oracle system password.

2. Create the Oracle user bicsuite

Open a Windows command line window and run the Oracle SQLPlus Server command line utility.

```
C:> sqlplus SYSTEM/oracle_system_password
```

Create the user bicsuite with:

```
SQL> create user bicsuite identified by bicsuite_password;
```

Assign the required access privileges:

```
SQL> grant CONNECT, RESOURCE, CREATE VIEW, CREATE PROCEDURE  
      TO bicsuite;
```

3. Create and initialise the database tables

Open a Windows command line window if you have not already done so. Navigate to the BICsuite sql directory and run the following commands:

```
C:> cd /d %BICSUITEHOME%\sql  
C:> sqlplus bicsuite/bicsuite_password @ora/install.sql
```

4. Configure the database connection in the BICsuite Server configuration file

Open a Windows command line window if you have not already done so. If you have not set the environment variable BICSUITECONFIG, use the location %BICSUITEHOME%\etc instead of %BICSUITECONFIG%.

Edit %BICSUITECONFIG%\server.conf and modify the following properties:

```
DbPasswd=bicsuite_password  
DbUrl=jdbc:oracle:thin:@hostname:1521:XE  
DbUser=bicsuite  
JdbcDriver=oracle.jdbc.OracleDriver
```

The hostname must be the same as the computer's hostname in both the example above and the configuration file: Entering `echo %USERDOMAIN%` in a Windows command line window will return this name.

```
Hostname=<enter the hostname here>
```

5. Configure the BICsuite Java Class Path for the Oracle JDBC

Open a Windows command line window if you have not already done so. If you have not set the environment variable BICSUITECONFIG, use the location %BICSUITEHOME%\etc instead of %BICSUITECONFIG%.

Edit %BICSUITECONFIG%\JAVA_CONF.BAT so that the environment variable BICSUITEJDBC is set to Oracle JDBC jar.

Example:

```
SET BICSUITEJDBC=%ORACLE_HOME%\jdbc\lib\ojdbc14.jar
```

Installing the Zope server

Introduction

A Zope Application Server has to be installed before you can use the BICsuite!Web user interface.

For typesetting reasons some lines in the examples are shown with line breaks. This is evident where the line has an indent and there is no prompt at the beginning of the line *not* use a return at the end of the line.

Some files from the distribution are required to install the BICsuite!Web user interface. The first three steps of the BICsuite Server installation have to be performed before Zope can be installed.

Installation (Zope2)

1. Download and install Python 2.7 from www.python.org

The default directory for the Python installation is `C:\Python27`. In our example, we are using the installation directory `C:\Program Files\Python27`. In the following we refer to this directory as `PYTHONDIR`.

2. Download and install Python setuptools

Important: Make sure you use the correct version for Python 2.7 !

3. Optionally, download and install pywin32

Important: Make sure you use the correct version for Python 2.7 !

This step is only necessary if the BICsuite!Web Zope server is to be started as a service.

4. Install the Zope2 software

Install the latest release of Zope2. In a browser, visit <http://download.zope.org/Zope2/index> to find the latest release. At the time this document was written, 2.13.29 was the most recent release.

Open a Windows command line window. Navigate to the virtual Python environment for the Zope installation and install Zope2.

Example:

```
C:> cd /d C:\Program Files\Python27
C:> Scripts\pip install -r https://raw.githubusercontent.com/zopefoundation/Zope/2.13.
```

Note:

In some versions of the Windows operating system, this command returned an error the first time it was called (`rmdir` directory not empty) and the command had to be repeated.

