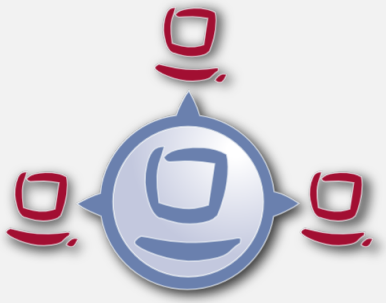

opsi Getting Started opsi-Version 4.0.7

uib gmbh
Bonifaziusplatz 1b
55118 Mainz
Tel.:+49 6131 275610
www.uib.de
info@uib.de

Stand: December 17, 2018

Contents

Chapter 1

Copyright

Chapter 2

Introduction

These instructions explain in detail the installation and starting of an opsi-server. It starts from the provided installation package and leads to the test installation of a client.

The network configuration described here is exemplary and relates to a network without a concurrent DHCP server (i.e. the first trials are done in an isolated test network with an opsi-server and a few clients).

We strongly recommend that the first trials of opsi be done in a test network, separated from other DHCP servers. A temporary connection to the main network can be used for downloads of actual product packages.

uib provides consulting services for the integration of opsi into your existing production environment.

2.1 Steps for Installation and Getting Starting

The four steps to install and start an opsi-server are:

1. base installation of the server
2. configuration of the server:
   configuration of the network, setting passwords (opsi administrator, pcpatch, samba), server updates
3. download and installation of the required opsi client products to the opsi server
4. completion of the system software base packages for Windows using the original Windows DVD or other sources.

At that point, a client can be automatically integrated into the opsi server.

Depending on your requirements, opsi offers different types of base installations. The procedure to perform different types of base installations are described in Chapter Chapter 4 of this manual. The types of installations included offer the possibility to either use an existing VMware machine or to perform a direct installation of opsi to the host machine.

2.2 Hardware Requirements

For a opsi-server the following hardware is recommended:

- Intel-x86-compatible PC
- 2GB RAM or higher
- a hard disk with 60 GB capacity or more

The requirements of the server are moderate in testing environments. In the case of production environments it is recommended to increase the capabilities of the host system.

We recommend in the case of testing with a Virtual machine, that the host computer should have at least a dual core processor and at least 4GB of RAM. For testing purposes, a test client can be run as another Virtual machine on the same host computer.

2.3 Configuration Requirement

Your server and your network have to comply the following requirements to install and work with opsi:

- **valid DNS domain name**
  Your DNS domain name should contain at least a domain name and a top level domain. So the full qualified domain name (FQDN) should contain one or more dots. The top level domain must contain at least two chars.
  Valid domain names are e.g.: *domain.local* , *uib.de*, *subdomain.domain.de*. An invalid example: *mydomain.d* because this is only one character at the top-level domain An invalid example: *mydomain* because this is only a top-level domain
  see also:
  http://en.wikipedia.org/wiki/Domain_name

- **valid DNS hostname**
  The hostnames (also the client hostnames) have to follow standard naming rules. They may contain the ASCII letters a-z, digits 0-9 and the hyphen -. No underscores are allowed.
  see also:
  http://en.wikipedia.org/wiki/Hostname

- **correct name resolution for the server**
  Execute the following command and check the result:

```
getent hosts $(hostname -f)
```

  The result should look like the following example:
  192.168.1.1 server.domain.tld server
  The result has the scheme:
  <IP-Number> <full qualified hostname> <hostname>
  If the result looks different from the above example (contains eg. *127.0.0.1* or *localhost*), or the full qualified hostname does not contain one or more dots, then you must correct your name resolution (DNS or `/etc/hosts` file).

Note
The names must be in accordance of the rules of a DNS system but a DNS server is not required for the usage of opsi.

Note
opsi does not required an *Active Directory* or similar. Integrating opsi is possible but not required.

Check needed network ports at: Section 4.8

Chapter 3

opsi Support Matrix (opsi runs on which server)

Here an overview on which distributions and releases the opsi-server will run:

3.1 Supported distributions for server

(Stand / as of 25.08.2018)

Distribution	Opsi 4.1	Opsi 4.0.7
Debian 9 *Stretch*	✔	✔
Debian 8 *Jessie*	✔	✔
Debian 7 *Wheezy*	✖	⚠
Ubuntu 18.04 LTS *Bionic Beaver*	✔	✖
Ubuntu 16.04 LTS *Xenial Xerus*	✔	✔
Ubuntu 14.04 LTS *Trusty Tahir*	✖	✔
Ubuntu 12.04 LTS *Precise Pangolin*	✖	⚠
RHEL 7	✔	✔
RHEL 6	✖	✔
CentOS 7	✔	✔
CentOS 6	✖	✔
SLES 15	🚶	✖
SLES 12SP3	✔	✔
SLES 12SP2	✔	✔
SLES 12SP1	✔	✔
SLES 12	✔	✔

SLES 11SP4	✗	✓
SLES 11SP3	✗	⚠
openSuse Leap 15	🚧	✗
openSuse Leap 42.3	✓	✓
openSuse Leap 42.2	✗	⚠
openSuse Leap 42.1	✗	⚠
openSuse 13.2	✗	⚠
UCS 4.3	✓	✗
UCS 4.2	✓	✓
UCS 4.1	✗	⚠
UCS 4.0	✗	⚠
UCS 3.3	✗	✗
UCS 3.2	✗	⚠

✓: Supported ✗: Unsupported 🚧: Under development ⚠: Discontinued

Chapter 4

Installation

4.1 opsi-server Base Installation

This chapter describes different types of opsi-server installations. You can choose your installation type and skip the other instructions.

If all required steps are successful, then the server system is properly configured and ready to start. Before you run the opsi-server, you should update your system according to the chapter *Update of the opsi-server*.

For evaluation purposes, we recommend using the Virtual-Machine provided by uib.

Please follow the instructions by entering the commands in the

```
marked fields
```

(via cut-and-paste from this document)

If you encounter any problems, please ask for help at https://forum.opsi.org

4.1.1 Starting up the uib preconfigured Virtual Machine

An *opsi-server* can be installed as a virtual machine, because the load on the system is low. A ready-to-use and pre-configured virtual machine is provided by uib. You can download the VMware or Virtualbox files from the uib website. The free of charge VMware player or Virtualbox is sufficient to run this machine.

You may also use VMware ESX.

4.1.1.1 First Start

VMware

If you have a server running VMware or a VMware player, it only takes a few mouse clicks to install a base *opsi-server*:

- Download the opsi-ServerVM from http://uib.de/de/opsi/opsi-testen-download/
- Unzip the file and a directory *opsivm* will be generated.
- Start the VMware player. Open "Open a Virtual Machine" and search for the file `opsivm.ovf` in the *opsivm* directory. You can import the server with an new name. The virtual machine will still boot.

ESXi-Server

- Download the opsi-ServerVM from http://uib.de/de/opsi/opsi-testen-download/

- Unzip the file and a directory *opsivm* will be generated.
- Start the vSphere Client.
  Install a new client with *File* / *Deploy OVF Template...* and answer the following questions.

Virtualbox

- Download the opsi-ServerVM from http://uib.de/de/opsi/opsi-testen-download/
- Unzip the file and a directory *opsivm* will be generated.
- Start the Virtualbox.
  At the menu *File* / *Import Appliance* select your `opsivm.ovf` file and import it.

General

The VMware player is free of charge and available for all common operating systems at vmware.com. Usually it can be installed without any problems, as long as the resources of the host computer (especially memory) meet the needs of running software systems in parallel.

4.1.1.2 Language selection

The first step is to choose the preferred language:

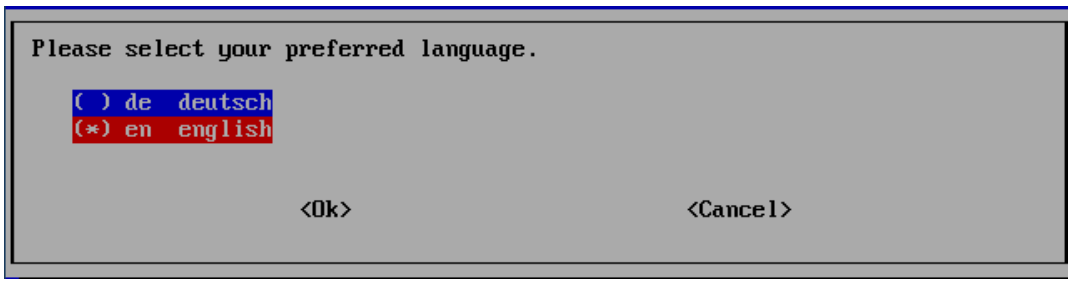

Figure 4.1: Language selection

4.1.1.3 First boot

The opsi-server needs to be connected to the Internet to work properly. The script `1stboot.py` will automatically start at the first boot in order to configure the opsi-server network settings.

If something goes wrong while running *1stboot.py*, then you may run `1stboot.py` again from the command line.

Warning
You cannot use `1stboot.py` to rename your *opsi-server* afterwards!

The log file of `1stboot.py` is located at `/var/lib/1stboot/1stboot.log`.

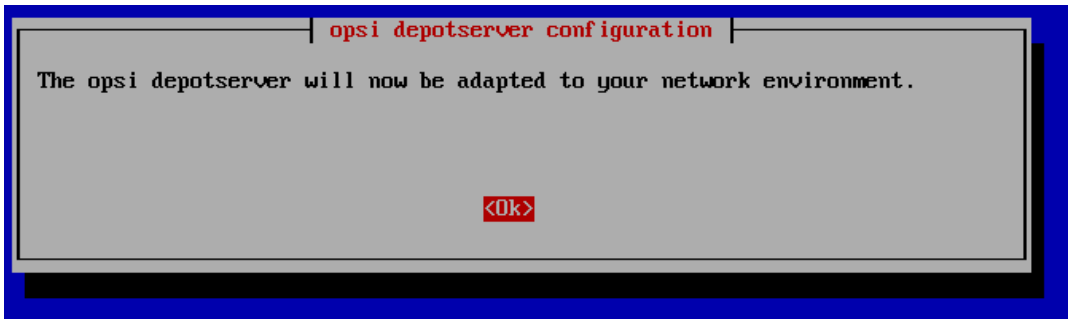

Figure 4.2: Startup mask

Fill in the configuration information for your network and answer the questions.

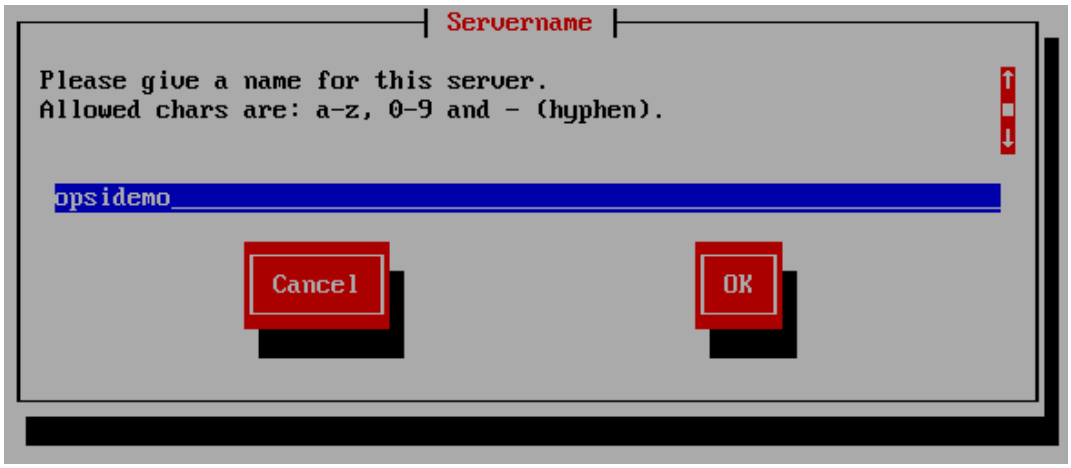

Figure 4.3: Input mask

In the following, you will be asked for:

server name
Name of this server (without domain) e.g. `opsidemo`

domain
DNS-Domain (not Windows-Domain) the name has to include a dot e.g. `opsi.local`

ip address
Address of this server e.g. `192.168.1.50`

netmask
Net mask of this server e.g. `255.255.255.0`

windows domain
Name of the Windows Domain (not the DNS domain)

country
For the creation of the SSL-certificate: Identification of the nation (2 capital letter) e.g. `DE`

state
For the creation of the SSL-certificate: Identification of the federal state e.g. `RPL`

city
For the creation of the SSL-certificate: Identification of the city e.g. `Mainz`

organization
For the creation of the SSL-certificate: Identification of the company e.g. `uib gmbh`

organizational unit
For the creation of the SSL-certificate: Identification of the bureau (optional)

email address
For the creation of the SSL-certificate: mail address (optional)

gateway
IP-address of the Internet gateway e.g. `192.168.1.1`

proxy
If required for the Internet access the proxy information: e.g. `http://myuser:mypass@192.168.1.5:8080`

DNS server
ip address of the name server e.g. `192.168.1.1`

mail relay
ip address of the mail server e.g. `192.168.1.1`

tftp server
ip of the tftp server (usually the server)

Password of root
Password of root

Password of adminuser
Password of local opsi-admin.

After the program *1stboot.py* finishes, the server VMware machine will be rebooted.

4.1.1.4 Second Start

After the reboot, login as *adminuser* with your password.

The graphical user interface of the opsi-server should have already started (implemented as a sustainable window manager). A "Firefox" browser window might appear at startup, and display further instructions and information. This information can serve as a reference to the getting started document (the document you are currently reading).

If you get a message that there is no network connection, try rebooting the server. This might solve the problem.

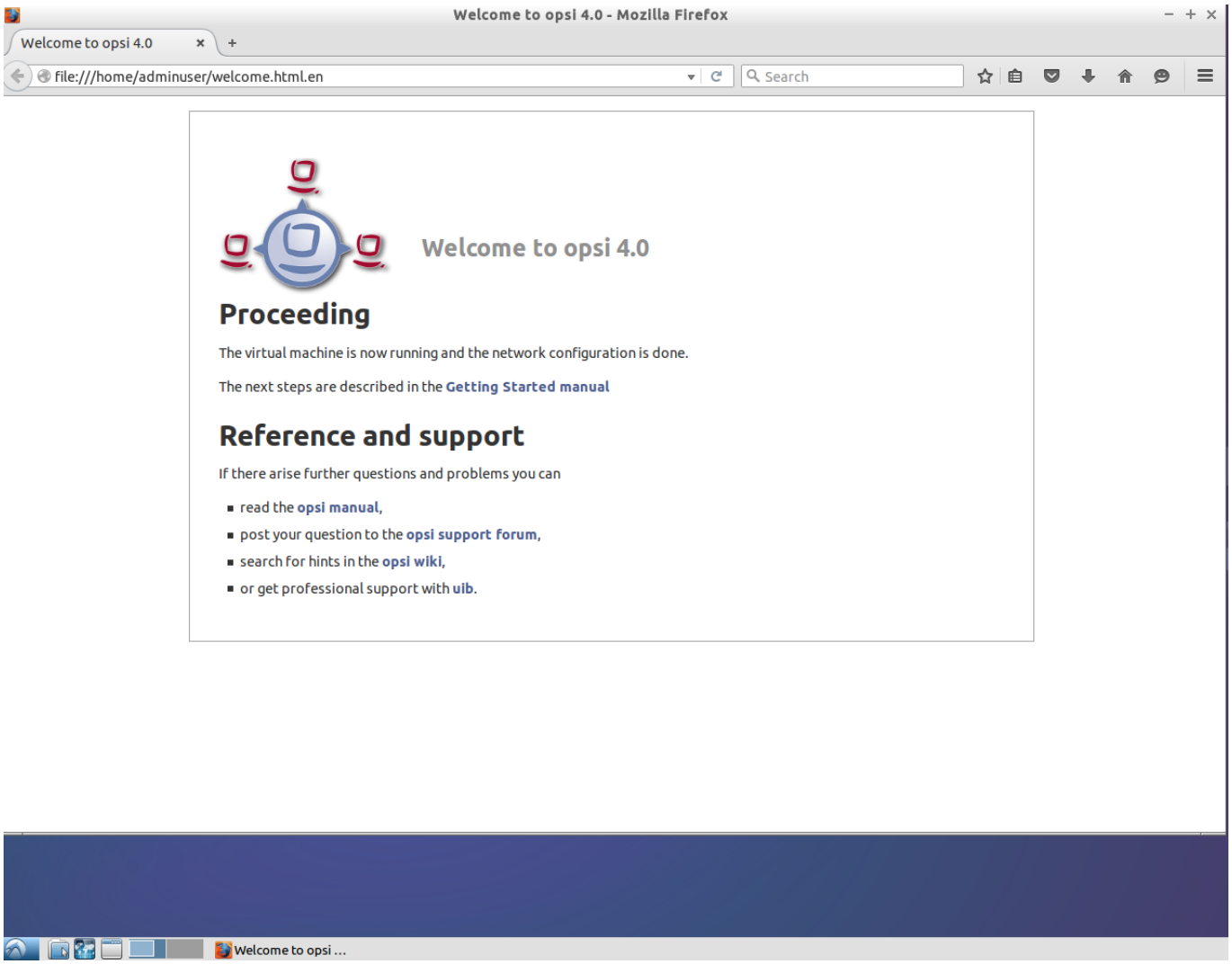

Figure 4.4: View of fresh started opsi-server

If the network was correctly configured in the previous steps, then you should be able to remotely access the opsi-server, for example:

- use *ssh* at the command line to access to the server (*ssh* should already be installed on linux systems, for Windows use putty: http://www.chiark.greenend.org.uk/~sgtatham/putty/)

Use *root* as the user name, and authenticate with the root password.

4.1.1.5 Terminal Window

In the following sections, some commands have to be entered into a command line interface. It may be the easiest way to work through these instructions.

The commands are input into a window called a "terminal window". Here are examples that explain how to access a terminal window:

- Remote access per ssh on the *opsi-server* (see Section 3.1.1.4 of the last chapter)

- Open a terminal window in the opsi-server graphical interface with a click on the terminal icon in the icon bar.
- Open a terminal window in the opsi-server graphical interface with a right mouse click inside the interface, and choose "Terminal".
  Note: the graphical interface has many working applications that are reachable using the variety of buttons in the upper-left-hand corner of the display.

We recommend cutting and pasting commands from this handbook directly into the opsi-server terminal window (most applications support cut and paste).

Example snippets from configuration files are formatted like this:

```
depoturl = smb://smbhost/sharename/path
```

Example snippets for commands that you have to execute are formatted like this:

```
cd /tmp
ls -l
```

Angle brackets < > mark abstract names. When entering commands, please replace the *<abstract name>* with a real name.
For example: The file share, where opsi places the software packages, may abstractly be noted as *<opsi-depot-share>*. If the real file share is `/var/lib/opsi/depot`, then you have to replace the abstract name by this exact string. The location of the package *<opsi-depot-share>/ooffice* becomes `/var/lib/opsi/depot/ooffice`. .

4.1.1.6 Check the Network Connection

If the network configuration is correct, and the computer is connected to the Internet, then you can access any Internet address using the browser in the start window.

If the Internet connection is not working, then you have to open a terminal window (maybe remote access isn't possible, except using the server terminal window) and then perform the necessary network connection checks and fixes.

You can re-enter the network configuration by entering this command in the terminal window:

```
1stboot.py
```

A reboot is forced with the command:

```
reboot
```

If the network connection works, then you can install opsi packages or update them, and configure the environment for the first installation test. If you want to use the VMware machine (and not install the opsi-server directly to your host system), then skip to Section 4.2.

4.1.1.7 Update the opsi-Server

To update your opsi-Server you need to double click the Icon "Update Manager" on the desktop.

4.1.1.8 Install the standard opsi-products

By performing a double click the Icon *opsi-product-updater FirstRun* the minimal opsi Products will be installed. To do this please introduce the current password for the adminuser. Through this installation the actual stock of opsi-products, incl. templates for Windows- and Linux-OS, will be downloaded from http://download.uib.de/-opsi4.0/products/ and installed on the server. For more information you can see here Section 4.5.

4.1.1.9 Starting opsi-Server Interface

For a description of the Server Interface check Section 4.7.

You have a running opsi server now, i.e. the opsi application itself is fully configured.

You can now proceed with:

- Chapter 5
- Chapter 6

4.1.2 Installation on a Debian / Ubuntu

Important
Check here if your Server Operating System Version is supported by opsi: Chapter 3
Please note the Configuration Requirements: Section 2.3.

In this chapter, we assume you are familiar with the debian-package system (you will find information about the debian-package in the appropriate Debian books, in the manual pages or in the Debian documentation.

Please note that an opsi-server needs storage in `/var/lib/opsi`. A minimum of 16 GB of free space is recommended.

For the following installation commands, aptitude should be installed:

```
apt-get install aptitude
```

We recommend the following software installations:

```
aptitude install wget lsof host p7zip-full cabextract openbsd-inetd pigz
```

opsi needs samba, which can be installed as:

```
aptitude install samba samba-common smbclient cifs-utils
```

If you plan to use MySQL as a backend (i.e. for Inventory and license management), then you should install a mysql-server:

```
aptitude install mysql-server
```

Check the opsi-server entry in `/etc/hosts`, or the output of

```
getent hosts $(hostname -f)
```

The result should look like the following example:
192.168.1.1 server.domain.tld server
The result has the scheme:
<IP-Number> <full qualified hostname> <hostname>
If the result looks different than the above example (contains eg. *127.0.0.1* or *localhost*), or the full qualified hostname does not contain one or more dots, then you must correct your name resolution (DNS or `/etc/hosts` file).

To start with the installation of opsi create the file
/etc/apt/sources.list.d/opsi.list with the following content in it depending
on your operating system:

Ubuntu 16.04 LTS *Xenial Xerus*:

```
deb http://download.opensuse.org/repositories/home:/uibmz:/opsi:/opsi40/xUbuntu_16.04 ./
```

Ubuntu 14.04 LTS *Trusty Tahir*:

```
deb http://download.opensuse.org/repositories/home:/uibmz:/opsi:/opsi40/xUbuntu_14.04 ./
```

Debian 9 *Stretch*:

```
deb http://download.opensuse.org/repositories/home:/uibmz:/opsi:/opsi40/Debian_9.0 ./
```

Debian 8 *Jessie*:

```
deb http://download.opensuse.org/repositories/home:/uibmz:/opsi:/opsi40/Debian_8.0 ./
```

Execute the following command in order to import the signature key of the repository:

Ubuntu 16.04 LTS *Xenial Xerus*:

```
wget -O - http://download.opensuse.org/repositories/home:/uibmz:/opsi:/opsi40/xUbuntu_16.04/Release.key | apt-key add -
```

Ubuntu 14.04 LTS *Trusty Tahir*:

```
wget -O - http://download.opensuse.org/repositories/home:/uibmz:/opsi:/opsi40/xUbuntu_14.04/Release.key | apt-key add -
```

Debian 9 *Stretch*:

```
wget -O - http://download.opensuse.org/repositories/home:/uibmz:/opsi:/opsi40/Debian_9.0/Release.key | apt-key add -
```

Debian 8 *Jessie*:

```
wget -O - http://download.opensuse.org/repositories/home:/uibmz:/opsi:/opsi40/Debian_8.0/Release.key | apt-key add -
```

All:
Check for key import success:

```
apt-key list
```

should contain the output:
`pub 2048R/D8361F81 2017-09-30 [verfällt:  2019-12-09] uid home:uibmz:opsi OBS Project <home:uibmz:opsi@build.opensuse.org>`

Execute the following commands in order to install opsi at your server:

```
aptitude update
aptitude safe-upgrade
aptitude remove tftpd
update-inetd --remove tftpd
aptitude install opsi-atftpd
aptitude install opsi-depotserver
aptitude install opsi-configed
aptitude install wimtools
```

If you are asked for the tftp directory during the tftpd-installation answer with `/tftpboot`. If you are asked about multicast support answer with *no*.

During the installation of the opsiconfd, you will be asked for information for the creation of a local SSL certificate.

During the opsi-server installation, you have to allow the patching of the file `smb.conf`. Answer the question with *yes*. Also, you will be asked for a password for the user *pcpatch*. Set a new password, and please remember this password when continuing with the following sections.

If you would like to run the opsi management interface *opsi-configed* directly on the server, then you need to install the Java Runtime Environment. In order to install these packages, enter these commands:

Debian: In the file `/etc/apt/sources.list`, add the branches `non-free` and `contrib` to your repositories.

The result may look like this:

```
deb http://ftp.de.debian.org/debian/ lenny main non-free contrib
deb-src http://ftp.de.debian.org/debian/ lenny main non-free contrib

deb http://security.debian.org/ lenny/updates main non-free contrib
deb-src http://security.debian.org/ lenny/updates main non-free contrib
```

Note
If you are running *squeeze*, then the repositories have to be *squeeze* instead of *lenny*.

To install the Java JRE execute:

```
aptitude update
aptitude install openjdk-7-jre icedtea-7-plugin
```

Ubuntu Trusty: Since Oneric, we recommend using version 7 of the OpenJDK. To install version 7 of the Java JRE execute:

```
aptitude update
aptitude install openjdk-7-jre icedtea-plugin
```

Ubuntu Xenial: To install version 8 of the Java JRE execute:

```
aptitude update
aptitude install openjdk-8-jre icedtea-plugin
```

Debian 8 (Jessie) specialities: The bootimage has issues to mount the `opsi_depot`-Share over mount.cifs. To avoid these problems you can either configure ID mapping in Samba or disable `winbind`. If you do not rely on `winbind` we recommend to disable the daemon.

Disable starting of winbindd:

```
systemctl disable winbind
```

or

```
insserv -r winbind
```

For a sane ID mapping configuration specify a configuration with limited mapping range in `smb.conf` and then restart Samba.

To configure ID mapping you can insert the following into the `[global]` section of `smb.conf`.

```
idmap config * : range = 1000-1999999
```

Assuming all of the above steps completed successfully, we can assume that the network is properly configured. Next continue on with Section 4.2.3

4.1.3 Installation on a Univention Corporate Server (UCS)

Important
Check here if your Server Operating System Version is supported by opsi: Chapter 3
Please note the Configuration Requirements: Section 2.3.

The installation on a Univention Corporate Server is possible through the Univention App Center as well as on the classic way by using the repositories maintained by uib.

Note
Installations on a system with 32bit and 64bit are currently both supported.
In the future the support for 32bit will be dropped. Therefore we recommend to only use opsi on a system with a 64bit architecture.

4.1.3.1 Installation through Univention App-Center

In the Univention App-Center an automatic installation of the opsi-Server is available. The installation-app for opsi can be found in the UCS-Management-Webinterface in the category "System". Over the App-Center opsi can be installed on UCS4-Serverroles Master and Backup. If you want to update a existing opsi4ucs-Installation please check the next Chapter for further information.
The following will describe what will be happen during a installation:

- The pre-requirements for a MySQL-Installation will be fulfilled.
- The MySQL-Backend will be configured to use for inventory data in opsi (dispatch.conf will be modified) if this is the first opsi-server in the environment.
- The opsi-Samba-Shares will be checked, if they are configured in the right way to work with opsi.
- Required packages for the opsi-Installation will be installed: opsi4ucs, opsi-atftpd, p7zip-full, cabextract, mysql-server.
- *opsi-product-updater* will be called to download and install the minimal opsi-Packages for the first work with the new installation.
  If an existing config server is detected *opsi-product-updater* will be configured to do so from the config server.

Please note that no automatic transfer of clients to opsi takes place. More information at Section 4.1.3.6.

After the described steps the opsi-Installation on a UCS-Server over the Univention App-Center is finished. Please continue with following chapter: Section 4.6

4.1.3.2 Upgrading an existing opsi-Installation from UCS 3 to UCS 4 (over the App-Center)

Since opsi 4.0.5 the group *opsifileadmins* replaces the group *pcpatch* in UCS. This group has already been introduced with the support of UCS 3.0, but only on installations that had Samba 4 and the Univention Directory Services (Samba4-AD). In all other Variants and roles the group continues to be, as it was before *pcpatch*.

Since this situation represents a problem not only on the installation, but could also lead to potential problems with migrations (especially of Samba3 on Samba4) since the release of 4.0.5 the group *pcpatch* will be created as *opsifileadmins*.

Warning
To implement the integration package in a clean way, an already existing group *pcpatch* will be renamed automatically to *opsifileadmins*. This is done via the join script. If your config server is run on a Master or backup, the join script will be executed automatic.

The main reason for this drastic measure is that the manual rename of this group is not trivial, because it comes to be a primary group. Therefore it is recommended before installing this update to make sure that your group is still named *pcpatch*. If so, the update should be started with the config server and soon afterwards on the depot servers as well. Otherwise the operation on multi depot-environments could lead to issues. This should not be the case, if your group is already named *opsifileadmins*. Nevertheless, it is recommended after importing the update to check every opsi server to verify complete functionality.

4.1.3.3 Manual opsi-installation on UCS (without App-Center)

Make sure the conditions in Section 2.3 are fulfilled.

Necessary preparations:

- The command

```
hostname -f
```

  must return a full qualified domain name containing two dots, e.g. *opsiserver.domain.local*

- The command

```
getent hosts $(hostname -f)
```

  has to show the IP-adress of the network interface which the client has to connect with. If the command shows the address *127.0.0.1* or *127.0.0.2*, you have to correct your name resolution in `/etc/hosts` file.

- Samba has to be configured. For the use of a server with the member role `univention-samba` has to be used instead of `univention-samba4`.
- If the machine should act as a Configserver a MySQL-server should to be installed.
- If the machine should also work as DHCP-server, then the daemon dhcpd has to be configured and should be running.

The installation of opsi is possible on the roles master, backup, slave and member. For the installation on a member you need to read Section 4.1.3.4!

The following documentation describes an installation on a master with Samba4.

Caution
When installing on role *Slave* the server must be already joined and Samba 4 has to be installed first.
UCS configuration is usually done on the *master* while the installation and configuration of opsi happen on the *slave*.

The classic installation with the user: *pcpatch* in the primary group: *pcpatch* does not work with UCS. Samba4 has placed fundamental restrictions on the Active-Directory, so groups with the same name as a user are no longer allowed. For this reason the configuration file `/etc/opsi/opsi.conf` has been introduced. This file controls how the group used for Samba access will be named. More specifically for UCS the group name *pcpatch* will be renamed to *opsifileadmins*. This means that users that need clearance for opsi (opsi package builder for example) can't be members of the group *pcpatch* but must be member of *opsifileadmins*. This peculiarity applies only to UCS and is different to other distributions and chapters in the opsi-documentation. Furthermore, since UCS 3 the user *pcpatch* is created as an domain user. For more information about this new configuration file please refer the opsi-manual.

- Next add the opsi4ucs repository:*

UCS 4.2:

```
echo 'deb http://download.opensuse.org/repositories/home:/uibmz:/opsi:/opsi40/Univention_4.2/ /' > /etc/apt/sources.\
    list.d/opsi.list
```

Now import the key to the repository system with the following command:

UCS 4.2

```
wget -O - http://download.opensuse.org/repositories/home:/uibmz:/opsi:/opsi40/Univention_4.2/Release.key | apt-key add \
    -
```

For the installation the following commands must be entered:

```
univention-install opsi-atftpd
univention-install opsi4ucs
univention-install cabextract mysql-server p7zip-full wimtools
```

If the role of the target systems different than Master or Backup then the following commands run the opsi4ucs Join-Script:

```
univention-run-join-scripts
```

A link to the management interface can be found at the URL `https://<servername>:4447`.

Finally the *opsi_depot* release point must be released in UDM. To realize this settings set the link to *yes* under Advanced Settings → Advanced Samba Settings: *follow symlinks*. The same should be done for the *opsi_depot_rw*, so the driver integration will run without problems. If the directory */var/lib/opsi/depot* was found on an extra partition or hard disk then the option for wide links should be set to *yes*.

To make sure that opsi is running with the proper settings restart opsi by entering the following commands:

```
opsi-setup --init-current-config
opsi-setup --set-rights
service opsiconfd restart
service opsipxeconfd restart
```

Since UCS 3 there is no direct contact between the Univention LDAP backend and opsi. Clients must first be created in LDAP using opsi over udm including all system information (in particular the MAC address). Deleting the LDAP clients in Univention does not mean that the client was also deleted under opsi. A solution for this is in Section 4.1.3.6.

Since opsi has already been run on the server we assume that the network configuration is correct.
Continue with the installation be skipping forward to Section 4.2.3.

Warning
The Unix commands used in the following chapters are for Debian systems. You may have to change them to match your Linux system.

4.1.3.4 Hints about installing opsi on an UCS server with the role *member*

Warning
Running opsi on a member server is affected by certain limitations. Therefore we recommend beginners to run their opsi systems on a different role.

Installing opsi on a server with the role member is possible. However an automated installation through the Univention App Center is currently not possible.

After an installation you need to make sure that the setting for the user that will be used to access the depot is set with the current domain. Control the host parameter `clientconfig.depot.user` for this. Let's assume that the domain is `backstage`, then the value has to be `backstage\pcpatch`. If it is `memberserver\pcpatch` then it has to be changed.

Setting the password for the user `pcpatch` through `opsi-admin` fails because of the missing AD write access of a member server. To change the password you have to do so **additionally** on a server with write access - a master, backup or slave.

4.1.3.5 PXE-Boot configuration for operating system installation

If the PXE-Boot should used for OS installations the DHCP-service on the relevant UCS-System has to be reconfigured. There are two characteristics which UCS differentiate from other supported distributions.

- The configuration is not made automatically during the opsi installation on an active UCS-Infraestructure because the configuration already exists.
- The opsi-atftpd is not configured as usual using the directory `/tftpboot` as base directory, instead the `/var/lib/univention-client-boot` is used. All important files of opsi-linux-bootimage will be linked from */tftpboot* in the base directory. The side effect is that the DHCP-Option file named `pxelinux.0` will be replace instead with `linux/pxelinux.0`.

You have to set guide lines to realized the mentioned configurations at the UCS-System. These guide lines are depend on existent guide lines and has to realize appropriate. If opsi was installed on an UCS-test system without existing guide lines you need an installed DHCP-service at first. If the DHCP-service is installed already the easiest way to create guide lines in the UMC-web interface (Univention Management Console) is from UCS-server. Therefore choose the category "Domain" and subjacent the module DHCP-server. As next you have to choose the service (in a testing system you find only one entry usally). In the following detail views choose guide lines in the menu. The needed guide line is a DHCP-Boot guide line. During the guide line configuration dial as default entry `cn=default-settings` (sould be the only entry) and choose *edit*. Under the normal settings DHCP-boot for the option Bootserver enter the IP from opsi-server and insert as boot-file name `pxelinux.0`.

Warning
If the guide lines are configured like mentioned above, this configurations </opsi-getting-started-installation-config-backend>be affected for every device which is operated with DHCP from this server. So once again the description is only conceived for evaluation purposes which will be also testing not only opsi but also UCS. In a productive UCS environment you should never configure the guide lines during the installation as described previously.

Optional you can run the udm-commands at the console. You can find more informations in the UCS-documentation.

4.1.3.6 Synchronising data from LDAP to opsi

In an opsi4ucs installation Windows-Clients have to be created in the UDM first and in a second step they have to be created in opsi. Changes to the client in UDM will not be passed on to opsi. For example if a client's MAC address changes in LDAP and in opsi a netboot-product is set to setup, the boot configuration would be provided with a wrong MAC address.

A solution for this is the extension *opsi-directory-connector*. Please consult the manual for more information.

4.1.4 Installation on openSUSE

Important
Check here if your Server Operating System Version is supported by opsi: Chapter 3
Please note the Configuration Requirements: Section 2.3.

Necessary preparations:

- The command

```
hostname -f
```

have to return a full qualified domain name containing two dots, e.g. *opsidemo.domain.local*

- The command

```
getent hosts $(hostname -f)
```

have to send the IP-adress of the network gateway the client connected with. If the command send the result *127.0.0.1* or *127.0.0.2* then file `/etc/hosts` has to be corrected.

- Samba has to be installed and configured.
- If the machine should act as configserver `mariadb-server` should be installed.
- If the machine should also act as DHCP-server then the daemon dhcpd has to be configured and be active.

You can use zypper to add the opsi-SUSE-Repository:

openSUSE Leap 42.3:

```
zypper addrepo http://download.opensuse.org/repositories/home:uibmz:opsi:opsi40/openSUSE_Leap_42.3/home:uibmz:opsi:\
    opsi40.repo
zypper refresh
```

After adding the repository, you may start the opsi installation:

```
zypper refresh
  Do you want to reject the key, trust temporarily, or trust always? [r/t/a/? shows all options] (r): a
zypper -v install opsi-depotserver opsi-configed
zypper -v install wimtools
chkconfig opsiconfd on
chkconfig opsipxeconfd on
service xinetd restart
chkconfig xinetd on
opsi-setup --auto-configure-samba
service smb restart
service nmb restart
service opsiconfd restart
service opsipxeconfd restart
```

Please make sure that your firewall configuration allows access to the tftp Port (69/UDP) and the opsi ports (4447/TCP and 4441/TCP).

If you used a tool like yast or autoyast to help you with your network configuration, then the tool may have created an entry in your `/etc/hosts` file that has the following pattern:

```
127.0.0.2 <fqdn> <hostname>
```

If you want opsi to manage the configuration of the DHCP server, then you need to correct this entry to point to the server's public IP adress.

Assuming all of the above steps completed successful we can assume that the network is properly configured. Next continue on with Section 4.2.3

Warning
The unix commands used in the following chapters are working on Debian systems. You may have to change them to match your linux system.

4.1.5 Installation on Suse Linux Enterprise Server (SLES)

Important
Check here if your Server Operating System Version is supported by opsi: Chapter 3
Please note the Configuration Requirements: Section 2.3.

Necessary preparations:

- The command

```
hostname -f
```

  send back a full qualified domain name containing two dots, e.g. *opsidemo.domain.local*

- Check the opsi-server entry in `/etc/hosts` or the output of

```
getent hosts $(hostname -f)
```

  have to send the IP-adress of the network gateway the client connected with. If the command send the result *127.0.0.1* or *127.0.0.2* then file `/etc/hosts` has to be corrected.

- Samba has to be configured.
- `python-pyasn1` needs to be installed.
- If the machine should also act as DHCP-server then the daemon dhcpd has to be configured and be running.

You can use zypper to add the SLES-Repository:

SLES 12SP3:

```
zypper addrepo http://download.opensuse.org/repositories/home:uibmz:opsi:opsi40/SLE_12_SP3/home:uibmz:opsi:opsi40.repo
zypper mr -p 50 home_uibmz_opsi_opsi40
```

SLES 12SP2:

```
zypper addrepo http://download.opensuse.org/repositories/home:uibmz:opsi:opsi40/SLE_12_SP2/home:uibmz:opsi:opsi40.repo
zypper mr -p 50 home_uibmz_opsi_opsi40
```

SLES 12SP1:

```
zypper ar http://download.opensuse.org/repositories/home:/uibmz:/opsi:/opsi40/SLE_12_SP1/home:uibmz:opsi:opsi40.repo
zypper mr -p 50 home_uibmz_opsi_opsi40
```

SLES 12:

```
zypper ar http://download.opensuse.org/repositories/home:/uibmz:/opsi:/opsi40/SLE_12/home:uibmz:opsi:opsi40.repo
zypper mr -p 50 home_uibmz_opsi_opsi40
```

SLES 11SP4:

```
zypper ar http://download.opensuse.org/repositories/home:/uibmz:/opsi:/opsi40/SLE_11_SP4/home:uibmz:opsi:opsi40.repo
zypper mr -p 50 home_uibmz_opsi_opsi40
```

After adding the repository, you may start the opsi installation:

```
zypper refresh
  Do you want to (r)eject the Key, (t)emporary or (a)lways trust? [r/t/a/?] (a): a
zypper -v install opsi-depotserver opsi-configed
zypper -v install wimtools
chkconfig opsiconfd on
chkconfig opsipxeconfd on
service xinetd restart
opsi-setup --auto-configure-samba
service smb restart
service nmb restart
service opsiconfd restart
service opsipxeconfd restart
```

In case you used a tool like yast or autoyast to help you with your network configuration it's possible the tool may have created an entry in your `/etc/hosts` file like:

```
127.0.0.2 <fqdn> <hostname>
```

If you want opsi to manage the configuration of the DHCP server, then you need to correct this entry to point to the server's public IP adress.

Assuming all of the above steps completed successful we can assume that the network is properly configured. Next continue on with Section 4.2.3

Warning
The unix commands used in the following chapters are working on Debian systems. You may have to change them to match your linux system.

4.1.6 Installation on RedHat Enterprise Linux (RHEL)

Important
Check here if your Server Operating System Version is supported by opsi: Chapter 3
Please note the Configuration Requirements: Section 2.3.