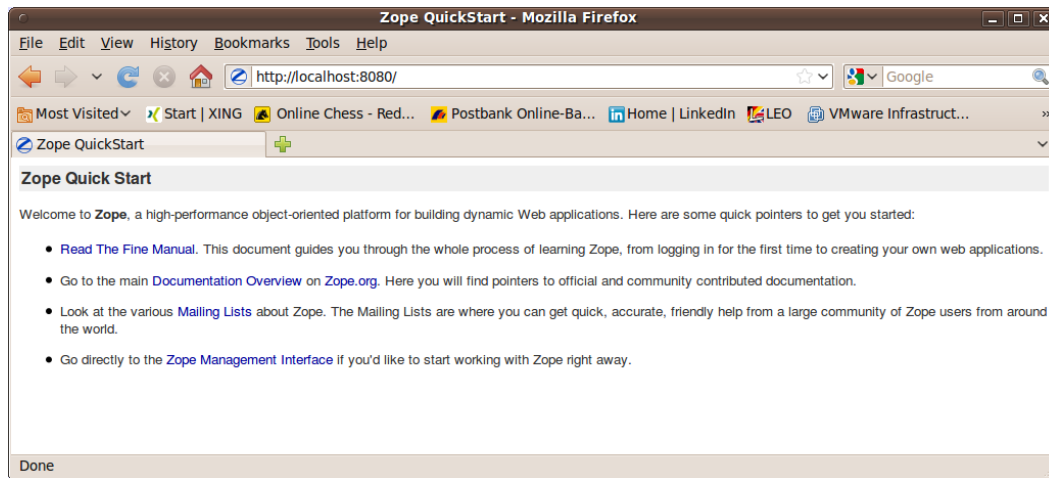


Figure 3.1: Zope Quicks Start Seite

5. Create a Zope instance for BICsuite!Web

Open a Windows command line window. Navigate to the virtual Python environment for the Zope installation and create the Zope instance `bicsuiteweb`. The used password here can be replaced by any password of your own choice. The username must be `sdmsadm` though.

Example:

```
C:> cd /d C:\Program Files\Python27
C:> Scripts\mkzopeinstance -d ..\bicsuiteweb -u sdmsadm:sdmsadm_password
```

Important:

If Oracle has been installed on this computer, the Zope standard port 8080 may already be in use. In this case, the port will have to be changed in the file

`bicsuiteweb\etc\zope.conf`

Here it is assumed that there are no conflicts concerning the use of the port and that Zope can be addressed at port 8080.

Test:

```
C:> cd /d C:\Program Files\bicsuite
C:> bicsuiteweb\bin\runzope
```

In the browser, the URL <http://localhost:8080> should now display the Zope quick start page as shown in Figure 3.1.

Zope can now be exited again with `CRTL+C` or by closing the Windows command line window.