Necessary preparations:

- The command

  ```
  hostname -f
  ```

  send back a fully qualified domain name containing two dots, e.g. *opsidemo.domain.local*

- The command

  ```
  getent hosts $(hostname -f)
  ```

  have to send the IP-adress of the network gateway the client connected with. If the command send the result *127.0.0.1* or *127.0.0.2* then file `/etc/hosts` has to be corrected.

- Install xinetd:

  ```
  yum install xinetd
  ```

- Install Samba and a database server:

```
yum install mariadb-server samba
```

- Configure samba and database server:

```
systemctl start mariadb.service
mysql_secure_installation
service smb start
service nmb start
service xinetd start
chkconfig smb on
chkconfig nmb on
chkconfig mariadb on
chkconfig xinetd on
```

- If the machine should also act as DHCP-server then the daemon dhcpd has to be configured and be running.

Register at the Red Hat Network:

```
rhn_register
```

Add the opsi RHEL Repository:

RHEL 6

```
yum -y install wget
cd /etc/yum.repos.d
wget http://download.opensuse.org/repositories/home:/uibmz:/opsi:/opsi40/RedHat_RHEL-6/home:uibmz:opsi:opsi40.repo
yum makecache
```

RHEL 7

```
yum -y install wget
cd /etc/yum.repos.d
wget http://download.opensuse.org/repositories/home:/uibmz:/opsi:/opsi40/RHEL_7/home:uibmz:opsi:opsi40.repo
yum makecache
```

After the repository is added you may start the opsi installation:

```
yum install p7zip cabextract
yum remove tftp-server
yum install opsi-depotserver opsi-configed
yum install wimtools
service opsiconfd restart
service opsipxeconfd restart
opsi-setup --auto-configure-samba
chkconfig opsiconfd on
chkconfig opsipxeconfd on
service smb restart
service nmb restart
```

It could happened that the following dialog windows prompt you if you want to import the GPG key of the repository:

```
Importing GPG key 0xD8361F81 "home:uibmz OBS Project <home:uibmz@build.opensuse.org>" from http://download.opensuse.\
  org/repositories/home:/uibmz:/opsi:/opsi40/RedHat_RHEL-6/repodata/repomd.xml.key
Is this ok [y/N]: y
```

Please answer with *y*.

Please make sure that your iptables and SELinux configuration allows the access to the tftp Port (69/UDP) and the opsi ports (4447/TCP and 4441/TCP).

If you want opsi to manage the configuration of the DHCP server, then you need to correct this entry, and point it to the server's public IP address.

Assuming all of the above steps were completed successfully we can assume that the network is properly configured. Next, please continue on with Section 4.2.3

Warning
The unix commands used in the following chapters are working on Debian systems. You may have to change them to match your linux system.

4.1.7 Installation on CentOS Server

Important
Check here if your Server Operating System Version is supported by opsi: Chapter 3
Please note the Configuration Requirements: Section 2.3.

Necessary preparations:

- The command

```
hostname -f
```

  send back a fully qualified domain name containing two dots, e.g. *opsidemo.domain.local*

- The command

```
getent hosts $(hostname -f)
```

  send the IP-adress of the network gateway that the client is connected with. If the command send the result *127.0.0.1* or *127.0.0.2* then file `/etc/hosts` has to be corrected.

- Install xinetd:

```
yum install xinted
```

- Install samba and database server:

```
yum install mysql-server samba
```

  Under RHEL 7 mysql-server was replaced by mariadb-server. You can use the following command for the installation:

```
yum install mariadb-server samba samba-client
```

- Configure samba and database server:

```
systemctl start mariadb.service
mysql_secure_installation
service smb start
service nmb start
service xinetd start
chkconfig smb on
chkconfig nmb on
chkconfig mariadb on
chkconfig xinetd on
```

- If the machine should also act as DHCP-server then the daemon dhcpd has to be configured and up and running.

Add the opsi CentOS Repository:

CentOS 6

```
yum -y install wget
cd /etc/yum.repos.d
wget http://download.opensuse.org/repositories/home:/uibmz:/opsi:/opsi40/CentOS_CentOS-6/home:uibmz:opsi:opsi40.repo
yum makecache
```

CentOS 7

```
yum -y install wget
cd /etc/yum.repos.d
wget http://download.opensuse.org/repositories/home:/uibmz:/opsi:/opsi40/CentOS_7/home:uibmz:opsi:opsi40.repo
yum makecache
```

After adding the repository you may start the opsi installation:

```
yum makecache
yum install p7zip p7zip-plugins cabextract
yum install opsi-depotserver opsi-configed
yum install wimtools
service opsiconfd restart
service opsipxeconfd restart
opsi-setup --auto-configure-samba
chkconfig opsiconfd on
chkconfig opsipxeconfd on
service smb restart
service nmb restart
```

It could happened that the following dialog window prompt you if you want to import the GPG key of the repository:

```
Importing GPG key 0xD8361F81 "home:uibmz OBS Project <home:uibmz@build.opensuse.org>" from http://download.opensuse.\
  org/repositories/home:/uibmz:/opsi:/opsi40/CentOS_CentOS-5/repodata/repomd.xml.key
Is this ok [y/N]: y
```

Please answer with *y*.

Please make sure that your iptables and SELinux configuration allow access to the tftp Port (69/UDP) and the opsi ports (4447/TCP and 4441/TCP).

Assuming all of the above steps were completed successfully we can assume that the network is properly configured. Next continue on with Section 4.2.3

Warning
The unix commands used in the following chapters are working on Debian systems. You may have to change them to match your linux system.

4.2 Update and Configuration of the opsi-server

4.2.1 Proxy Entry in apt-configuration File

If necessary please adapt the file `/etc/apt/apt.conf` to your network configuration (enter the correct proxy or comment/delete unnecessary lines). You can edit your file with a program like midnight commander:

```
mcedit /etc/apt/apt.conf
```

4.2.2 Update of the opsi-server

Update the opsi-server with the commands:

```
aptitude update
aptitude safe-upgrade
```

Tip
During the installation, you may be asked to modify the `smb.conf` file. Please select Yes. If you have modified the `smb.conf` file before you should save the default and make a diff on both files later. If you answered the question with default before reading this tip with *no* you can reconfigure samba from an opsi-server console with following command: `opsi-setup --auto-configure-samba`

4.2.3 Backend Configuration

opsi supports different backends for data management.

The most important backends are:

- file (storage in files)
- mysql (storage in a MySQL database)

Note
Some distributions use *MariaDB* instead of *MySQL*.
The *mysql* backend will also work with MariaDB.

Caution
Since MySQL server version 5.7 the *strict mode* is enabled by default. This mode prevents the command *opsi-setup --configure-mysql* to finish properly, with the correct configurations. To disable the strict mode please edit the file `/etc/mysql/mysql.conf.d/mysqld.cnf`.
In the `[mysqld]` section add the following line underneath the section name:
`sql_mode=NO_ENGINE_SUBSTITUTION`

Now the service `mysql` has to be restarted: `service mysql restart`

Note
The use of the mysql backend for inventory data is free and does not required activation.
More information about the activation of co-financed modules can be found here.

Besides these main backends there are some special backends:

- **opsipxeconfd** (the service for the opsi pxe boot)
- **dhcpd** (used for configuring and restarting the local dhcp service at opsi-server)
- **jsonrpc** (redirects all calls to a other server via JSON-RPC)

We recommend initializing the mysql backend (in order to use it for e.g. inventory data). Therefore enter the command:

```
opsi-setup --configure-mysql
```

It is assumed that MySQL is installed and configured. We then require the credentials for an database administrator. For specific information on installation and configuration of your database please refer to the manuals of your distribution.

The command will ask for information to database access and then use the provided credentials to create a database and an user with appropriate rights to access that database for opsi.

The following screen shots show example parameters for a MySQL configuration setup:

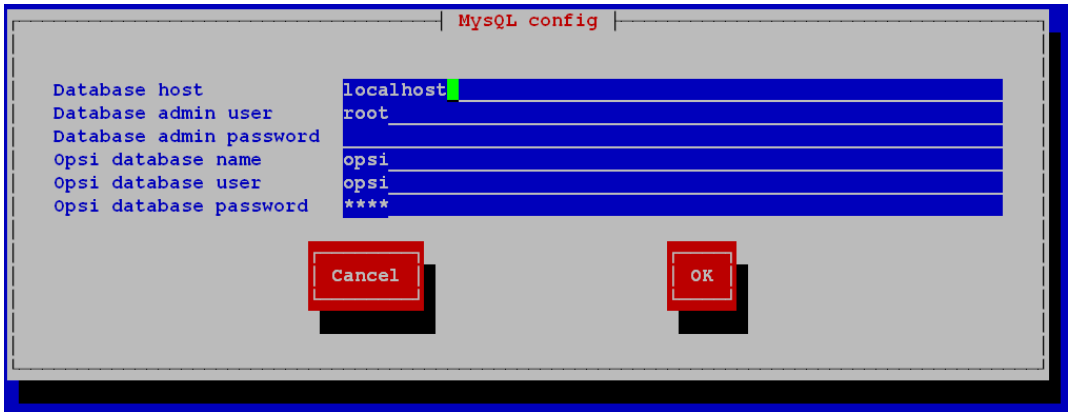

Figure 4.5: Dialog opsi-setup --configure-mysql: Input mask

```
Connecting to host 'localhost' as user 'root'
Successfully connected to host 'localhost' as user 'root'
Creating database 'opsi'
Database 'opsi' created
Creating user 'opsi' and granting all rights on 'opsi'
User 'opsi' created and privileges set
Testing connection to database 'opsi' as user 'opsi'
Successfully connected to host 'localhost' as user 'opsi'
Updating backend config '/etc/opsi/backends/mysql.conf'
Backend config '/etc/opsi/backends/mysql.conf' updated
Initializing mysql backend
```

Figure 4.6: Output: opsi-setup --configure-mysql: Output

You may accept the defaults for all questions except the *Database Admin Password*. The *Database Admin Password* is *linux123* on the pre-installed opsi-VM, otherwise it is the password you entered during the mysql-server installation.

Different kinds of data may be stored in different types of backends. For some actions (such as method calls) more than one backend has to be involved. Therefore, the different method calls can be used by more than one backend. These method-to-backend(s) calls are configured in the file `/etc/opsi/backendManager/dispatch.conf`.

Here an example:

```
# = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = =
# =      backend dispatch configuration                                      =
# = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = =
#
# This file configures which methods are dispatched to which backends.
# Entries has to follow the form:
```

```
# <regular expression to match method name(s)> : <comma separated list of backend name(s)>
#
# Backend names have to match a backend configuration
# file basename <backend name>.conf beneath /etc/opsi/backends.
# For every method executed on backend dispatcher
# the first matching regular expression will be decisive.

# Recommended standard configuration (dhcpd not at the opsi server)
#    file as main backend, mysql as hw/sw invent
#     and license management backend and opsipxeconfd backend:
backend_.*         : file, mysql, opsipxeconfd
host_.*            : file, opsipxeconfd
productOnClient_.* : file, opsipxeconfd
configState_.*     : file, opsipxeconfd
license.*          : mysql
softwareLicense.*  : mysql
audit.*            : mysql
.*                 : file
```

You will find explanations and examples at the top of this file. At the first column is the name of the opsi method being called (with wildcard *) and after the colon is the list of backends used by that opsi method. For every called method procedure the first column of this list is proved to determine which backend have to be used. The last line (.*) matches all opsi method calls.

The default backend during the installation of package is the file backend as main backend.

Caution
Make sure that every backend used is listed in the line starting with *backend_.**.

Now at the initial start-up (even without changing the file) you should call:

```
opsi-setup --init-current-config
opsi-setup --set-rights
service opsiconfd restart
service opsipxeconfd restart
```

The access rights for calling the opsi methods are configured in `/etc/opsi/backendManager/acl.conf`.

4.2.4 Set Samba Configuration and Change Passwords

Opsi requires certain samba shares. To ensure that they are configured please enter the following command:

```
opsi-setup --auto-configure-samba
```

A *pcpatch* user is created on the system. This user can install software on a client PC. The *pcpatch* user allows access to the configuration data on the host shares. The user *pcpatch* needs to get a correct password - once as system user, as samba user and as opsi user.

In a terminal window the program *opsi-admin* should be called which will set the pcpatch-password for opsi, unix and samba.

```
opsi-admin -d task setPcpatchPassword
```

After sending the command enter the password.

4.2.5 Checking the Java Configuration

The opsi-servers and the connected clients are administrated with the program opsi-configed. It communicates with the opsi server via HTTPS therefore it can be used on every computer which can build a HTTPS connection to the opsi server. The computer where the program is installed currently requires a Java Runtime Enviromnent (JRE), at least JRE version 8 (internal version 1.8).

If you would like to run the opsi management interface *opsi-configed* directly on the server you need to install a graphical environment and the Java Runtime Environment on it. If this is not needed or possible on the server this chapter can be skipped.

A graphical environment and Java Runtime Environment are already installed in the virtual machine provided by uib.

To check the Java version enter the following command in a terminal window:

```
java -version
```

If the shown version is too old install a newer one.

Tip
Mind the resolution of the display of your virtual machine. Usage of opsi-configed is clumsy with a resolution of less than 1024x768. To improve graphics and mouse drivers in a virtual machine it is helpful to install *VMware tools* for a VMware machine or the *virtual-guest-additions* for a VirtualBox machine.

4.2.6 Create Users and administrate the groups opsiadmin / pcpatch

In the following example, we create the user *adminuser*, which is a similar procedure to creating an account for yourself.

Let's create the user:

```
useradd -m -s /bin/bash adminuser
```

now set the unix password:

```
passwd adminuser
```

and now the samba password:

```
smbpasswd -a adminuser
```

Caution
Do not use the char *§* as part of the passwords. It becomes impossible to login at the opsi web service.

Create and test the group membership:

```
usermod -aG opsiadmin adminuser
getent group opsiadmin
```

the getent command should have a result like:
opsiadmin:x:1001:opsiconfd,adminuser

All users who build opsi packages (`opsi-makeproductfile`), install opsi packages (`opsi-package-manager`), or manually edit the configuration files have to also be in the group *pcpatch* :

```
usermod -aG pcpatch adminuser
```

Test the results by entering:

```
getent group pcpatch
```

The result should look like
pcpatch:x:992:adminuser

To make `sudo opsi-set-rights` available, please execute:

```
opsi-setup --patch-sudoers-file
```

`opsi-set-rights`
does the same as `opsi-setup --set-rights` , but can be executed not only as root, but also with sudo from members of the group pcpatch (or opsi-file-admins):
Example:

```
sudo opsi-set-rights .
```

root is allowed to do anything, and does not have to be explicitly registered in the group.

4.3 DHCP Configuration

Important:

It is essential for opsi that the DHCP has the correct addresses. To simplify the setup, the opsi-server VM is delivered with a running DHCP server. In the situation where a DHCP server already exists, it should be configured to work with opsi. Both alternatives are described below.

4.3.1 Using a DHCP Server at the opsi-server

Using the opsi-Server VM: The opsi server VM has a installed DHCP server.
The DHCP server on the opsi-server VM is configured with no free leases, so no unknown clients will get an IP-Number from this DHCP server.
If you create a client at the opsi-server using the opsi-configed, it will also create a dhcp entry for this client. Therefore you have to supply the IP-number and the MAC-address.

Your own installation: If you want to use the opsi server as DHCP server, you have to install the DHCP server package.

e.g.

```
aptitude install isc-dhcp-server
```

After the installation you must configure the dhcp configuration for opsi. This is done by the following command:

```
opsi-setup --auto-configure-dhcpd
```

To execute the command to restart the DHCP server from opsiconfd listed in the configuration file */etc/opsi/backends/dhcpd.conf*, an entry under /etc/sudoers is required. This is created using the command

```
opsi-setup --patch-sudoers-file
```

If deemed necessary, the permissions of the DHCPD configuration file can be checked, which should look like this, for example:

```
-rw-r--r-- 1 opsiconfd opsiadmin 80174 Dec 22 14:37 /etc/dhcp/dhcpd.conf
```

4.3.2 Using an External DHCP Server

Using the opsi-Server VM: If you use an external DHCP server, then you may want to uninstall the DHCP server at the opsi-server, which is done by entering:

```
aptitude remove dhcp3-server
```

or

```
aptitude remove isc-dhcp-server
```

Your own installation: Since opsi 4.0.3 no dhcp server is automatically installed by dependency to the opsi server packages

Next you have to configure your external DHCP server, and provide the client information to the external DHCP server, such that clients know that our opsi-server is now the boot server. If your external DHCP runs on Linux, then you need the following entries for the clients in the `/etc/dhcp3/dhcpd.conf` file.

next-server <ip of opsi-server>;
filename "linux/pxelinux.0";

Replace *<ip of opsi-server>* by the IP-number of the opsi-server.

If the opsi server runs on openSUSE or SLES, then `filename=opsi/pxelinux.0`.
If the opsi server runs on UCS, then `filename=pxelinux.0`.

If you are using a Windows server, then the corresponding entries may be *bootserver* or *startserver* and *bootfile* or *startfile* (*Options 66 / 67*).

If you create a client at the opsi-server, then you only have to supply the MAC-address, but not the IP-number.

4.3.3 Checking the Backend Configuration for DHCP Entries

Regardless of whether or not you use the dhcp of the opsi-server, you have to configure the opsi-server.

The file `/etc/opsi/backendManager/dispatch.conf` defines which opsi method uses which backends (file, ldap, mysql).

The lines with *backend_. and* host_. entries configure how changes at host entries are handled. If you are using the DHCP server on the opsi-server, then the backend dhcpd has to be added here.
The accordant entry using the file backend is (e.g.):

```
backend_.*          : file, opsipxeconfd, dhcpd
host_.*             : file, opsipxeconfd, dhcpd
```

If the local DHCP isn't used, then the backend dhcpd is not required:

```
backend_.*          : file, opsipxeconfd
host_.*             : file, opsipxeconfd
```

After adapting the correct backend configuration, you should execute:

```
opsi-setup --init-current-config
opsi-setup --set-rights
service opsiconfd restart
service opsipxeconfd restart
```

4.4 Configure how the opsi-server gets the Client's IP-Address

In the default method of the opsi software deployment, only the client must know how to contact the opsi-server.

However, if you would like to use one of the opsi *push* features (like send messages to the client, or fire *on_demand* events, or get session information, or start remote control software), then the server needs to know how to get the IP-Address of the client.

How the opsi server does this, depends on your DNS/DHCP configuration and policy. There are a large number of possible configurations. So here we show two different example configurations:

1. The clients are not known by the DNS (only by netbios), and they get dynamically assigned frequently changing IP-Numbers by the DHCP.
2. The DNS always provides the correct IP-Address of a client.

To configure the opsi server to your situation, you may change the following parameters:

- The entry `resolveHostAddress` in the file `/etc/opsi/backends/hostcontrol.conf`
  This option controls whether the name resolution of a opsi-client address is primarily done by the opsi database or by the name resolution of the operating system of the *opsi-server*.
  If this option is `True`, the *opsi-server* first tries to get the IP-Address of an *opsi-client* using the name resolution from the operating system (DNS, /etc/hosts). If the operating system DNS name resolution fails, then the opsi database is used.
  To use the opsi database during the first attempt, you have to set this option to `False`.
- The entry `update ip` at the file `/etc/opsi/opsiconfd.conf`
  If this entry is *yes*, then the opsi-server will update it's own IP database whenever the opsi-server gets a client IP-Address (e.g. at every web service contact of a client). The default is *yes*.

If you are running configuration example 1, then you should probably set `resolveHostAddress` to *False* and `update ip` to *yes*.

If you are running configuration example 2, then the best configuration is to set `resolveHostAddress` to *True* and `update ip` to *no*.

You should decide for yourself which combination fits the needs of your situation.

If you changed anything while configuring your environment, then you should reload the opsiconfd:

```
service opsiconfd restart
```

4.5 Install the Minimal opsi Products

One important and new feature of opsi 4.0 is a simple tool that updates opsi products from a configured repository with the command `opsi-product-updater`. This tool compares the version of the locally installed products with the versions available at the repository, and then performs an upgrade if necessary. It is also possible to install additional products using `opsi-product-updater`. The `opsi-product-updater` configuration is located in the file `/etc/opsi/{opsi-product}-updater.conf`. The default repository is http://download.uib.de/opsi4.0/products/, and there the directories netboot/ and localboot/, for linux-clients also in opsi-linux/.

The `opsi-product-updater` has the following core features:

- **autoInstall**:
  Install all available products from the repository.
- **autoUpdate**:
  Update products from the repository if the server has an older version.

- **autoSetup**: After installation of an updated product, switch the requested action to *setup* for all clients which have this product installed.

For more details regarding this function, refer to the opsi-manual.

You should now download and install the opsi products with the command:

```
opsi-product-updater -i -vv
```

If the `opsi-product-updater` fails, it may be necessary to add a required proxy to the configuration file:

```
[repository_uib]
proxy =
```

Please notice that the OS-Installation products like win10 aren't ready for action after installation. The installation has to be supplemented by the installation files from the corresponding installation media (i.e. CD, see Section 6.1.3).

4.6 Installing and Checking the Activation File

Event though opsi is open source, there are some components which are not free at the moment. These components are developed in a project that is co-funded by various partners. Which means that until the complete development costs are recuperated by the co-funders, the modules are only allowed to be used by the co-funders or for evaluation purposes. Once the costs of development have been financed, then we will release these modules to everybody for free. To control the use of these components until they are free, there is a activation file `/etc/opsi/modules`, which is protected against changes via electronic signature. If this activation file doesn't exist, then only the modules which are part of the free core of opsi will be available.

If you would like to evaluate opsi's modules, then a temporary activation file can be obtained by contacting *info@uib.de*.

If you become a co-funder, you will get an unlimited activation file. Copy the file to `/etc/opsi` as root.

Then do the command:

```
opsi-setup --set-rights /etc/opsi
```

You may check your activation state with one of the following methods:

Using *opsi-configed*, choose the menu entry Help/opsi-Module, which shows a window with the activation state.

Or at the command line, you can enter `opsi-admin` with the method *backend_info*. (Remark: Never give your activation file or the output of this command to a third party without deleting the signature).

```
opsi-admin -d method backend_info
```

Example:

```
{
"opsiVersion" : "4.0.1",
"modules" :
          {
          "customer" : "uib GmbH",
          "dynamic_depot" : true,
          "vista" : true,
          "treeview" : true,
          "license_management" : true,
          "swondemand" : true,
          "expires" : "2011-04-30",
          "valid" : true,
          "multiplex" : true,
          "signature" : "THIS-IS-NOT-A-VALID-SIGNATURE",
          "vpn" : true,
          "mysql_backend" : true,
          "high_availability" : true
          }
}
```

Note that you only need the modules-file for additional usage, and not for the general use of opsi.

4.7 Starting the management interface (opsi-configed)

Opsi offers a user-friendly configuration editor interface with the opsi-configed application.

There are different ways to start opsi-configed:

- Enter the following address in a browser: `https://<opsi-server>:4447/configed.jnlp`
  The configuration interface will then load through Java Web Start.
  A java version of at least 1.7 is required on the computer running the browser.
- Another option would be to right-mouse-click on the desktop of your opsi-server to open the context menu and then choose *opsi config editor*.
  In this case a java runtime environment must be installed on the server.

The opsi-configed interface is mostly self-explanatory. However, here are some hints: Any changes have to be saved in order to show any effect. Saving changes can be done with the check-mark button. To see changes that may have taken place, you have to reload the data, which can be done with the button at the top left hand corner that has the reload arrows. Reloading can also be done with a right mouse click, and selecting reload.

A more detailed description can be found in the opsi manual.

4.8 Network Ports used by a opsi installation

This is an overview of ports and network protocols that will be used.

- opsi-server Webservice: TCP 4447
  Client to server, depot to server (bidirectional, self connection).
- opsi-client Webservice: TCP 4441
  Server to client, client self connection via localhost.
- opsi-client Webservice: TCP 4442
  Client self connection via localhost.
- opsi-client Notifier: TCP 45000 - 65536
  Client self connection via localhost. A random port in the given range is selected.
- TFTP: UDP 69
  Client to server
- CIFS/SMB: UDP 137 / UDP 138 (netbios) / TCP 139 / TCP 445
  Client to server (bidirectional).
  According to the version of client OS.
- WEBDAV: TCP 80
- WINEXE: TCP 139
- SSH (optional): TCP 22
- DNS: TCP 53
- WakeOnLan (WOL): UDP 12287
  Server to client.
- HTTP: TCP 80
  E.g. server updates from http://download.opensuse.org/
- HTTPS: TCP 443
  opsi-packages from https://download.uib.de (opsi-package-updater)

Chapter 5

Integration of Existing Clients

To integrate existing Windows clients into opsi, the opsi-client-agent has to be installed on these systems. There are different ways to do this, which are described below. After you have done so, you should see the client in the opsi-configed after selecting the tab *Clients*.

5.1 Installation of the opsi-client-agent

5.1.1 Usage of service_setup.cmd

This method is the first choice for installations on a single computer. service_setup.cmd can also be used for repair purposes. For mass roll-out, see the chapter below.

5.1.1.1 service_setup.cmd on Windows NT6

1. logon to the Windows client with administrative privileges
2. mount the shared directory on the opsi server at `\\<opsiserver>\opsi_depot` to a drive letter
3. on the drive letter from the previous step, start the script `opsi-client-agent\service_setup.cmd`
   Do not start the script elevated (via right mouse click: *as Administrator*) because an elevated script may have no access to the network share.
4. The script copies the needed files to a temporary local directory and starts from there the opsi-script (winst32.exe) elevated in order to do the installation. Therefore you may see here an UAC Message.
5. The script connects to the opsi-webservice in order to create the client on the server side and get the pckey. The first connection is established with the user/password combination provided in the config.ini. If the connection fails, a login window will pop up, where the user can type in a Service-URL (opsi-config-server), with a user and a password. The provided user needs to be member of the group *opsiadmin*.

Caution
Der Client rebootet nach der Installation.

5.1.1.2 service_setup_NT5.cmd on Windows NT5

1. logon to the Windows client with administrative privileges
2. mount the shared directory on the opsi server at `\\<opsiserver>\opsi_depot` to a drive letter
3. on the drive letter from the previous step, start the script `opsi-client-agent\service_setup_NT5.cmd`
4. The script copies the needed files to a temporary local directory and starts from there the opsi-script (winst32.exe) in order to do the installation.
5. The script connects to the opsi-webservice in order to create the client on the server side and get the pckey. The first connection is established with the user/password combination provided in the config.ini. If the connection fails, a login window will pop up, where the user can type in a Service-URL (opsi-config-server), with a user and a password. The provided user needs to be member of the group *opsiadmin*.

Warning
During installation, the client reboots without notice!

5.1.2 Usage of the opsi-deploy-client-agent

The `opsi-deploy-client-agent` script installs the opsi-client-agent directly from the opsi-server to the clients. Requirements for the clients are:

- an open C$ share
- an open admin$ share
- an administrative account

On the server side we require the program `winexe`. A statically linked binary of version 0.90 is part of opsi-client-agent. For setting up clients with a Windows version newer than Windows 7 *winexe* is required to be **newer** than version 1.0.

Winexe may be provided through the repositories of the used distribution. For some distributions you may find a newer version at http://download.opensuse.org/repositories/home:/uibmz:/opsi:/opsi40-testing/. If you do not find any packages suiting your distribution or want to create a current version of the program yourself you will find further information on the project page at http://sourceforge.net/projects/winexe/.

The script creates the client on the server, then copies the installation files and the configuration information including the pckey to the client. After copying the necessary information, `opsi-deploy-client-agent` starts the installation on the client.

Two copy methods are possible. The first method will use the *mount*-command to locally mount the C$ share of the client on the server. The second variant will use *smbclient* for mounting. After mounting the client the files will be copied.

The script can work with IP addresses, hostnames or FQDNs. It will automatically detect what type of address it is processing.

With the `opsi-deploy-client-agent` script you can batch install a list of clients. Therefore you can give the FQDN's of multiple clients as last argument or you give with the opsion `-f` the name of a text file which has one FQDN per line.

The script itself is located in `/var/lib/opsi/depot/opsi-client-agent`.

Run this script with *root* privileges.

It may be possible that you have to make the script executable with:
`opsi-set-rights /var/lib/opsi/depot/opsi-client-agent/opsi-deploy-client-agent`

```
bonifax:/home/uib/oertel# cd /var/lib/opsi/depot/opsi-client-agent
bonifax:/var/lib/opsi/depot/opsi-linux-client-agent# ./opsi-deploy-client-agent --help
usage: opsi-deploy-client-agent [-h] [--version] [--verbose]
                                [--debug-file DEBUGFILE] [--username USERNAME]
                                [--password PASSWORD]
                                [--use-fqdn | --use-hostname | --use-ip-address]
                                [--ignore-failed-ping]
                                [--reboot | --shutdown | --start-opsiclientd]
                                [--hosts-from-file HOSTFILE]
                                [--skip-existing-clients]
                                [--threads MAXTHREADS] [--smbclient | --mount]
                                [--keep-client-on-failure | --remove-client-on-failure]
                                [host [host ...]]

Deploy opsi client agent to the specified clients. The c$ and admin$ must be
accessible on every client. Simple File Sharing (Folder Options) should be
disabled on the Windows machine.

positional arguments:
  host                  The hosts to deploy the opsi-client-agent to.

optional arguments:
  -h, --help            show this help message and exit
  --version, -V         show program's version number and exit
  --verbose, -v         increase verbosity (can be used multiple times)
  --debug-file DEBUGFILE
                        Write debug output to given file.
  --username USERNAME, -u USERNAME
                        username for authentication (default: Administrator).
                        Example for a domain account: -u
                        "<DOMAIN>\\<username>"
  --password PASSWORD, -p PASSWORD
                        password for authentication
  --use-fqdn, -c        Use FQDN to connect to client.
  --use-hostname        Use hostname to connect to client.
  --use-ip-address      Use IP address to connect to client.
  --ignore-failed-ping, -x
                        try installation even if ping fails
  --reboot, -r          reboot computer after installation
  --shutdown, -s        shutdown computer after installation
  --start-opsiclientd, -o
                        start opsiclientd service after installation
  --hosts-from-file HOSTFILE, -f HOSTFILE
                        File containing addresses of hosts (one per line).If
                        there is a space followed by text after the address
                        this will be used as client description for new
                        clients.
  --skip-existing-clients, -S
                        skip known opsi clients
  --threads MAXTHREADS, -t MAXTHREADS
                        number of concurrent deployment threads
  --smbclient           Mount the client's C$-share via smbclient.
  --mount               Mount the client's C$-share via normal mount on the
                        server for copying the files. This imitates the
                        behaviour of the 'old' script.
  --keep-client-on-failure
                        If the client was created in opsi through this script
                        it will not be removed in case of failure. (DEFAULT)
  --remove-client-on-failure
                        If the client was created in opsi through this script
                        it will be removed in case of failure.
```

5.2 Rollout from Existing Products

5.2.1 Usage of opsi standard products: opsi-configed

One of the opsi standard products is the product `opsi-configed`. This product install the opsi Management Interface. This Application is a Java Program so the Java Runtime Engine is also needed and will be installed automatically.

Using *opsi-configed*, choose the client by pressing the tab *Clients*, which puts *opsi-configed* in the mode *Configuration of clients*.

If you haven't done so yet, reload all the data by clicking the reload button at the top left corner of the *opsi-configed* interface (or use the *File* menu).

Switch to the tab *Product configuration*, look for the line with the product-id `opsi-configed`. Go to the column *Requested Action*, and select the action *setup* using a left mouse click. Finally, save the new action with a click on the checkmark button at the top (or by right clicking the mouse and selecting *save*).

Now reboot the client, and the products `javavm` and `opsi-configed` should automatically be installed. After the installation you should find the entry `opsi-configed` in the `Start Menue`

5.2.2 Hard- and Software Inventory with the Products hwaudit and swaudit

Using *opsi-configed*, choose the client by pressing the tab *Clients*, which puts *opsi-configed* in the mode *Configuration of clients*.

If you haven't done so yet, reload all the data by clicking the reload button at the top left corner of the *opsi-configed* interface (or use the *File* menu).

Switch to the tab *Product configuration*, look for the lines that audit the software and hardware of the system (*hwaudit* and/or *swaudit*). Go to the column *Requested Action*, and select the action *setup* using a left mouse click. Finally, save the new action with a click on the checkmark button at the top (or by right clicking the mouse and selecting *save*).

Now reboot the client, and the *hwaudit* and/or *swaudit* should automatically start. The client scans the hardware and/or software inventory and sends the results back to the server.

To see the changes at the *opsi-configed* management interface, select reload with the button at the top or with a right mouse click. You may see the update after selecting the tabs *Hardware information* and *Software inventory*.

5.2.3 Hardware Inventory with the Netboot Product hwinvent

Using *opsi-configed*, choose the client by pressing the tab *Clients*, which puts *opsi-configed* in the mode *Configuration of clients*.

If you haven't done so yet, reload all the data by clicking the reload button at the top left corner of the *opsi-configed* interface (or use the *File* menu).

Switch to the tab *Netboot products*, look for the line that has *hwinvent*. Go to the column *Requested Action*, and select the action *setup*. Finally, save the new action with a click on the checkmark button at the top (or by right clicking the mouse and selecting *save*).

Now reboot the client (over PXE), and the bootimage with *hwinvent* should start automatically. At first, the client reboots using the Linux boot image, and then it scans the hardware and sends the results back to the server.

To see the changes at the *opsi-configed* management interface, select reload with the button at the top or with the mouse. You may see the update after selecting the tab *Hardware information*.

Chapter 6

Installation of a New Windows OS using opsi

6.1 Creating a New Client using the opsi Management Interface

You need a client (with a minimum of 1024 MB RAM) that is able to boot per PXE over the network. For an initial test, we suggest you download a corresponding vmware-image at download.uib.de (http://download.uib.de/-vmware_pxeclient.zip). The advantage of vmware (virtual hardware) is that is supports the standard drivers from windows.

Now you have to create a client in the opsi system. Start the installation with either a) the opsi-configed, or b) the command line.

Graphic frontend of opsi-configed: Using *opsi-configed*, choose the client by pressing the tab *Clients*, which puts *opsi-configed* in the mode *Configuration of clients*.

From the menu, choose *OpsiClient/Create new opsi client* and enter the following for the client:

- IP-name
- DNS domain (if different from the default)
- client description
- IP-number (required if you can not use DNS to resolve the address of the client)
- MAC-address (required if the opsi-server is also your DHCP server or if you want to use PXE boot with this client)

The client will be created in the opsi database. If the client is configured as a PXE-client, then it will also be configured in the DHCP on the opsi-server.

Command line opsi-admin An opsi client may also be created at the command line:

```
opsi-admin -d method host_createOpsiClient <client-id> [opsiHostKey] [description] [notes] [hardwareAddress] [ipAddress\
    ] [inventoryNumber] [oneTimePassword] [created] [lastSeen]
```

e.g.:

```
opsi-admin -d method host_createOpsiClient testclient.domain.local "null" "Testclient" "" 00:0c:29:12:34:56 192.168.0.5
```

To see all created clients, in *opsi-configed* choose the client by pressing the tab *Clients*, which puts *opsi-configed* in the mode *Configuration of clients*, and reload the data by pressing F5 or use the context menu.

6.1.1 Hardware Inventory with the Netboot Product hwinvent

Using *opsi-configed*, choose the client by pressing the tab *Clients*, which puts *opsi-configed* in the mode *Configuration of clients*.

If you haven't done so yet, reload all the data by clicking the reload button at the top left corner of the *opsi-configed* interface (or use the *File* menu).

Switch to the tab *Netboot products*, look for the line that has *hwinvent*. Go to the column *Requested Action*, and select the action *setup*. Finally, save the new action with a click on the checkmark button at the top (or by right clicking the mouse and selecting *save*).

Now reboot the client (over PXE), and the bootimage with *hwinvent* should start automatically. At first, the client reboots using the Linux boot image, and then it scans the hardware and sends the results back to the server.

To see the changes at the *opsi-configed* management interface, select reload with the button at the top or with the mouse. You may see the update after selecting the tab *Hardware information*.

6.1.2 Create a New Client using the opsi-client-bootcd

At the opsi download site you will find ISO images of the opsi-client-bootcd in http://download.uib.de/opsi4.0/. Just download the newest image and burn it to a cd-rom or dvd-rom. Boot your computer from this one. You should see the following image:

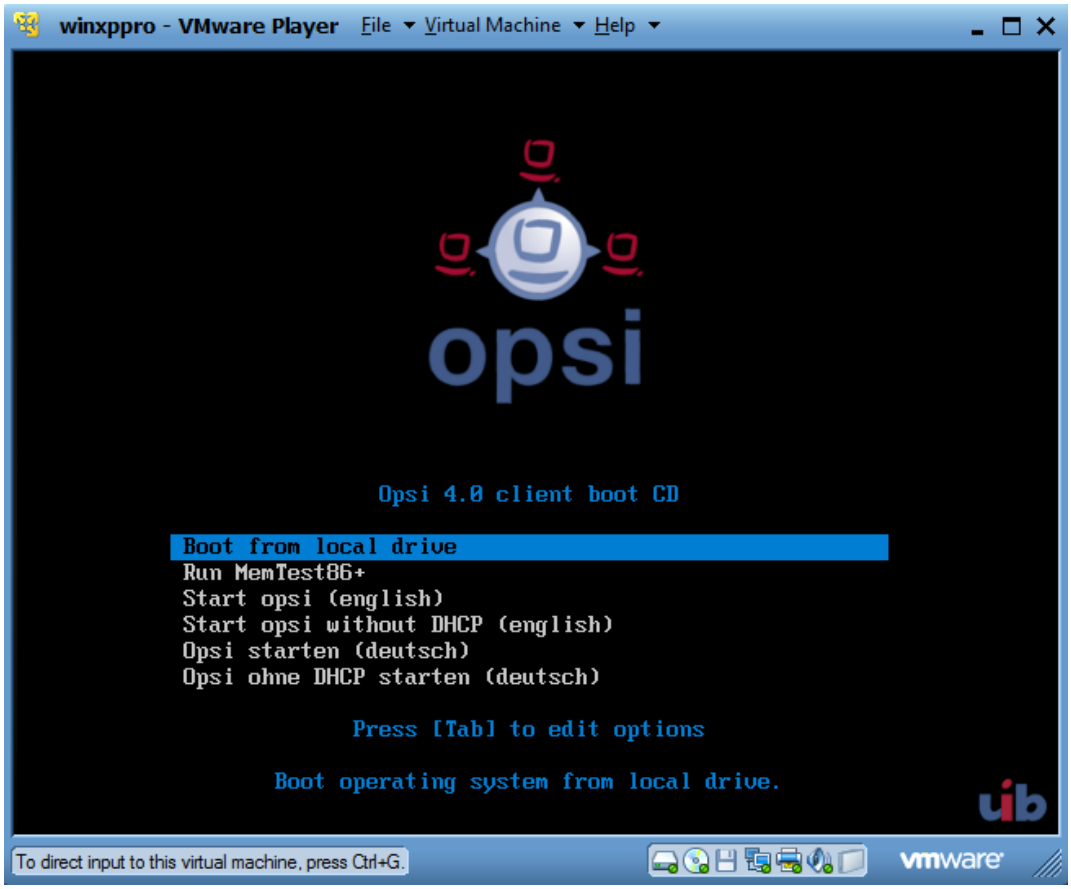

Figure 6.1: Start image opsi-client-boot-cd

Choose *Start opsi (English)*. After a while, the following screen will appear. If your DHCP server gives IP-numbers to unknown DHCP clients, then most fields will already have valid values. You have to complete the missing data. You must at least give the hostname.

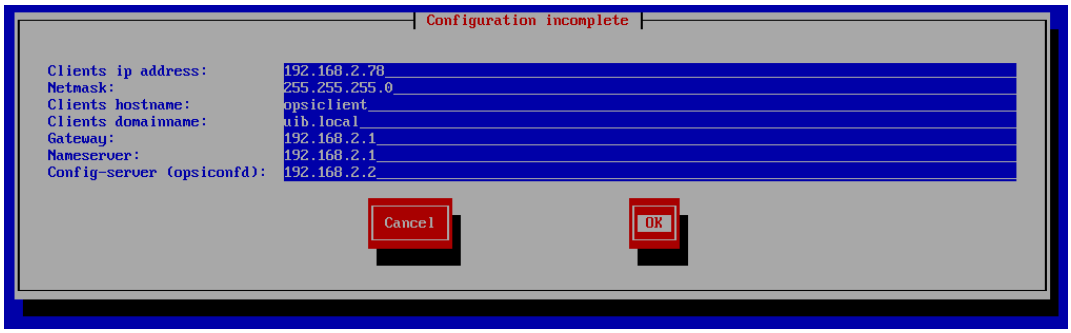

Figure 6.2: bootimage/boot-cd configuration screen

Confirm with *OK*.