6. Install the BICsuite!Web components

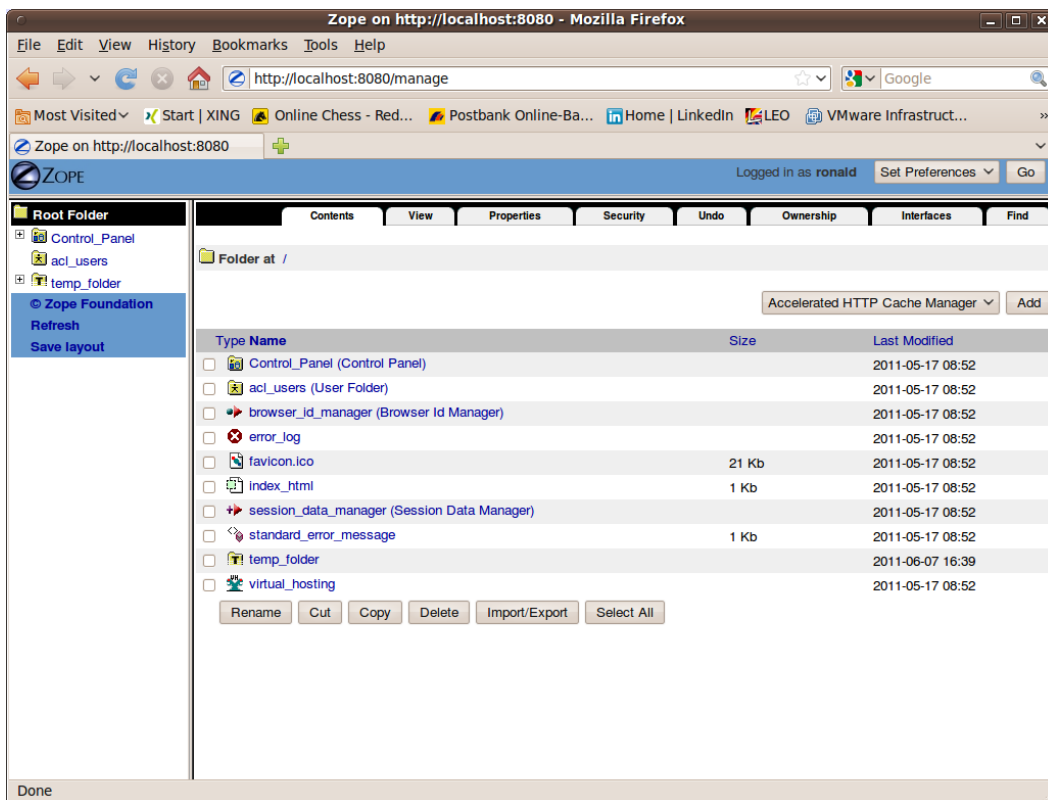


Figure 3.2: Zope Management Oberfläche

The Zope installation has to be extended with several components before the BICSuite!Web components can be installed.

```
C:> cd /d C:\Program Files\bicsuite\bicsuiteweb
C:> mkdir Extensions
C:> cd Extensions
C:> copy "%BICSUITEHOME%\zope\*.py" .
C:> cd ..\Products
C:> mkdir BICsuiteSubmitMemory
C:> cd BICsuiteSubmitMemory
C:> copy "%BICSUITEHOME%\zope\BICsuiteSubmitMemory\*.py" .
C:> cd ..\..\import
C:> copy "%BICSUITEHOME%\zope\SDMS.zexp" .
```

The Zope instance now has to be started again so to register the changes on the Zope side.

Open a Windows command line window.

```
C:> cd /d C:\Program Files\bicsuite\bicsuiteweb\bin
C:> runzope
```

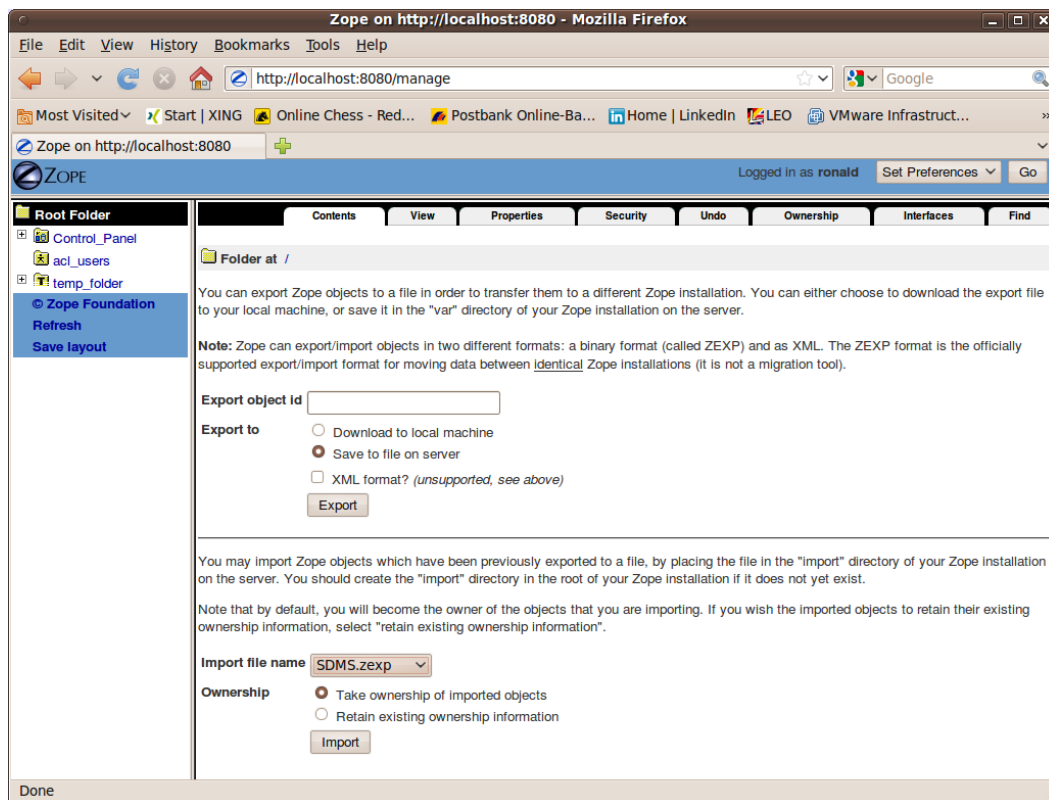


Figure 3.3: Zope Import Dialog

The Zope Management user interface, like in Figure 3.2 is now opened at the address

`http://localhost:8080/manage`

in a browser. This is done with the user `sdmsadm` together with the password you have assigned (in this guide this is `sdmsadm_password`).

The front end software is now loaded into Zope (Import button, see also Figure 3.3):

- Import it into the folder `/ SDMS.zexp`.
- In the folder `/SDMS/Install`, mark and copy the folders `User` and `Custom`.
- Create the folders `User` and `Custom` in the folder `/` by pasting them.

If everything has been carried out without any errors, the interface will look like the screenshot as in Figure 3.4.