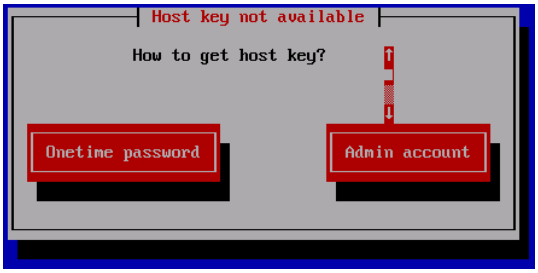

Figure 6.3: bootimage / boot-cd: Choose how to create Client

Choose *Admin account*. This means that the client should register himself at the opsi-server. This procedure must be authorized.

Figure 6.4: bootimage / boot-cd: netboot product list

Now you may choose the operating system that you would like to install (or e.g. hwinvent for testing).

6.1.3 OS-Installation: Complete the Base Package for Windows

The opsi OS-packages contain only the files that are necessary to perform our automated OS installation, but not the operating system software itself (Windows Operating Systems).

If you want to make use of fully automated OS installations of Windows 7/8.1/10}, you have to complete these packages as described below.

6.1.4 NT 6 family: Win10 / Win8.1 / Win7 / 2008R2

In order to perform an OS Installation, a so-called WinPE is being used as a *Live OS*. You can create it using an opsi package (ADK 8.1,10), or do it all yourself manually following the steps described. Generally speaking, the Windows-Version of the PE does not matter with regard to the Windows OS version being installed. Still, working drivers should be present for at least disk and network devices. Microsoft recommends 32-Bit PE for x86 installations , and 64-Bit PE for x64 installations.

"To install a 64-bit version of Windows you must use a 64-bit version of Windows PE. Likewise, to install a 32-bit version of Windows, you must use a 32-bit version of Windows PE."
http://technet.microsoft.com/en-us/library/cc766093.aspx

Regardless of how you want to make your PE, you'll need an "Assessment and Deployment Kit" (ADK, Win8.1 bzw 10), or its predecessor "Windows Automated Installation Kit" (Windows AIK; bis Windows 7), and installing it on a supported Windows OS (use x64):

ADK Windows 8.1 / 10: https://docs.microsoft.com/en-us/windows-hardware/get-started/adk-install

It's sufficient to install *Windows Preinstallation Environment (Windows PE)* and the dependencies automatically selected. Stick with the suggested install path in `Program Files (x86)`.
WAIK Windows 7: http://www.microsoft.com/downloads/details.aspx?displaylang=en&FamilyID=696dd665-9f76-4177-a811-39c26d3b3b34 This site provides you with an ISO file, which may then be burnt to a CD or mounted. The content of this CD must be installed in an OS mentioned in the previous system requirements.

6.1.4.1 Creating a PE

The easiest available method for tweaking your PE requires a computer that has opsi-client-agent installed, as well as a Windows ADK (Win8.1,Win10). Doing all steps manually is being described below Section 6.1.4.3.

6.1.4.2 Creating a PE using opsi

- Set the localboot- produkt `opsi-winpe` to `once` for the client you intend to use, if desired adjust the produkt properties to `x86` instead of `x64` at the lower right side, and save (right click > save)
- (in case the opsi-product `opsi-winpe` is missing, install it onto your opsi server via CLI `opsi-product-updater -i -p opsi-winpe -vv`)
- launch an installation event for the client (right click > on-demand , or reboot)
- after successful run of this action move or copy the contents of the now existing folder on your client C:\winpe_<ARCH>\media\ into the folder (already existing) within the OS folder you want to use at `\\opsiserver\opsi_depot_rw\<OS>\winpe\`
- finally run the CLI command on your new opsi server. Done.

```
opsi-setup --set-rights
```

6.1.4.3 Manually Creating a PE

Here is how to manually tweak a PE. The console commands are very similar in 32- or 64-bit versions, except for the **<ARCH>** entries. These have to be set to either **x86** , **amd64** or **ia64**.

6.1.4.4 Tweaking a WinPE (WAIK / Windows 7)

Start a terminal as Administrator with elevated rights (Start ⇒ Programs ⇒ Accessories ⇒ right click on "Command Prompt" ⇒ "Run as" ⇒ Administrator)

- Copy the WinPE

```
"%ProgramFiles%\Windows AIK\Tools\PETools\copype.cmd" <ARCH> C:\winpe
```

- Mount Image:

```
"%ProgramFiles%\Windows AIK\Tools\<ARCH>\imagex.exe" /mountrw "C:\winpe\winpe.wim" 1 "C:\winpe\mount"
```

- replace startnet.cmd

```
echo c:\opsi\startnet.cmd > "C:\winpe\mount\Windows\System32\startnet.cmd"
```

(Remark: The file `startnet.cmd` will be created by the opsi linux boot image after the script `setup.py` is executed. The `startnet.cmd` contains the call to wpeinit.)

- Unmount the Image

```
"%ProgramFiles%\Windows AIK\Tools\<ARCH>\imagex.exe" /commit /unmount "C:\winpe\mount"
```

- Move the WinPE now. From the target dir more files will be moved to the server.

```
move "C:\winpe\winpe.wim" "C:\winpe\ISO\sources\boot.wim"
```

- Copy the contents of `C:\winpe\ISO` to `/var/lib/opsi/depot/win7/winpe` (or `/var/lib/opsi/depot/win2008/winpe`).
  Adjust the file access rights by entering e.g.:

```
opsi-setup --set-rights /var/lib/opsi/depot/<productid>/winpe
```

6.1.4.5 Tweaking a WinPE (ADK / Windows 8/10)

run Start > "Windows Kits" > "Windows ADK" > "Deployment and Imaging Toolkits Environment" from the Start Menu. You will see a command prompt, but with some enironment variables set.

- Copy the WinPE

```
copype.cmd <ARCH> C:\winpe
```

- Mount the Image

```
dism /Mount-Wim /WimFile:C:\winpe\media\sources\boot.wim /index:1 /MountDir:c:\winpe\mount
```

- replace startnet.cmd

```
echo c:\opsi\startnet.cmd > "C:\winpe\mount\Windows\System32\startnet.cmd"
```

(Remark: The file `startnet.cmd` will be created by the opsi linux boot image after the script `setup.py` is executed. The `startnet.cmd` contains the call to wpeinit.)

- Unmount the Image

```
dism /Unmount-Wim /MountDir:c:\winpe\mount /Commit
```

- Copy the contents of `C:\winpe\media` to `/var/lib/opsi/depot/<productid>/winpe` .
  Adjust the file access rights by entering:

```
opsi-setup --set-rights /var/lib/opsi/depot/<productid>/winpe
```

6.1.4.6 Extending a PE

In some cases it is useful to extend a PE. Especially when using Dell-Hardware. Dell provides special network and storage drivers for use in PE. These instructions only work with Windows 7. (Windows Vista does not inherit the needed DISM- Deployment Image Servicing and Management.) These instructions assume that you have already completed the chapter "Creating a PE".

Note
The Windows Automated Installation Kit is not needed for following instructions.

The first step is to download Dell-PE-drivers from the Dell-Website. For Windows 7, you will need the WINPE 3.0 Drivers from Dell. The downloaded CAB-File must be extracted to the local disk. This can be done with 7zip or the command-line-tool Expand.exe. For simplicity, we recommend creating a directory called "dell-driver" on the local disk, and then extracting the CAB-File into this directory.

- Use dism to scan the image, in order to determine the required index number. Normally a PE-image is a one-image-file, so you can use the index 1, but it is better to check at first. Start a terminal as administrator (Start ⇒ Programs ⇒ Accessories ⇒ right click on "Command Prompt" ⇒ "Run as" ⇒ (Administrator) and run the following command:

```
dism /Get-WimInfo /WimFile:C:\winpe\ISO\sources\boot.wim
```

In the output of this command, you can see which images are included in the image file.

- The next command mounts the image for modification:

```
dism /Mount-Wim /WimFile:C:\winpe\ISO\sources\boot.wim /index:1 /MountDir:c:\winpe\mount
```

- To integrate the extracted drivers into the mounted image, you need to execute this command:

```
dism /Image:C:\winpe\mount /Add-Driver /Driver:c:\dell-driver\winpe\x64 /Recurse
```

If the architecture is 32Bit, the `x64` must be replaced with `x86`. The Driver-CAB from Dell inherits drivers for both architectures.

Note
If only one driver has to be integrated, then leave out the option `/Recurse`, and point directly to the driver-inf-File instead of the driver-directory. With the option `/ForceUnsigned` it is possible to integrate unsigned drivers to the image.

- For the changes to be committed, the images must be unmounted:

```
dism /Unmount-Wim /MountDir:c:\winpe\mount /Commit
```

- Copy the directory `C:\winpe\ISO` with the target name `winpe` to `/var/lib/opsi/depot/win7/` (or `/var/lib/opsi/depot/win2008`).
  Adjust the file access rights by entering(e.g.):

```
opsi-setup --set-rights /var/lib/opsi/depot/win7/winpe
```

6.1.4.7 unattend.xml

The control file for the unattended installation is the XML file `unattend.xml`, which you can find under `/var/lib/opsi/depot/win7/custom`. If you would like to make any modifications to this file, then do it in this directory and not in the opsi directory.

The file `unattend.xml` that comes with the opsi package, contains the activating of the Administrator account with the password *nt123*.

Documentation of the file `unattend.xml` can be found (after the installing WAIK) in the directory `c:\Program Files\Windows\Waik\docs\chms`.

6.1.4.8 Driver Integration

Place your driver directories in `/var/lib/opsi/depot/<product-id>/drivers/drivers`. Then call the script `create_driver_links.py` in the folder `/var/lib/opsi/depot/<product-id>/`.

Please keep in mind that only signed drivers are accepted. Therefore, if you want to use driver packs like the driver packs from driverpacks.net, be sure to use only the Windows 7/8.1/10 versions.

6.1.4.9 Providing the Installation Files

Copy the complete installation DVD to
`/var/lib/opsi/depot/win7/installfiles` Adjust the file access rights by entering:

```
opsi-setup --set-rights /var/lib/opsi/depot/win7/installfiles
```

6.1.4.10 Installation Log files

- `c:\Windows\Panther\setupact.log`:
  Logs until the end of setup phase 4 (running under WinPE)
- `c:\Windows\Panther\setupact.err`:
  Error log including the end of setup phase 4 (running under WinPE)
- `c:\Windows\Panther\UnattendGC\setupact.log`:
  Logs a specialize phase
- `c:\Windows\Panther\UnattendGC\setupact.err`:
  Error log for a specialize phase
- `c:\Windows\System32\Winevt\Logs\*`
- `c:\Windows\ntbtlog.txt` (only when the startup protocol is activated)

6.1.5 Windows Product Key

If you are using the opsi license management module, then you may administrate your Windows product keys using the license management software. Information on how to do this can be found in the opsi manual.

If you don't want to use the license management module, then the product key can simply be made up using the product properties.

While creating a client, you can use the opsi management interface to enter the product key:

- choose a client
- change to the tab *nettboot products*
- select the product (e.g. win7-x64)
- change to the product property *productkey* (on the right lower corner of the opsi management interface)
- type in your key
- leave the input field and save the changes

A other possibility is to use the command line. While working with an opsi server, you can read and/or change the server defaults. To read the server default use (you may need to modify the productId and you must change *<opsiserver.domain.local>* with the fqdn from your opsiserver. Be sure that you enter the commands in one line):

```
opsi-admin -d method productPropertyState_getObjects [] '{"productId":"win10-x64","objectId":"opsiserver.domain.local"\
    }'
```

The easiest way to modify the defaults, is to modify the file, and then update the objects with the modified file.

The first step would be to view the contents of an actual configuration file (you may need to modify the productId and you must change *<opsiserver.domain.local>* with the fqdn from your opsiserver. Be sure that you enter the commands in one line):

```
opsi-admin -d method productPropertyState_getObjects [] '{"productId":"win10-x64","objectId":"opsiserver.domain.local"\
    }' > /tmp/property_config.json
```

The second step would be to modify the file `/tmp/property_config.json`, and change the entries and values. Finally, you must update the objects using this modified file (enter this command in one line):

```
opsi-admin -d method productPropertyState_updateObjects < /tmp/property_config.json
```

You can check that the modifications were successful using the following command (you may need to modify the productId and you must change *<opsiserver.domain.local>* with the fqdn from your opsiserver. Be sure that you enter the commands in one line):

```
opsi-admin -d method productPropertyState_getObjects [] '{"productId":"win10-x64","objectId":"opsiserver.domain.local"\
    }'
```

6.1.6 Start the Windows Installation

To start a windows installation:

- choose a client
- change to the tab *netboot products*
- select the product (e.g. win7-x64)
- set the *action request* to *setup*
- save the changes by clicking the red check mark (which then changes to green)

Now the client should load the opsi-linux-bootimage via the network and start it up. Before the Windows installation starts, you might have to confirm.

Caution
This refers to clients with a hard drive larger than 2 terabyte. On a non UEFI-system the possible largest partition size is 2 terabyte. When you have a larger partiton scheme the installation will fail. This is a technical restriction. You have to configure the system partition with a maximal size of 2 terabyte and therefore configure two partitions. This can be done with the product-properties. On the other hand the UEFI-module bypasses this restriction by using another partition table.

6.1.7 Structure of the Unattended Installation Products

This chapter describes the Windows netboot products.

6.1.7.1 Directory Tree Overview

```
<productid>-
          |-i386/                              NT5 only: Installations files
          |-installfiles/                      NT6 only: Installations files
          |-winpe/                             NT6 only
          |-opsi/                              scripts and templates by opsi.org
          |  |-$oem$/                                 NT5 only: $oem$ according to MS
          |  |-postinst.d/                            scripts after OS-install by opsi.org
          |  !-unattend.(txt/xml).template            Template by opsi.org
          |-custom/                            scripts and templates by customer
          |  |-$oem$/                                 NT5 only: $oem$ according to MS by customer
          |  |-postinst.d/                            scripts after OS-install by customer
          |  !-unattend.(txt/xml)                     unattend.txt by customer
          |-drivers/                           drivers directory
          |  |-drivers/                        drivers directory
          |  |-pciids/                         symbolic links to drivers
          |  |-vendors/                        symbolic links to drivers
          |  |-classes/                        symbolic links to drivers
          |  |-usbids/                         symbolic links to drivers
          |  |-hdaudioids/                     symbolic links to drivers
          |  |-pci.ids                         PCI-IDs DB
          |  !-usb.ids                         USB-IDs DB
          |-setup.py                           installation script
          |-<productid>_<version>.control      meta data (only for info)
          |-<productid>.files                  file list (created automatically)
          |-create_driver_links.py             driver management script
          |-show_drivers.py                    driver management script
          |-download_driver_pack.py            driver management script
          !-extract_driver_pack.py             driver management script
```

6.1.7.2 File Descriptions

- `setup.py`
  This is the installation script which is executed by the boot image.
- `<productid>_<version>.control`
  Contains the meta data of the product as prepared from the package maintainer. These files are here for information purposes only. There will be no effect after changing these files.
- `<productid>.files`
  This file is created automatically and should not be changed.
- `create_driver_links.py`
  `show_drivers.py`
  `download_driver_pack.py`
  `extract_driver_pack.py`
  These are scripts for the simplified driver integration, which is described in its own chapter ("Simplified driver integration for the automatic OS installation").

6.1.7.3 Directory installfiles / winpe

- `installfiles`
  This directory contains the all files from the windows installation DVD (NT6 = Windows 7 and above).
- `winpe`
  From Windows 7, this directory contains a bootable winpe image among other files.

6.1.7.4 Directories opsi and custom

Both directories contain scripts and configuration files for the OS installation. During the installation process, the directories work together in such a way that they give the priority usage to the files in the custom directories.

The opsi directory contains files and templates that are maintained by opsi.org, and maybe replaced during the next update. So it's not a good idea to make specific (or customized) changes to these files in this location. Please use the custom directory for this purpose, because that directory is not subject to any changes by opsi.org.

The subdirectory `postinst.d` contains scripts which are executed after the OS installation is completed by the `postinst.cmd` program. These scripts are needed to install the opsi-client-agent, among other software. The scripts will be executed in alphabetic order. To make it easier to see the order in which the scripts will be executed, the name always starts with a 2 digit number (`10_dhcp.cmd`). If you want to make extensions, then please do so in the custom/postinst.d directory and start numbers between the 10, 20, 30 ,... (e.g. `13_myscript.cmd`). The starting numbers 10, 20, 30,... are reserved for use by opsi org / uib gmbh. The script `99_cleanup.cmd` is the last one and initiate a reboot.

6.1.7.5 Directory drivers

This directory is used for the integration of drivers and is described in the following chapter.

6.1.8 Simplified Driver Integration during the Automatic Windows Installation

If a group of computers have drivers that are not part of the Windows default installation, it's best to put these computers into a pool and integrate their drivers during installation time.

Opsi supports the automatic integration of drivers into the installation, and therefore simplifies driver deployment. In this case, the drivers simply need to be place into the correct directory. When the installation script is called it searches through these directories and creates a catalog. The boot image automatically uses this catalog to embed

the correct drivers. Opsi supports the automatic installation of standard drivers, USB drivers, HD audio drivers, and disk controller drivers (text-mode drivers).

In order for a driver to be immediately installed with the Windows installation, you must place the drivers on the server in a specific format. The drivers must be placed in the drivers directory, with the format **.inf* , where the file name describes the driver for the Windows setup program. The drivers in the *setup.exe* or **.zip* are not used here. If you have a computer that already has the drivers installed, then you can extract the appropriate drivers using the program *double driver* (http://www.boozet.org/dd.htm).

There are many levels of driver integration:

- General driver packages
- Preferred drivers that belong to your hardware, but are not assigned to specific computers
- Drivers that will be manually assigned to computers
- Drivers that will be automatically assigned to the computers using the fields <vendor>/<model>

Below is a detailed discussion about how to include each of these drivers

6.1.8.1 General Driver Packages

When the hardware configuration is very heterogeneous, then it may be reasonable to work with general driver packages.
General drivers can be placed under `./drivers/drivers`.
You can find example of general driver packages here http://driverpacks.net/ .
Download the appropriate driver package to a temporary directory, and then unpack the driver package using the opsi script extract_driver_pack.py as such:

```
./extract_driver_pack.py <path to the temporary directory with the compressed driverpacks>
```

This will unpack and store the drivers in the directory `./drivers/drivers/`.
It may be the case that the drivers found by opsi in this location do not necessarily work with your hardware.
For the drivers which are found in `./drivers/drivers/`, the driver will be matched to the corresponding hardware using the PCI IDs (i.e. USB- or HD_Audio-ID) in the description file, and then integrated into the Windows setup as needed.

6.1.8.2 Preferred Drivers

In the case that you have to support special hardware, and you can find the additional drivers from the manufacturers, then use the following procedure to include them in the installation.
Place the additional drivers in their own directory under:
`./drivers/drivers/preferred`.
(the naming and depth of the directory structure is not important). Drivers that are found in the directory `./drivers/drivers/preferred` will be integrated into the Windows setup, assuming that opsi finds a suitable match to the drive hardware based off of the PCI IDs (i.e. USB or HD_Audo-ID) in the description file.
Problems can occur when the same PCI ID of the drivers is found in `preferred`. In this case, a direct mapping of the drivers to the devices is needed.

6.1.8.3 Drivers that will be Manually Assigned to Computers

When installing additional drivers based on the PCI-IDs or USB-IDs, they should be installed under the directory `./drivers/drivers/additional` (where name and depth of the directory structure is not important). You can map one or more drivers to a client using the Product-Property *additional_drivers* and a list of driver directories under `./drivers/drivers/additional`. The directories specified by *additional_drivers* are searched recursively until all drivers are found. This method can be used to make a specific directory based on the client type (i.e. dell-optiplex-815).

When a driver is found under the drivers directory that is specified by *additional_drivers* and also matches the PCI identifier, then other drivers in `drivers/preferred` or `drivers/` will not be used. Therefore the drivers under *additional_drivers* add functionality that would not have been found with the normal drivers. Also, the drivers that are manually bound to a client using *additional_drivers* receive priority over other drivers (*additional_drivers* can be thought of as *super-preferred*).

6.1.8.4 Drivers that will be Automatically Assigned to the Computers using the Fields <vendor>/<model>

The previously described mechanisms that directly map drivers to devices is automated since the 4.0.2 Release 2 of opsi. The opsi-linux-bootimage will search the directory `./drivers/drivers/additional/byAudit` for a director name that matches the field *Vendor* that was given in the Hardware Inventory. This *Vendor* directory will be search for a *Model* directory that corresponds to what is seen in Hardware Inventory. If this directory is found, then it will be manually assigned to the product property *additional_drivers*.
The directory name *byAudit* is case sensitive. The directory names for *Vendor* and *Model* are not case sensitive (*Dell* and *dELL* are treated the same way).

Since opsi 4.0.5 one can use opsi-configed for uploading the drivers Automatic driver upload (see opsi-manual "Automatic driver upload")

The opsi-linux-bootimage looks first for drivers in

- `<vendor>/<model> (<sku>)`
- If nothing is found as a first fallback at `<vendor>/<model>`
- If nothing is found then as second fallback at motherboard `<vendor>/<motherboard-model>`

6.1.8.5 Structure of the Driver Directory and Driver Files:

```
/var/
  !-lib/
     !-opsi/depot/
        !-<productid>/
           !-drivers
              |-classes/               (Links to driver device classes)
              |-hdaudioids/            (Links to HD-Audio drivers)
              |-pciids/                (Links to PCI-ID drivers)
              |-pci.ids                (PCI database)
              |-usbids/                (Links to USB-ID drivers)
              |-usb.ids                (USB database)
              |-vendors/               (Links to manufacturer drivers)
              !-drivers                (place for general driver packages)
                 |-additional/         (manually assigned drivers)
                    |-byAudit/         Model-specific drivers that
                       |-<vendor>              will be assigned by
                          |-<model>            the Hardware Inventory
                 |-buildin/            (data for the i386 version)
                 |-preferred/          (certified drivers)
                 |-exclude/            (excluded drivers)
                 !-mydriverpacks/      (example driver packages)
```

6.1.8.6 Processing of the Different Levels of Driver Integration

The top priority is given to drivers that are found using the property *additional_drivers* or using the inventory data in `./drivers/drivers/additional/byAudit`. Tests will be made during driver integration to determine which drivers are already installed on which hardware devices. When a driver is not found for a device, the following method will be used to search for drivers.

For devices that have drivers that were not listed in *additional_drivers*, opsi will search for, and integrate, an appropriate driver based off of the PCI ID (ie. USB-, HD_Audio-ID).

Integration of drivers means the following:

- The driver will be copied to the local hard drive at `c:\drv\<num>`.
- The Windows Setup will search for the driver sub-directory under `c:\drv\` , where the sub-directory name will be read from the unattended file in the root *unattend* sub-directory.

6.1.8.7 Driver Addition and Checking

After any changes in the directory `./drivers/drivers` have been made, call the following command from the Netboot Products root directory in order to set the permissions:

```
opsi-setup --set-rights ./drivers
```

Then call the script `./create_driver_links.py`. The script searches through the directories under *./drivers/drivers* and generates a list of links using PCI-IDs, USB-IDs, HD-Audio-IDs that enables hardware to know about drivers. The script will use the drivers in the preferred directories.

The script `setup.py` examines the hardware of the installed computers and identifies the necessary drivers. These will be copied to the disk and the file unattended.txt will be patched. The script `create_driver_links.py` examines the NT5 products one at a time in the *i386* tree and extracts the .inf files of the necessary drivers into *windows_builtin*. If you make a change to the i386 directory tree (i.e. after installing a service pack), then delete that directory and run `create_driver_links.py` again. The recognized drivers for NT6 products are found in WinPE at *windows_builtin*.

With the following command, one can see the hardware inventory for a client:

```
./show_drivers.py <clientname>
```

When this is command is called, it will show a selection for which drivers would be chosen for installation to the Bootimage via PCI-IDs, USB-IDs, HD-Audio-IDs, and *additional_drivers* (or *byAudit*), and which hardware still has no driver.

Use the output of `show_drivers.py` to check and see which drivers need to be installed.

It could be that manufacturers include different drivers for different operating systems (i.e. Vista vs. Win7) or different configurations (ie. SATA vs. SATA RAID). The `create_driver_links.py` cannot make this distinction. If you think the wrong driver has been installed, then move the driver to the *drivers/exclude* directory and then call `create_driver_links.py` again. Drivers in the directory *drivers/exclude* are not used during the integration.

Example output of a `show_drivers.py` call:

```
./show_drivers.py pcdummy

PCI-Devices
   [(StandardsystemgerÃ¤te), PCI Standard-PCI-zu-PCI-Bridge]
      No driver - device directory  /var/lib/opsi/depot/<productid>/drivers/pciids/1022/9602 not found
   [ATI Technologies Inc., Rage Fury Pro (Microsoft Corporation)]
      Using build-in windows driver
   [(Standard-IDE-ATA/ATAPI-Controller), Standard-Zweikanal-PCI-IDE-Controller]
      /var/lib/opsi/depot/<productid>/drivers/drivers/D/M/N/123
   [Realtek Semiconductor Corp., Realtek RTL8168C(P)/8111C(P) PCI-E Gigabit Ethernet NIC]
      /var/lib/opsi/depot/<productid>/drivers/drivers/preferred/realtek_gigabit_net_8111_8168b
   [IEEE 1394 OHCI-konformer Hostcontroller-Hersteller, OHCI-konformer IEEE 1394-Hostcontroller]
      No driver - device directory '/var/lib/opsi/depot/<productid>/drivers/pciids/197B/2380' not found
   [Advanced Micro Devices, Inc., AMD AHCI Compatible RAID Controller]
      /var/lib/opsi/depot/<productid>/drivers/drivers/preferred/ati_raid_sb7xx
   [(Standard-USB-Hostcontroller), Standard OpenHCD USB-Hostcontroller]
      No driver - device directory '/var/lib/opsi/depot/<productid>/drivers/pciids/1002/4397' not found
   [ATI Technologies Inc, ATI SMBus]
      /var/lib/opsi/depot/<productid>/drivers/drivers/preferred/ati_smbus

USB-Devices
   [(Standard-USB-Hostcontroller), USB-VerbundgerÃ¤t]
      /var/lib/opsi/depot/<productid>/drivers/drivers/preferred/brother_844x_pGerb
   [Microsoft, USB-DruckerunterstÃ¼tzung]
      /var/lib/opsi/depot/<productid>/drivers/drivers/preferred/brother_844x_pGerb
```

```
Additional drivers
   [ati_hdaudio_azalia]
     /var/lib/opsi/depot/<productid>/drivers/drivers/additional/ati_hdaudio_azalia
```

Example with *additional_drivers*:

```
 ./show_drivers.py e5800
Manually selected drivers (additional)
   [hp_e5800]
      [/var/lib/opsi/depot/<productid>/drivers/drivers/additional/hp_e5800/sp52852/Vista64/HDXHPAI3.inf]
      [/var/lib/opsi/depot/<productid>/drivers/drivers/additional/hp_e5800/sp52852/Vista64/HDX861A.inf]
      [/var/lib/opsi/depot/<productid>/drivers/drivers/additional/hp_e5800/sp52852/Vista64/HDXHPAI1.inf]
      [/var/lib/opsi/depot/<productid>/drivers/drivers/additional/hp_e5800/sp52852/Vista64/HDXCPC.inf]
      [/var/lib/opsi/depot/<productid>/drivers/drivers/additional/hp_e5800/sp52852/Vista64/HDXHPAI2.inf]
      [/var/lib/opsi/depot/<productid>/drivers/drivers/additional/hp_e5800/sp50134/autorun.inf]
      [/var/lib/opsi/depot/<productid>/drivers/drivers/additional/hp_e5800/sp50134/ibxHDMI/IntcDAud.inf]
      [/var/lib/opsi/depot/<productid>/drivers/drivers/additional/hp_e5800/sp50134/HDMI/IntcHdmi.inf]
      [/var/lib/opsi/depot/<productid>/drivers/drivers/additional/hp_e5800/sp50134/Graphics/kit24890.inf]
      [/var/lib/opsi/depot/<productid>/drivers/drivers/additional/hp_e5800/sp50134/IIPS/Impcd.inf]
      [/var/lib/opsi/depot/<productid>/drivers/drivers/additional/hp_e5800/sp54284/Realtek 64bit/hp64win7.inf]

PCI-Devices
   [8086:27C8]  Intel : Intel(R) N10/ICH7 Family USB Universal Host Controller - 27C8
      /var/lib/opsi/depot/<productid>/drivers/drivers/preferred/R293337/WIN7
   [8086:27DA]  Intel : Intel(R) N10/ICH7 Family SMBus Controller - 27DA
      /var/lib/opsi/depot/<productid>/drivers/drivers/preferred/R293337/WIN7
   [8086:27C9]  Intel : Intel(R) N10/ICH7 Family USB Universal Host Controller - 27C9
      /var/lib/opsi/depot/<productid>/drivers/drivers/preferred/R293337/WIN7
   [8086:27DF]  Intel : Intel(R) ICH7 Family Ultra ATA Storage Controllers - 27DF
      /var/lib/opsi/depot/<productid>/drivers/drivers/preferred/R293337/WIN7
   [8086:27CA]  Intel : Intel(R) N10/ICH7 Family USB Universal Host Controller - 27CA
      /var/lib/opsi/depot/<productid>/drivers/drivers/preferred/R293337/WIN7
   [8086:2E30]  Intel : Intel(R) 4 Series Chipset Processor to I/O Controller - 2E30
      /var/lib/opsi/depot/<productid>/drivers/drivers/not_preferred/x64/C/Intel/1
   [8086:27CB]  Intel : Intel(R) N10/ICH7 Family USB Universal Host Controller - 27CB
      /var/lib/opsi/depot/<productid>/drivers/drivers/preferred/R293337/WIN7
   [8086:2E32]  Intel Corporation : Intel(R) G41 Express Chipset
      Manually selected [hp_e5800] /var/lib/opsi/depot/<productid>/drivers/drivers/additional/hp_e5800/sp50134/Graphics
   [8086:27CC]  Intel : Intel(R) N10/ICH7 Family USB2 Enhanced Host Controller - 27CC
      /var/lib/opsi/depot/<productid>/drivers/drivers/preferred/R293337/WIN7
   [8086:244E]  Intel : Intel(R) 82801 PCI-BrÃ¼cke - 244E
      Using build-in windows driver
      This driver will not be integrated, because same device already integrated in: '/var/lib/opsi/depot/<productid>/\
    drivers/drivers/not_preferred/x64/C/Intel/1/dmi_pci.inf'
   [8086:27D0]  Intel : Intel(R) N10/ICH7 Family PCI Express Root Port - 27D0
      /var/lib/opsi/depot/<productid>/drivers/drivers/preferred/R293337/WIN7
   [8086:27B8]  Intel : Intel(R) ICH7 Family LPC Interface Controller - 27B8
      /var/lib/opsi/depot/<productid>/drivers/drivers/preferred/R293337/WIN7
   [8086:27D2]  Intel : Intel(R) N10/ICH7 Family PCI Express Root Port - 27D2
      /var/lib/opsi/depot/<productid>/drivers/drivers/preferred/R293337/WIN7
   [8086:27C0]  Intel : Intel(R) N10/ICH7 Family Serial ATA Storage Controller - 27C0
      /var/lib/opsi/depot/<productid>/drivers/drivers/preferred/R293337/WIN7
   [8086:27D8]  Microsoft : High Definition Audio-Controller
      No driver - device directory '/var/lib/opsi/depot/<productid>/drivers/pciids/8086/27D8' not found
   [10EC:8136]  Realtek : Realtek RTL8102E/RTL8103E-Familie-PCI-E-Fast-Ethernet-NIC (NDIS 6.20)
      Manually selected [hp_e5800] /var/lib/opsi/depot/<productid>/drivers/drivers/additional/hp_e5800/sp54284/Realtek \
    64bit

USB-Devices
   [0461:0010]  (StandardsystemgerÃ¤te) : USB-EingabegerÃ¤t
      No driver - vendor directory '/var/lib/opsi/depot/<productid>/drivers/usbids/0461' not found
   [0461:4D20]  (StandardsystemgerÃ¤te) : USB-EingabegerÃ¤t
      No driver - vendor directory '/var/lib/opsi/depot/<productid>/drivers/usbids/0461' not found
   [058F:6366]  Kompatibles USB-SpeichergerÃ¤t : USB-MassenspeichergerÃ¤t
      No driver - vendor directory '/var/lib/opsi/depot/<productid>/drivers/usbids/058F' not found
   [0461:0010]  (Standard-USB-Hostcontroller) : USB-VerbundgerÃ¤t
      No driver - vendor directory '/var/lib/opsi/depot/<productid>/drivers/usbids/0461' not found
```

```
HD-Audio-Devices
   [10EC:0662]  Realtek High Definition Audio
      Manually selected [hp_e5800] /var/lib/opsi/depot/<productid>/drivers/drivers/additional/hp_e5800/sp52852/Vista64
```

Example with *byAudit*:

```
 ./show_drivers.py pctry5detlef
Manually selected drivers (additional)
   [/var/lib/opsi/depot/<productid>/drivers/drivers/additional/byAudit/nvidia/awrdacpi]
      [/var/lib/opsi/depot/<productid>/drivers/drivers/additional/byAudit/nvidia/awrdacpi/pctry5detlef/Display/Radeon \
     X300-X550-X1050 Series Secondary (Microsoft Corporation - WDDM)/atiilhag.inf]
      [/var/lib/opsi/depot/<productid>/drivers/drivers/additional/byAudit/nvidia/awrdacpi/pctry5detlef/Display/Radeon \
     X300-X550-X1050 Series (Microsoft Corporation - WDDM)/atiilhag.inf]
      [/var/lib/opsi/depot/<productid>/drivers/drivers/additional/byAudit/nvidia/awrdacpi/pctry5detlef/MEDIA/Realtek AC\
     '97 Audio/oem21.inf]

PCI-Devices
   [1002:5B70]  ATI Technologies Inc. : Radeon X300/X550/X1050 Series Secondary (Microsoft Corporation - WDDM)
      Manually selected [/var/lib/opsi/depot/<productid>/drivers/drivers/additional/byAudit/nvidia/awrdacpi] /var/lib/\
     opsi/depot/<productid>/drivers/drivers/additional/byAudit/nvidia/awrdacpi/pctry5detlef/Display/Radeon X300-X550-\
     X1050 Series Secondary (Microsoft Corporation - WDDM)
      Multiple selected [/var/lib/opsi/depot/<productid>/drivers/drivers/additional/byAudit/nvidia/awrdacpi] /var/lib/\
     opsi/depot/<productid>/drivers/drivers/additional/byAudit/nvidia/awrdacpi/pctry5detlef/Display/Radeon X300-X550-\
     X1050 Series (Microsoft Corporation - WDDM)
   [10DE:0053]  (Standard-IDE-ATA/ATAPI-Controller) : Standard-Zweikanal-PCI-IDE-Controller
      No driver - device directory '/var/lib/opsi/depot/<productid>/drivers/pciids/10DE/0053' not found
   [10DE:005D]  (StandardsystemgerÃ¤te) : PCI Standard-PCI-zu-PCI-BrÃ¼cke
      No driver - device directory '/var/lib/opsi/depot/<productid>/drivers/pciids/10DE/005D' not found
   [1022:1100]  AMD : AMD HyperTransport(tm)-Konfiguration
      Using build-in windows driver
   [10DE:0054]  (Standard-IDE-ATA/ATAPI-Controller) : Standard-Zweikanal-PCI-IDE-Controller
      /var/lib/opsi/depot/<productid>/drivers/drivers/preferred/evb_potsdam_fsc__esprimo_p625/\
     FTS_NVIDIASATAAHCIDRIVERVISTA64V103042MCP78__1026963/NVIDIA_SATA_AHCI_DRIVER_Vista64_V10.3.0.42_MCP78 (textmode \
     capable)
   [1022:1101]  AMD : AMD-Adresszuordnungskonfiguration
      Using build-in windows driver
   [10DE:0055]  (Standard-IDE-ATA/ATAPI-Controller) : Standard-Zweikanal-PCI-IDE-Controller
      /var/lib/opsi/depot/<productid>/drivers/drivers/preferred/evb_potsdam_fsc__esprimo_p625/\
     FTS_NVIDIASATAAHCIDRIVERVISTA64V103042MCP78__1026963/NVIDIA_SATA_AHCI_DRIVER_Vista64_V10.3.0.42_MCP78 (textmode \
     capable)
   [1022:1102]  AMD : AMD DRAM und HyperTransport(tm)-Nachverfolgungsmoduskonfiguration
      Using build-in windows driver
   [10DE:0057]  NVIDIA : NVIDIA nForce-Netzwerkcontroller
      Using build-in windows driver
   [1022:1103]  AMD : Sonstige AMD-Konfiguration
      Using build-in windows driver
   [10DE:0059]  Realtek : Realtek AC'97 Audio
      Manually selected [/var/lib/opsi/depot/<productid>/drivers/drivers/additional/byAudit/nvidia/awrdacpi] /var/lib/\
     opsi/depot/<productid>/drivers/drivers/additional/byAudit/nvidia/awrdacpi/pctry5detlef/MEDIA/Realtek AC'97 Audio
   [10DE:005E]  NVIDIA : NVIDIA nForce4 HyperTransport-BrÃ¼cke
      /var/lib/opsi/depot/<productid>/drivers/drivers/preferred/ga-ma78-pcbon4/chipset_win7-64/SMBUS
   [104C:8025]  Texas Instruments : OHCI-konformer Texas Instruments 1394-Hostcontroller
      No driver - device directory '/var/lib/opsi/depot/<productid>/drivers/pciids/104C/8025' not found
   [10DE:005A]  (Standard-USB-Hostcontroller) : Standard OpenHCD USB-Hostcontroller
      No driver - device directory '/var/lib/opsi/depot/<productid>/drivers/pciids/10DE/005A' not found
   [10DE:0050]  (StandardsystemgerÃ¤te) : PCI Standard-ISA-BrÃ¼cke
      No driver - device directory '/var/lib/opsi/depot/<productid>/drivers/pciids/10DE/0050' not found
   [10DE:005B]  (Standard-USB-Hostcontroller) : Standard PCI-zu-USB erweiterter Hostcontroller
      No driver - device directory '/var/lib/opsi/depot/<productid>/drivers/pciids/10DE/005B' not found
   [1002:5B60]  ATI Technologies Inc. : Radeon X300/X550/X1050 Series (Microsoft Corporation - WDDM)
      Manually selected [/var/lib/opsi/depot/<productid>/drivers/drivers/additional/byAudit/nvidia/awrdacpi] /var/lib/\
     opsi/depot/<productid>/drivers/drivers/additional/byAudit/nvidia/awrdacpi/pctry5detlef/Display/Radeon X300-X550-\
     X1050 Series Secondary (Microsoft Corporation - WDDM)
      Multiple selected [/var/lib/opsi/depot/<productid>/drivers/drivers/additional/byAudit/nvidia/awrdacpi] /var/lib/\
     opsi/depot/<productid>/drivers/drivers/additional/byAudit/nvidia/awrdacpi/pctry5detlef/Display/Radeon X300-X550-\
     X1050 Series (Microsoft Corporation - WDDM)
   [10DE:0052]  NVIDIA : NVIDIA nForce PCI-Systemverwaltung
```

```
      Using build-in windows driver
   [10DE:005C]  (StandardsystemgerÃ¤te) : PCI Standard-PCI-zu-PCI-BrÃ¼cke
      No driver - device directory '/var/lib/opsi/depot/<productid>/drivers/pciids/10DE/005C' not found

USB-Devices
   [1241:1111]  (StandardsystemgerÃ¤te) : USB-EingabegerÃ¤t
      No driver - vendor directory '/var/lib/opsi/depot/<productid>/drivers/usbids/1241' not found

HD-Audio-Devices
   No devices installed
```

TIPS

- Directory names `NDIS1` contain Vista-Drivers ; `NDIS2` contain Win7-Driver
- Some chip drivers contain description files, which specify hardware without actually providing drivers. An example would be the `cougar.inf` or `ibexahci.inf` from Intel. If such a *Pseudo-Driver* were to be placed in *additional_drivers* (or *byAudit*), then the other drivers in the *preferred* subdirectory may be excluded. It is better to move these directories to the *preferred* subdirectory.
- SATA drivers and SATA-RAID drivers refer to the same PCI ID. A SATA-RAID driver will not function with a single-disk system.
- Check the output of `./show_drivers.py` carefully !

Chapter 7

Integration of New Software Packages into the opsi Server

The primary objective of software distribution is to accomplish automatic software installation without user interaction. Software installation and user activity should be strictly separated. In most cases, the installation process requires administrative privileges which the user usually doesn't have. So the installation process has to be done independently from the user. This way, the user can neither interfere nor be affected by the software installation process.

In order to do this, you have to write a script for the script driven installer, which is called an *opsi-winst* script.