7. Configure the server connections

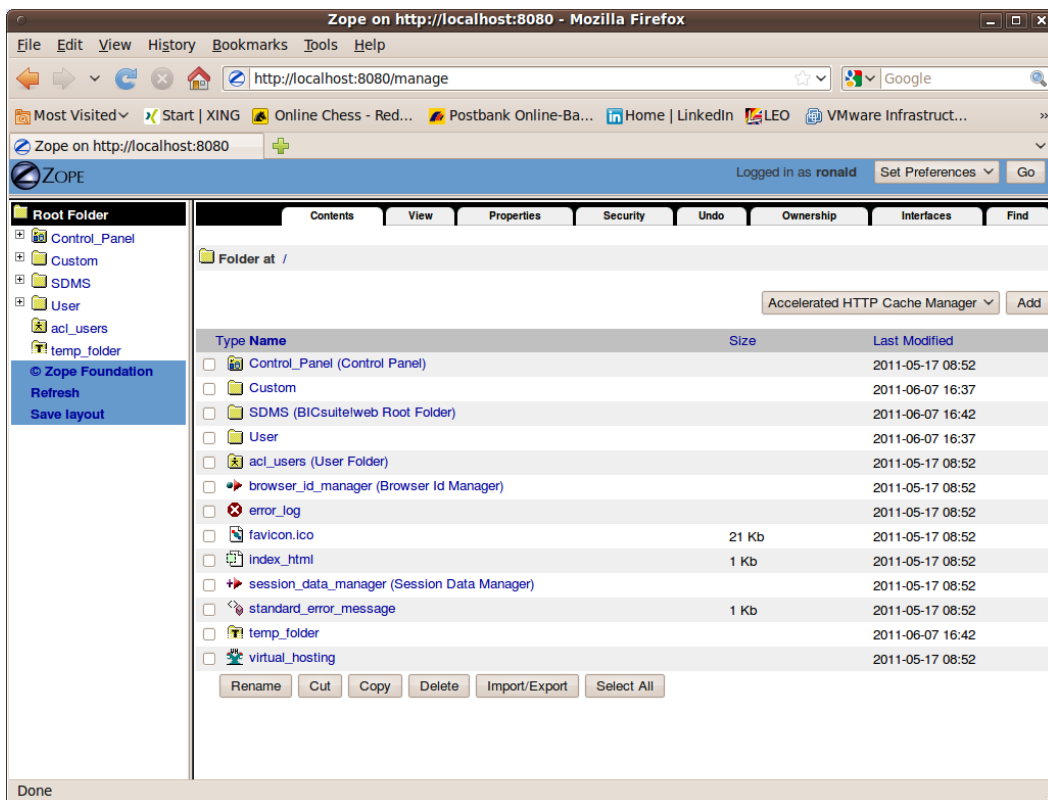


Figure 3.4: Zope result window

The server connections are also configured in the Zope Management user interface. To do this, log on as the user `sdmsadm`.

The Python script `SDMSServers` is edited in the Custom folder. This script returns a dictionary which has to contain an entry for every BICsuite!Server that is to be addressed by this BICsuite!Web installation. The entry looks like this:

```
# Server name that identifies the server in the BICsuite!Web
# user interface
'servername' : {
    # IP address or hostname at which the BICsuite!Server is running
    'HOST'      : 'hostname',

    # Port at which the BICsuite!Server is addressed
    'PORT'      : '2506',

    # BASIC, PROFESSIONAL, ENTERPRISE
    'VERSION'   : 'BASIC',
```

```

# Optional property stating whether BICsuite!Web should
# cache the server connections
'CACHE'      : 'Y'

# Optional property stating for how long cached BICsuite!Web
# server connections should continue to be valid
# Default setting is 60 seconds, only significant if 'CACHE' : 'Y'
'TIMEOUT'    : '60'
}

```

An entry with the name `DEFAULT` must exist for bootstrapping. This entry can be deleted after the user has been set up (who obviously should not then use this connection).

If a server is to be addressed via a secure SSL connection, the following additional properties have to be defined as well:

```

# Connection is set up via Secure Socket Layer
'SSL'        : 'true',

# If stated, the identity of the BICsuite! Server is verified
# The stated file must contain the BICsuite!Server server certificate
'TRUSTSTORE' : 'truststore.pem',

# If the BICsuite!Server requires client authentication,
# this property must be defined and the stated file
# must contain the client's certificate and private key.
# The certificate must be known to the server in its trust store.
'KEYSTORE'   : 'keystore.pem'

```

Note:

When using SSL, for performance reasons it is advisable to use cached server connections because setting up a secure connection is a resource-intensive operation.