7.1 A Brief Tutorial: How to write a opsi-winst Script

7.1.1 Introduction

This tutorial merely helps you getting started with opsi. It can't replace professional training (which you may order through uib), or thoroughly studying the complete opsi manuals (which might be time consuming and partially error prone if you lack background knowledge). uib now offers training in English, too.

Training and Support: Get Training by uib gmbh in Europe or possibly Northern America:
http://uib.de/en/support-training/support/

Manuals: The opsi Manuals can be found at: http://uib.de/en/opsi-documentation/documentation/ important for scripting:
opsi-winst reference card and opsi-winst manual

Wiki (Scripts, Tips, Links): http://forum.opsi.org/wiki

Support Forum (fast and free vendor support): http://forum.opsi.org

7.1.2 Methods of Non-Interactive Installation

Regardless of whether or not you are using opsi or another management product, there are three different ways to install software without user interaction:

1. **Unattended or Silent Installation**
   The original setup programs from the software manufacturer can be executed from within a opsi-winst script in *silent* or *unattended* mode. It depends on whether or not the program supports a silent installation mode. A special case of this method is the unattended installation of MSI packages.

2. **Interactive Setup with recorded Answers**
   The answers provided by the opsi administrator, while running the manufacturer's setup program during a pre-installation, can be automatically saved using the free tool *Autoit* or Autohotkey. That requires providing an autoIt script for an unattended installation.

3. **Analyze and Repackaging**
   The standard setup can be analyzed and *recorded* by Windows to do the installation tasks by the *opsi-winst* program. Usually that is something like file installation to the local file system, followed by patching the registry

---

Note
opsi supports all of these variants.
Usually a combination of all three different methods in one script provides the best result. For example, performing the basic installation using the original setup if available, and then doing some customizing by patching the registry or the file based configuration.

7.1.3 Structure of a opsi-script / opsi-winst Script

An example of a simpleopsi-winst script:

```
[Actions]
WinBatch_tightvnc_silent_install

[WinBatch_tightvnc_silent_install]
"%ScriptPath%\tightvnc-1.3.9-setup.exe" /silent
```

An opsi-winst script contains a **primary** and a **secondary** section. The section headers are in square brackets, similar to what you may have seen in ini-files. The primary section is noted by the identifier [Actions], and the secondary section is noted by the identifier [WinBatch_...].

The core work, like starting programs or copying files, is done in the secondary sections, not in the primary sections. These secondary sections are topic specific, and have a specific syntax that relates to their specific topic.

The name of a secondary section starts with a reserved word for that type of secondary section followed by a free identifier.

In the above example, the primary section `[Actions]` calls a secondary section `[WinBatch_tightvnc_silent_install]`.
This secondary section has the type `WinBatch`. The content of the secondary sections, of type *WinBatch*, are executed by the Windows API. In this case, the program `tightvnc-1.3.9-setup.exe` will be started with the parameter `/silent`.

7.1.4 Primary Sections

Initial
The Initial section is used to set runtime parameters.
This section is optional.

Actions
The section `[Actions]` is the main program.
Any part of the code that is called more then one time can be placed in sub sections.

Sub-sections
Primary sections which may be called multiple times or have their code in external files.

The primary sections are the main program which control the program flow. There you will find:

- Variables: strings and string lists
- if else endif statements
- for loops that traverse string lists
- Functions

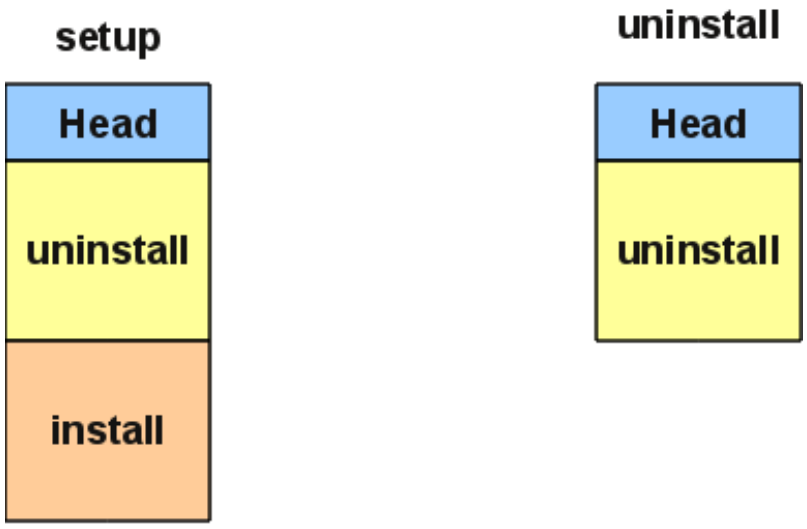

Figure 7.1: double code for deinstallation

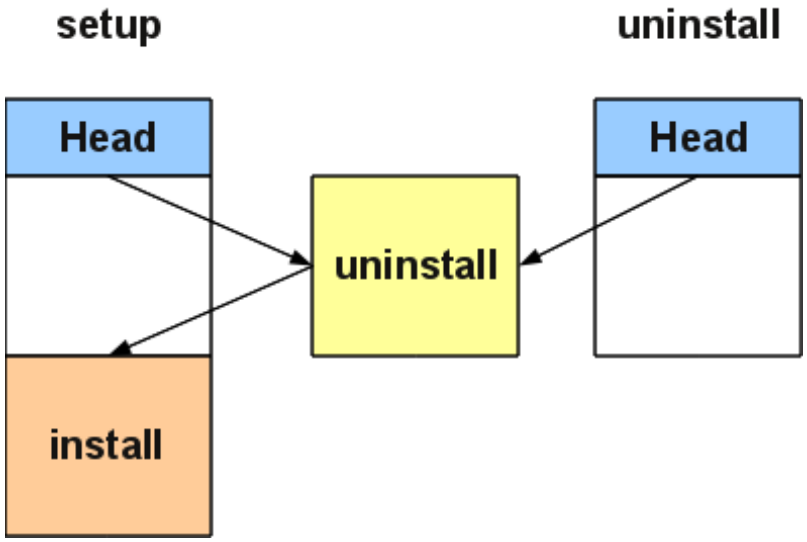

Figure 7.2: avoid double code by using sub sections

7.1.5 Important Kinds of Secondary Sections

Files
File operations include

- copying (regarding the internal version information, recursive, ...)

- deleting files or directories
- creating directories

WinBatch
It's used for calling programs using the Windows API. For example, WinBatch calls the setup programs in the silent mode.

DosBatch/DosInAnIcon
The content of these sections are interpreted by the `cmd.exe` like normal batch files.
A variant of *DosBatch* is *DosInAnIcon* which is run in a minimized window.

ExecWith
A program is given as a parameter, and then that program interprets the content of this section (e.g. AutoIt).

Registry
The *Registry* sections are used for registry manipulations.

Linkfolder
Link folder sections are used for the manipulation of start menus and desktop icons.

7.1.6 Global Constants

Global constants are placeholders which can be used in primary and secondary sections. These placeholders are replaced by their values at runtime.

Examples:

%ProgramFiles32Dir%
c:\program files

%Systemroot%
c:\windows

%System%
c:\windows\system32

%Systemdrive%
c:\

%Scriptpath%
<path to the running script>

7.1.7 Second Example: tightvnc

The following example shows a simple script that is used for a tightvnc installation. This script should contain only the winbatch call for the silent installation. If you call the sub-section silent installation more the one time, a confirmation window appears (which is a bug in the installer). This confirmation window will be closed by a *autoit* script if it appears.

tightvnc.ins:

```
[Actions]
Message "Install tightvnc 1.3.9 ..."
ExecWith_autoit_confirm "%ScriptPath%\autoit3.exe" WINST /letThemGo
WinBatch_tightvnc_silent_install
KillTask "autoit3.exe"

[WinBatch_tightvnc_silent_install]
"%ScriptPath%\tightvnc-1.3.9-setup.exe" /silent
```

```
[ExecWith_autoit_confirm]
; Wait for the confirm dialog which only appears if tightvnc was installed before as service
; Waiting for the window to appear
WinWait("Confirm")
; Activate (move focus to) window
WinActivate("Confirm")
; Choose answer no
Send("N")
```

7.1.8 Elementary Commands for Primary Sections

7.1.8.1 String Variable

Declaration of a variable
DefVar <variable name>

Setting a value
Set <variable name> = <value>

Example:

```
DefVar $ProductId$
Set $ProductId$ = "firefox"
```

Important
The use of string variables is different in primary versus secondary sections. In the primary section, the string variables are handled as independent objects. String variables can only be declared and set to values in primary sections. Therefore you have to use a operator (+) to concatenate variables and strings in a string expression.
Example:`"Installing "+$ProductId$+" ..."`
In secondary sections string variables are used as a placeholder for their values.
Example: `"Installing $ProductId$ ..."`
You should keep this in mind if you cut and paste string expressions between primary and secondary sections.
The advantage of handling string variables in this format is that is possible to use these variables in secondary sections that are interpreted by other programs (DosBatch / Execwith).

7.1.8.2 Message / showbitmap

Displaying text during runtime:
`Message <string>`

Example:

```
Message "Installing "+ $ProductId$ +" ..."
```

Displaying a picture during installation:
`ShowBitMap [<file name>] [<sub title>]`

Example:

```
ShowBitmap "%ScriptPath%\python.png" "Python"
```

7.1.8.3 if [else] endif

Syntax:

```
if <condition>
        ;statement(s)
[
else
        ;statement(s)
]
endif
```

7.1.8.4 Functions

HasMinimumSpace
Check for free space on the hard disk

FileExists
Check for the existence of a file or directory

7.1.8.5 Error, Logging and Comments

comment char *;*
Lines starting with the *;* char are simply ignored.

comment
writes a comment to the log file

LogError
writes error messages to the log file

isFatalError
aborts the script, and return the installation state *failed* to the server.

7.1.8.6 Requirements

requiredWinstVersion
Minimum required version of opsi-winst

7.1.9 Third example: The Generic Template *opsi-template*

This third template should be used as a rough guide whenever you create your own opsi product. Do not cut-and-paste from this manual, but instead look at http://download.uib.de for a new version of the *opsi-template* product package. Using the opsi-package-manager command you may install *opsi-template* (-i) or extract (-x) at your server and then grab the scripts.

setup32.opsiscript: installation script

```
; Copyright (c) uib gmbh (www.uib.de)
; This sourcecode is owned by uib
; and published under the Terms of the General Public License.
; credits: http://www.opsi.org/en/credits/

[Actions]
```

```
requiredWinstVersion >= "4.11.4.6"
ScriptErrorMessages=off

DefVar $MsiId$
DefVar $UninstallProgram$
DefVar $LogDir$
DefVar $ProductId$
DefVar $MinimumSpace$
DefVar $InstallDir$
DefVar $ExitCode$
DefVar $LicenseRequired$
DefVar $LicenseKey$
DefVar $LicensePool$
DefVar $displayName32$
DefVar $displayName64$

DefStringlist $msilist$

Set $LogDir$ = "%opsiLogDir%"

; ----------------------------------------------------------------
; - Please edit the following values                             -
; ----------------------------------------------------------------
;$ProductId$ should be the name of the product in opsi
; therefore please: only lower letters, no umlauts,
; no white space use '-' as a separator
Set $ProductId$       = "opsi-template"
Set $MinimumSpace$    = "1 MB"
; the path where the product will be found after the installation
Set $InstallDir$      = "%ProgramFiles32Dir%\<path to the product>"
Set $LicenseRequired$ = "false"
Set $LicensePool$     = "p_" + $ProductId$
; ----------------------------------------------------------------

if not(HasMinimumSpace ("%SystemDrive%", $MinimumSpace$))
        LogError "Not enough space on %SystemDrive%, " + $MinimumSpace$ + " on drive %SystemDrive
    % needed for " + $ProductId$
        isFatalError "No Space"
        ; Stop process and set installation status to failed
else
        comment "Show product picture"
        ShowBitmap "%ScriptPath%\" + $ProductId$ + ".png" $ProductId$

        if FileExists("%ScriptPath%\delsub32.opsiscript")
                comment "Start uninstall sub section"
                Sub "%ScriptPath%\delsub32.opsiscript"
        endif

        Message "Installing " + $ProductId$ + " ..."

        if $LicenseRequired$ = "true"
                comment "Licensing required, reserve license and get license key"
                Sub_get_licensekey
        endif

        comment "Start setup program"
        ChangeDirectory "%SCRIPTPATH%"
```

```
        Winbatch_install
        Sub_check_exitcode

        comment "Copy files"
        Files_install /32Bit

        comment "Patch Registry"
        Registry_install /32Bit

        comment "Create shortcuts"
        LinkFolder_install

endif

[Winbatch_install]
; Choose one of the following examples as basis for your installation
; You can use $LicenseKey$ var to pass a license key to the installer
;
; === Nullsoft Scriptable Install System
    ================================================================
; "%ScriptPath%\Setup.exe" /S
;
; === MSI package
    ============================================================================================
; You may use the parameter PIDKEY=$Licensekey$
; msiexec /i "%ScriptPath%\some.msi" /l* "$LogDir$\$ProductId$.install_log.txt" /qb-! ALLUSERS=1
    REBOOT=ReallySuppress
;
; === InstallShield + MSI
    ============================================================================================
; Attention: The path to the log file should not contain any whitespaces
; "%ScriptPath%\setup.exe" /s /v" /l* $LogDir$\$ProductId$.install_log.txt /qb-! ALLUSERS=1
    REBOOT=ReallySuppress"
; "%ScriptPath%\setup.exe" /s /v" /qb-! ALLUSERS=1 REBOOT=ReallySuppress"
;
; === InstallShield
    ============================================================================================
; Create setup.iss answer file by running: setup.exe /r /f1"c:\setup.iss"
; You may use an answer file by the parameter /f1"c:\setup.iss"
; "%ScriptPath%\setup.exe" /s /sms /f2"$LogDir$\$ProductId$.install_log.txt"
;
; === Inno Setup
    ============================================================================================
; http://unattended.sourceforge.net/InnoSetup_Switches_ExitCodes.html
; You may create setup answer file by: setup.exe /SAVEINF="filename"
; You may use an answer file by the parameter /LOADINF="filename"
; "%ScriptPath%\setup.exe" /sp- /silent /norestart /nocancel /SUPPRESSMSGBOXES

[Files_install]
; Example of recursively copying some files into the installation directory:
;
; copy -s "%ScriptPath%\files\*.*" "$InstallDir$"

[Registry_install]
; Example of setting some values of an registry key:
;
; openkey [HKEY_LOCAL_MACHINE\Software\$ProductId$]
```

```
; set "name1" = "some string value"
; set "name2" = REG_DWORD:0001
; set "name3" = REG_BINARY:00 af 99 cd

[LinkFolder_install]
; Example of deleting a folder from AllUsers startmenu:
;
; set_basefolder common_programs
; delete_subfolder $ProductId$
;
; Example of creating an shortcut to the installed exe in AllUsers startmenu:
;
; set_basefolder common_programs
; set_subfolder $ProductId$
;
; set_link
;       name: $ProductId$
;       target: <path to the program>
;       parameters:
;       working_dir: $InstallDir$
;       icon_file:
;       icon_index:
; end_link
;
; Example of creating an shortcut to the installed exe on AllUsers desktop:
;
; set_basefolder common_desktopdirectory
; set_subfolder ""
;
; set_link
;       name: $ProductId$
;       target: <path to the program>
;       parameters: <some_param>
;       working_dir: $InstallDir$
;       icon_file: <path to icon file>
;       icon_index: 2
; end_link

[Sub_get_licensekey]
if opsiLicenseManagementEnabled
        comment "License management is enabled and will be used"

        comment "Trying to get a license key"
        Set $LicenseKey$ = demandLicenseKey ($LicensePool$)
        ; If there is an assignment of exactly one licensepool to the product the following call
    is possible:
        ; Set $LicenseKey$ = demandLicenseKey ("", $ProductId$)
        ;
        ; If there is an assignment of a license pool to a windows software id, it is possible to
     use:
        ; DefVar $WindowsSoftwareId$
        ; $WindowsSoftwareId$ = "..."
        ; Set $LicenseKey$ = demandLicenseKey ("", "", $WindowsSoftwareId$)

        DefVar $ServiceErrorClass$
        set $ServiceErrorClass$ = getLastServiceErrorClass
        comment "Error class: " + $ServiceErrorClass$
```

```
        if $ServiceErrorClass$ = "None"
                comment "Everything fine, we got the license key '" + $LicenseKey$ + "'"
        else
                if $ServiceErrorClass$ = "LicenseConfigurationError"
                        LogError "Fatal: license configuration must be corrected"
                        LogError getLastServiceErrorMessage
                        isFatalError
                else
                        if $ServiceErrorClass$ = "LicenseMissingError"
                                LogError "Fatal: required license is not supplied"
                                isFatalError
                        endif
                endif
        endif
else
        LogError "Fatal: license required, but license management not enabled"
        isFatalError
endif


[Sub_check_exitcode]
comment "Test for installation success via exit code"
set $ExitCode$ = getLastExitCode
; informations to exit codes see
; http://msdn.microsoft.com/en-us/library/aa372835(VS.85).aspx
; http://msdn.microsoft.com/en-us/library/aa368542.aspx
if ($ExitCode$ = "0")
        comment "Looks good: setup program gives exitcode zero"
else
        comment "Setup program gives a exitcode unequal zero: " + $ExitCode$
        if ($ExitCode$ = "1605")
                comment "ERROR_UNKNOWN_PRODUCT  1605    This action is only valid for products
    that are currently installed."
                comment "Uninstall of a not installed product failed - no problem"
        else
                if ($ExitCode$ = "1641")
                        comment "looks good: setup program gives exitcode 1641"
                        comment "ERROR_SUCCESS_REBOOT_INITIATED 1641    The installer has
    initiated a restart. This message is indicative of a success."
                else
                        if ($ExitCode$ = "3010")
                                comment "looks good: setup program gives exitcode 3010"
                                comment "ERROR_SUCCESS_REBOOT_REQUIRED  3010    A restart is
    required to complete the install. This message is indicative of a success."
                        else
                                logError "Fatal: Setup program gives an unknown exitcode unequal
    zero: " + $ExitCode$
                                isFatalError
                        endif
                endif
        endif
endif
```

delsub32.opsiscript: external deinstallation sub section

```
; Copyright (c) uib gmbh (www.uib.de)
```

```
; This sourcecode is owned by uib gmbh
; and published under the Terms of the General Public License.
; credits: http://www.opsi.org/en/credits/

Set $MsiId$ = '{XXXXXXXX-XXXX-XXXX-XXXX-XXXXXXXXXXXX}'
Set $UninstallProgram$ = $InstallDir$ + "\uninstall.exe"

Message "Uninstalling " + $ProductId$ + " ..."

if FileExists($UninstallProgram$)
        comment "Uninstall program found, starting uninstall"
        Winbatch_uninstall
        sub_check_exitcode
endif
if not (GetRegistryStringValue32("[HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\
    Uninstall\" + $MsiId$ + "] DisplayName") = "")
        comment "MSI id " + $MsiId$ + " found in registry, starting msiexec to uninstall"
        Winbatch_uninstall_msi
        sub_check_exitcode
endif

comment "Delete files"
Files_uninstall /32Bit

comment "Cleanup registry"
Registry_uninstall /32Bit

comment "Delete program shortcuts"
LinkFolder_uninstall

[Winbatch_uninstall]
; Choose one of the following examples as basis for program uninstall
;
; === Nullsoft Scriptable Install System
    ================================================================
; maybe better called as
; Winbatch_uninstall /WaitforProcessending "Au_.exe" /Timeoutseconds 10
; "$UninstallProgram$" /S
;
; === Inno Setup
    ============================================================================================
; "$UninstallProgram$" /silent /norestart /SUPPRESSMSGBOXES /nocancel

[Winbatch_uninstall_msi]
msiexec /x $MsiId$ /qb-! REBOOT=ReallySuppress

[Files_uninstall]
; Example for recursively deleting the installation directory:
;
; del -sf "$InstallDir$\"

[Registry_uninstall]
; Example of deleting a registry key:
;
; deletekey [HKEY_LOCAL_MACHINE\Software\$ProductId$]

[LinkFolder_uninstall]
```

```
; Example of deleting a folder from AllUsers startmenu:
;
; set_basefolder common_programs
; delete_subfolder $ProductId$
;
; Example of deleting a shortcut from AllUsers desktop:
;
; set_basefolder common_desktopdirectory
; set_subfolder ""
; delete_element $ProductId$

[Sub_check_exitcode]
;(.... see above .....)
```

uninstall32.opsiscript: deinstallation script

```
requiredWinstVersion >= "4.11.4.6"
ScriptErrorMessages=off

DefVar $MsiId$
DefVar $UninstallProgram$
DefVar $LogDir$
DefVar $ExitCode$
DefVar $ProductId$
DefVar $InstallDir$
DefVar $LicenseRequired$
DefVar $LicensePool$

Set $LogDir$ = "%opsiLogDir%"

; ----------------------------------------------------------------
; - Please edit the following values                             -
; ----------------------------------------------------------------
Set $ProductId$       = "opsi-template"
Set $InstallDir$      = "%ProgramFiles32Dir%\<path to the product>"
Set $LicenseRequired$ = "false"
Set $LicensePool$     = "p_" + $ProductId$
; ----------------------------------------------------------------


comment "Show product picture"
ShowBitmap "%ScriptPath%\" + $ProductId$ + ".png" $ProductId$

Message "Uninstalling " + $ProductId$ + " ..."

if FileExists("%ScriptPath%\delsub32.opsiscript")
        comment "Start uninstall sub section"
        Sub "%ScriptPath%\delsub32.opsiscript"
endif

if $LicenseRequired$ = "true"
        comment "Licensing required, free license used"
        Sub_free_license
endif

[Sub_free_license]
comment "License management is enabled and will be used"
```

```
comment "Trying to free license used for the product"
DefVar $result$
Set $result$ = FreeLicense($LicensePool$)
; If there is an assignment of a license pool to the product, it is possible to use
; Set $result$ = FreeLicense("", $ProductId$)
;
; If there is an assignment of a license pool to a windows software id, it is possible to use
; DefVar $WindowsSoftwareId$
; $WindowsSoftwareId$ = "..."
; set $result$ = FreeLicense("", "", $WindowsSoftwareId$)
```

7.1.10 Interactive Creation and Testing of a opsi-winst Script

It is possible to interactively adapt and test your own opsi-winst script using winst32.exe.