8. Open the BICsuite!Web interface

The user interface is now available at the address

```
http://localhost:8080/SDMS
```

A logon prompt is displayed when this page is opened. When the user has logged on, the application is started by clicking the "Take Off" button.

How to work with the interface is described in the relevant documentation.

9. Automatically start the BICsuite!Web interface

To have BICsuite!Web interface start automatically when you log on, create a shortcut to

```
C:\Program Files\bicsuite\bicsuiteweb\bin\runzope.bat
```

in your startup folder.

Installation (Zope4+)

1. Download and install Python 3 from www.python.org

The at the time of this writing actual version of python3 can be downloaded from:

<https://www.python.org/ftp/python/3.9.2/python-3.9.2-amd64.exe>

In our example we use C:\Programme\Python3 as installation dirctory. In the following we refer to this dirctory as PYTHONDIR.

2. Download and install the Microsoft Visual C++ Build Tools

The Microsoft Visual C++ Build Tools required for the installation can be downloaded from:

<https://visualstudio.microsoft.com/de/thank-you-downloading-visual-studio/?sku=BuildTools&rel=16>

Run the executable to install the Build Tools.

3. Creating of a python3 virtual environment

For the following installation steps, open a windows console window with administrator privileges.

The Zope5 installaton will be created in C:\Users\BICsuite\Zope5. A different dirctory can be chosen. We refer to this diractory as ZOPE5DIR,

```
C:> set INSTALLDIR=C:\Users\BICsuite
C:> set ZOPE5ENV=Zope5
C:> set ZOPE5DIR=%INSTALLDIR%\%ZOPE5ENV%
C:> cd %INSTALLDIR%
C:> python -m venv %ZOPE5ENV%
```

4. Installation of the Zope5 software

Open an administrator windows console if not already open. Change your working diractory to the python virtual environment created in the previous step and install Zope5. At the time of this writing the current stable release of Zope 5 was Zope 5.1.2.

```
C:> set INSTALLDIR=C:\Users\BICsuite
C:> set ZOPE5ENV=Zope5
C:> set ZOPE5DIR=%INSTALLDIR%\%ZOPE5ENV%
C:> cd %ZOPE5DIR%
C:> Scripts\pip install -U pip wheel
C:> Scripts\pip install Zope[wsgi]==5.0 ^
-c https://zopefoundation.github.io/Zope/releases/5.1.2/constraints.txt
C:> Scripts\pip install Products.ExternalMethod
C:> Scripts\pip install Products.Sessions
C:> Scripts\pip install Products.SiteErrorLog
C:> Scripts\pip install Products.PythonScripts
```

5. Create the Zope instance for BICsuite!Web

In this example we will create the Zope instance at:

```
C:\Users\BICsuite\Zope5Instance
```

A different directory can be chosen. We refer to this directory as ZOPE5INSTANCE.

```
C:> set INSTALLDIR=C:\Users\BICsuite
C:> set ZOPE5ENV=Zope5
C:> set ZOPE5DIR=%INSTALLDIR%\%ZOPE5ENV%
C:> set ZOPE5INSTANCE=%INSTALLDIR%\Zope5Instance
C:> cd %INSTALLDIR%
C:> %ZOPE5ENV%\Scripts\mkwsgiinstance -d %ZOPE5INSTANCE% ^
-u sdmsadm:sdmsadm_password
```

Test:

Start the Zope5 instance by running the following command:

```
C> %ZOPE5DIR%\Scripts\runwsgi -v %ZOPE5INSTANCE%\etc\zope.ini
```

Your browser should show a 'Zope Auto-generated default page' for the Url <http://localhost:8080>.

Pressing Ctrl-C or closing the console windows will stop the running Zope instance.

Remarque: The Zope5 release used in this example was creating a flawed zope.ini file. This could be solved easily by editing the created zope.ini file. Replace all '\\' in path names by '/'.

6. Installation of the BICsuite!Web components

To install the BICsuite!Web components, the Zope5 instance has to be extended by additional components. We assume for this installation procedure that the environment variable BICSUITEHOME is set correctly.

```
C:> set INSTALLDIR=C:\Users\BICsuite
C:> set ZOPE5ENV=Zope5
C:> set ZOPE5DIR=%INSTALLDIR%\%ZOPE5ENV%
C:> set ZOPE5INSTANCE=%INSTALLDIR%\Zope5Instance
C:> cd %INSTALLDIR%
C:> xcopy %BICSUITEHOME%\zope4\Extensions %ZOPE5INSTANCE%\Extensions\
C:> xcopy %BICSUITEHOME%\zope4\Products\BICsuiteSubmitMemory ^
%ZOPE5DIR%\Lib\site-packages\Products\BICsuiteSubmitMemory\
C:> xcopy %BICSUITEHOME%\zope4\Products\StringFixer ^
%ZOPE5DIR%\Lib\site-packages\Products\StringFixer\
C:> mkdir %ZOPE5INSTANCE%\import
C:> copy %BICSUITEHOME%\zope4\import\SDMS.zexp %ZOPE5INSTANCE%\import
```

Now the Zope instance has to be started again to enable the additional components.

```
C> %ZOPE5DIR%\Scripts\runwsgi -v %ZOPE5INSTANCE%\etc\zope.ini
```

The Zope Management user interface is now opened at the address

`http://localhost:8080/manage`

in a browser. This is done with the user `sdmsadm` together with the password you have assigned (in this guide this is `sdmsadm_password`).

The front end software is now loaded into Zope (Import button):

- a) Import it into the folder / `SDMS.zexp`.
- b) In the folder /`SDMS/Install`, mark and copy the folders `User` and `Custom`.
- c) Create the folders `User` and `Custom` in the folder / by pasting them.

7. Configure the server connections

The server connections are also configured in the Zope Management user interface. To do this, log on as the user `sdmsadm`.

The Python script `SDMSServers` is edited in the `Custom` folder. This script returns a dictionary which has to contain an entry for every BICsuite Server that is to be addressed by this BICsuite!Web installation. The entry looks like this:

```
# Server name that identifies the server in the BICsuite!Web
# user interface
'servername' : {
    # IP address or hostname at which the BICsuite!Server is running
    'HOST'      : 'hostname',

    # Port at which the BICsuite!Server is addressed
    'PORT'      : '2506',

    # BASIC, PROFESSIONAL, ENTERPRISE
    'VERSION'   : 'BASIC',

    # Optional property stating whether BICsuite!Web should
    # cache the server connections
    'CACHE'     : 'Y'

    # Optional property stating for how long cached BICsuite!Web
    # server connections should continue to be valid
    # Default setting is 60 seconds, only significant if 'CACHE' : 'Y'
    'TIMEOUT'   : '60'
}
```

An entry with the name `DEFAULT` must exist for bootstrapping. This entry can be deleted after the user has been set up (who obviously should not then use this connection).

If a server is to be addressed via a secure SSL connection, the following additional properties have to be defined as well:

```
# Connection is set up via Secure Socket Layer
'SSL'      : 'true',

# If stated, the identity of the BICsuite! Server is verified
# The stated file must contain the BICsuite!Server server certificate
'TRUSTSTORE' : 'truststore.pem',

# If the BICsuite!Server requires client authentication,
# this property must be defined and the stated file
# must contain the client's certificate and private key.
# The certificate must be known to the server in its trust store.
'KEYSTORE'  : 'keystore.pem'
```

Note:

When using SSL, for performance reasons it is advisable to use cached server connections because setting up a secure connection is a resource-intensive operation.

8. Open the BICsuite!Web interface

The user interface is now available at the address

```
http://localhost:8080/SDMS
```

A logon prompt is displayed when this page is opened. When the user has logged on, the application is started by clicking the "Take Off" button.

How to work with the interface is described in the relevant documentation.

9. Install and start Zope5 as Windows Service

To install and start the Zope5 instance as a Windows Service, execute the following commands in a windows console with administrator privileges. We assume that the environment variable BICSUITEHOME is set correctly.

```
C:> set INSTALLDIR=C:\Users\BICsuite
C:> set ZOPE5ENV=Zope5
C:> set ZOPE5DIR=%INSTALLDIR%\%ZOPE5ENV%
C:> set ZOPE5INSTANCE=%INSTALLDIR%\Zope5Instance
C:> sc create Zope5 ^
start= auto ^
binPath= "%BICSUITEHOME%\bin\scrolllog -c Zope5
        -w %ZOPE5INSTANCE% log\logfile
        -e %ZOPE5DIR%\Scripts\runwsgi
        -v %ZOPE5INSTANCE%\etc\zope.ini" ^
displayName= "Zope5 BICsuite!Web Application Server"
C:> sc start Zope5
```

Remark: The line for the option binPath has been wrapped for layout reasons. When executing the sc command, the binPath must be given on a single line.