Start by creating a directory where you will build and test your script (e.g. `c:\test`), and then copy the template scripts from the opsi-template (`setup.ins`, `delsub.ins` und `uninstall.ins`) to this directory.

Start the opsi-winst (`winst32.exe`) program via a double mouse click. (On Windows 7 Clients, you must right-click on the mouse button and select "run as Administrator"). If the opsi-client-agent is installed on your computer you will find the opsi-winst at the directory `C:\program files\opsi.org\opsi-client-agent\opsi-winst`. If the {opsi-client} agent is not installed you will find the {opsi-winst} at the share '\\<opsiserver\opsi_depot_rw' in the directory `install\opsi-winst\files`.

After starting opsi-winst, you will see the following window:

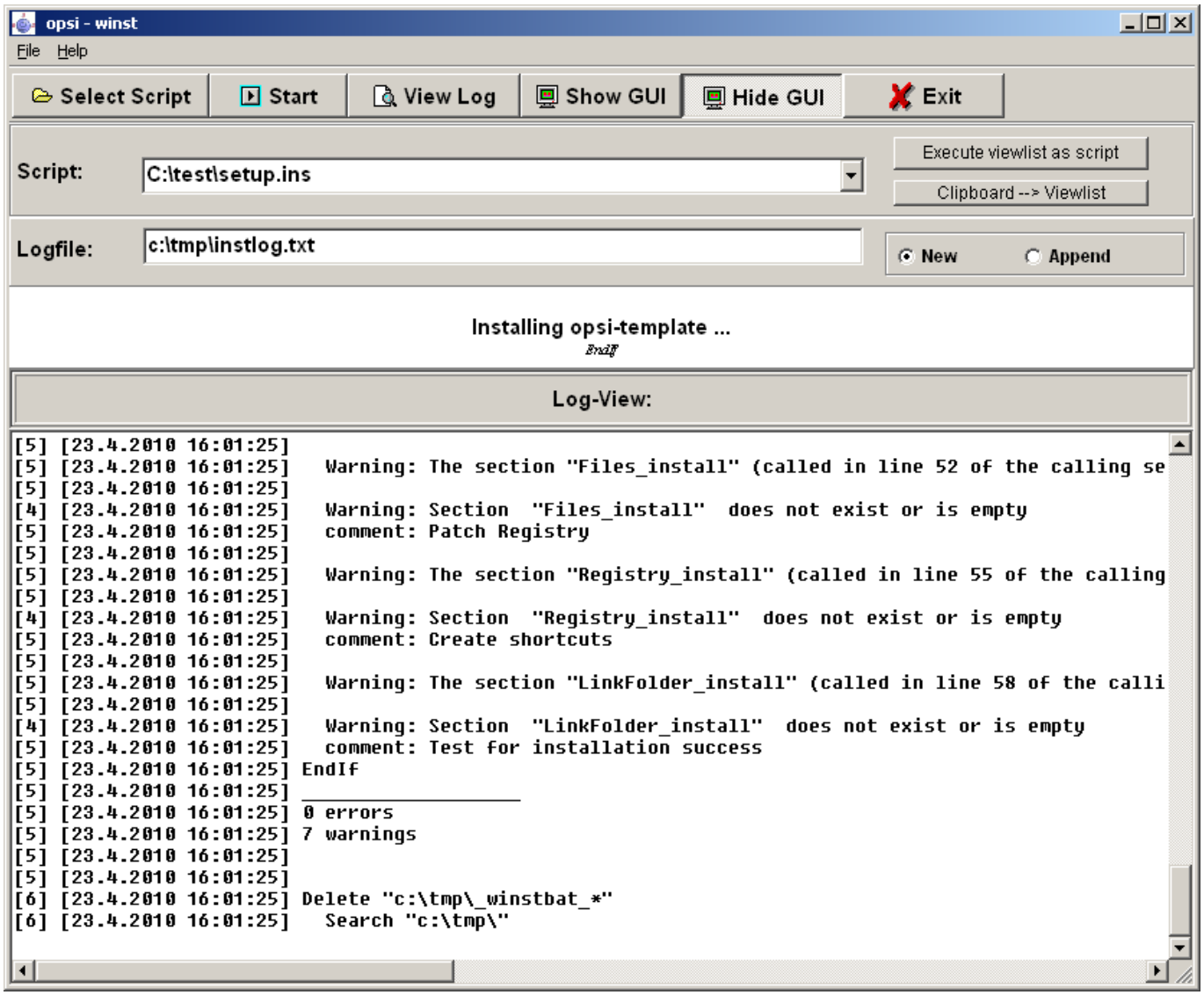

Figure 7.3: opsi-winst Started in Interactive Mode

- *Select Script* is used to choose the script that you want to execute.
- *Start* will start the execution of the selected script.
- *View Log* is used to read the log file from the script that was run most recently.

Select the *setup.ins* script and run it.

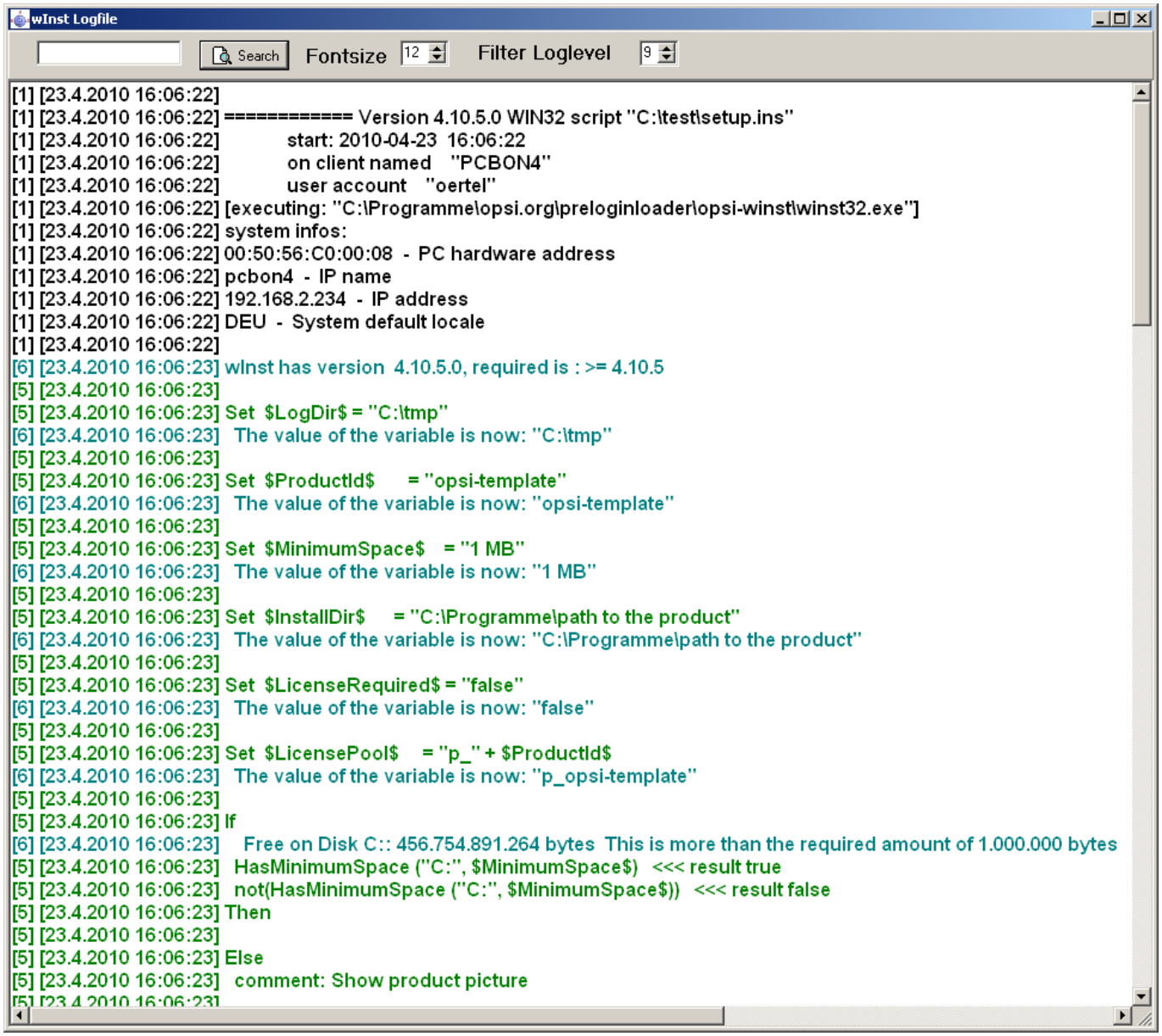

Figure 7.4: opsi-winst log view window

- Look at the log file to see how opsi-winst interpreted the script.
- After figuring out which setup.exe that you will use to install software, copy setup.exe to the directory where the scripts are located (e.g. `c:\test`).
- Open the `setup.ins` script with a editor. You may use any text editor you like. We suggest the *jEdit* with syntax highlighting for opsi-winst which is part of the essential opsi-products.

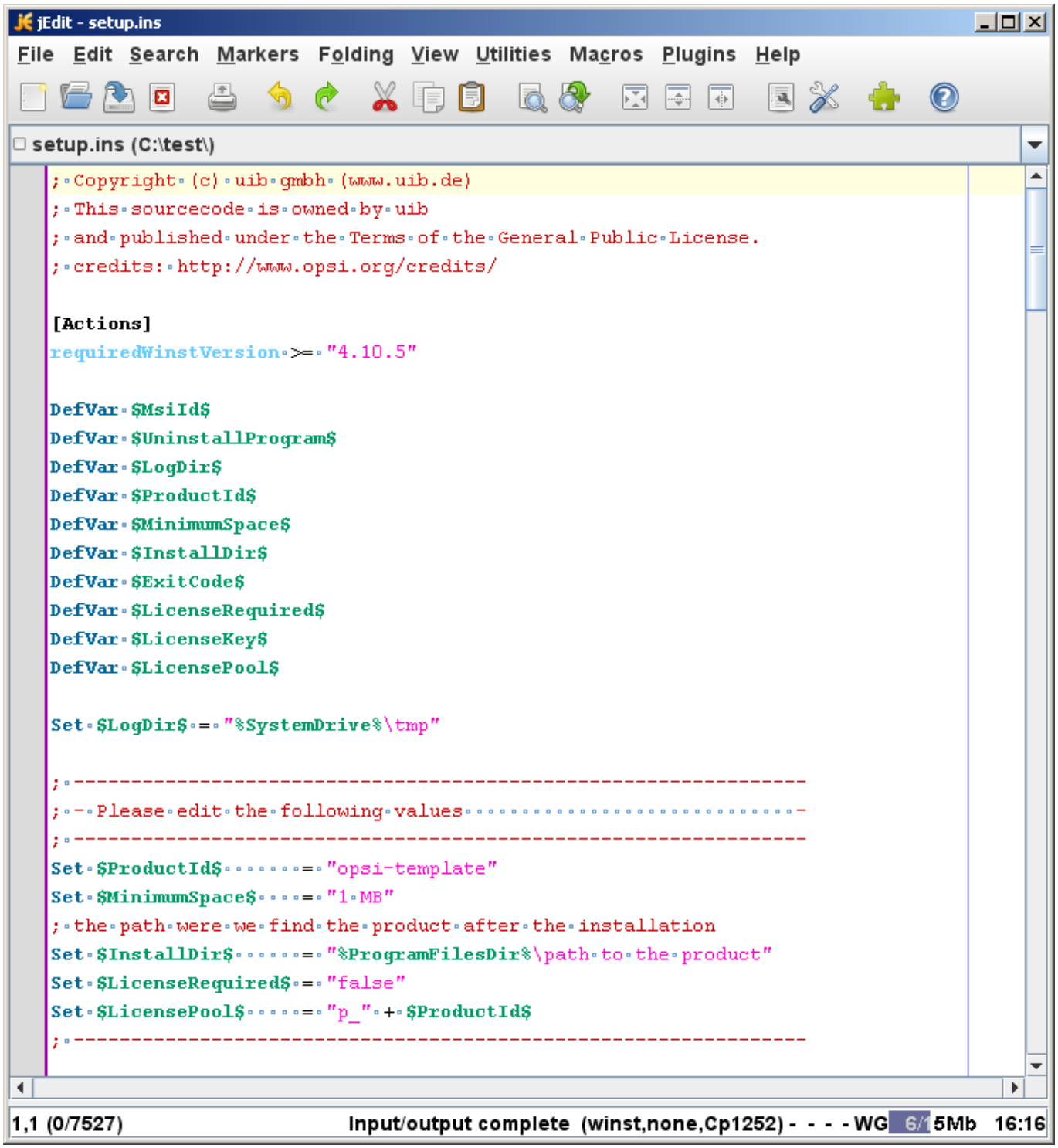

Figure 7.5: jEdit with a opsi script

- You may now change the script using the editor. Save the changes (keep the editor open).
- Now switch to the opsi-winst and start the script again. (You don't have to reselect the script. Just press the *start* button).
- Just have a look at the log again and see how the program flow changed according to your script changes.

- You can interactively develop a script until it fits your needs by performing these steps in this order:
  - Change the script and save
  - run the script
  - review the log

The next chapter contains some hints about handle any problems that may arise while building a opsi-winst script. Section 7.2.1.1 describes how to create an opsi-product from your scripts, and how to install the products on the opsi-server.

7.1.11 Suggestions on How to Solve Problems with opsi-winst Scripts

7.1.11.1 Search for Unattend or Silent Switches

For an unattended or silent setup, the original setup will be switched to an unattended non-interactive mode using the proper command line arguments.

The problem is to find the correct arguments

Look on the internet: Before you start integrating a new package, you'd better first have a look online to see if somebody has already done that job for you:

Ready to run opsi-winst scripts, built by the community, can be found at the opsi wiki.

A collection of links to web sites with switch collections can be found at opsi wiki: Software integration web-links.

Search the software producer's site: Many software manufacturers are aware of the needs of unattended software distribution, so there are often some hints and instructions in the product documentation or on the software producer's website.

Identify the manufacturer of the setup program: Most setup programs are built using frameworks like *Inno*, *NSIS*, *Installshield* or *Wise*. Each one of these setup frameworks has their own switch. The following method can be used to determine the framework and other necessary information: The input strings can be determined using the command line program *strings* given the setup program *setup.exe*, and the output framework names can be found using *grep* or *findstr*.

The Linux commands looks like this (change <mysetup.exe> to the name of your setup.exe):

```
strings <mysetup.exe> | grep -i -E "(inno|nsis|installshield|wise)"
```

Windows does not have a native `strings` command, so you will have to install it. You can download a `strings.exe` program from here: http://technet.microsoft.com/en-us/sysinternals/bb897439

To use this program, enter these commands at the command line interface (change <mysetup.exe> to the name of your setup.exe):

```
strings.exe <mysetup.exe> | findstr /i /r "inno installshield nsis wise"
```

The same method is used in the `opsi-setup-detector`. See the example below:

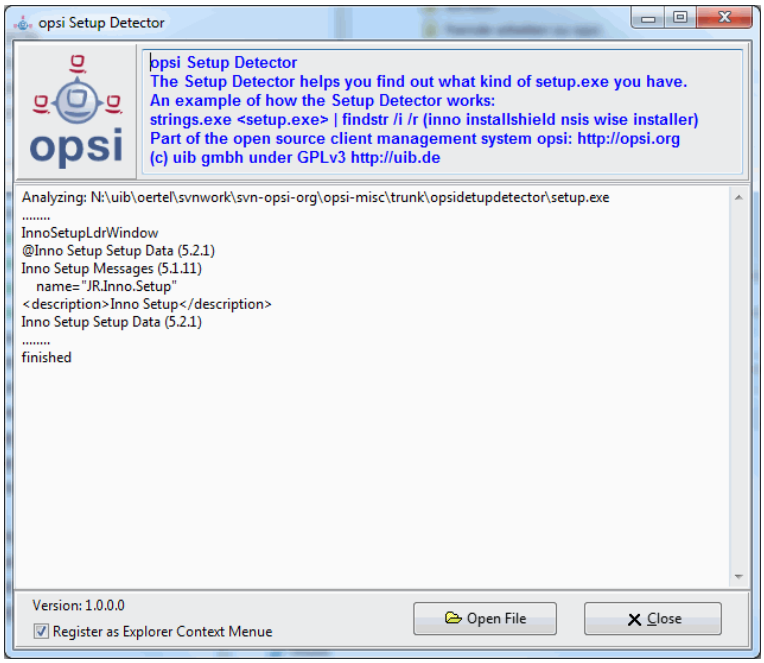

Figure 7.6: opsi setup detector

This GUI program can be called from the Windows context menu Explore.

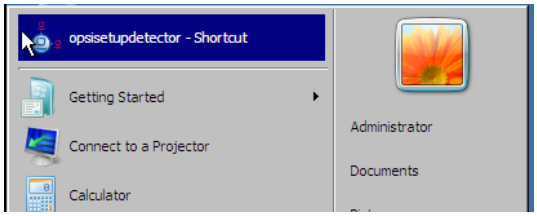

Figure 7.7: opsi setup detector in Windows Explore context menu

The *opsi setup detector* is part of the Windows package repositories and can be obtained through them.

A collection of links to web sites with switch collections can be found at opsi wiki: Software integration web-links.

7.1.11.2 Some Important opsi-winst Commands

A short overview of the opsi-winst commands can be found in the opsi-script reference card.

All syntax details are described in the opsi-script manual.

Here are some hints regarding important methods:

Stringlisten String lists can be powerful tools to review the output from other programs. Read the opsi-winst manual for details.

ExitWindows

- `ExitWindows /Reboot`
  Reboot after the script is finished
- `ExitWindows /ImmediateReboot`
  Reboot now

- `ExitWindows /ImmediateLogout` Exit the opsi-winst now

Product Properties For some products it is important to know which product properties can modify the installation in order to make a client-specific installation. Creating these properties is described below in "Creating an opsi package".

To evaluate these properties, opsi-winst provides the function `GetProductProperty`

```
if GetProductProperty("example-property", "no") = "yes"
        Files_copy_extra_files
endif
```

7.1.11.3 Installation When the User is Logged on

Before we begin, we assume that you have tried an unattended installation using an opsi-winst script, and the installation worked OK when the user had administrative privileges. However with some software products, you will see that the installation fails when started from within the opsi deployment software (opsi-client-agent). A possible reason for that difference might be that the installation process requires knowledge about the user environment or profile.

In the case of a MSI package, the option *ALLUSERS=1* might help. Example:

```
[Actions]
DefVar $MsiLogFile$
Set $MsiLogFile$ = %opsiLogDir% + "\myproduct.log"
winbatch_install_myproduct

[winbatch_install_myproduct]
msiexec /qb-! /l* $MsiLogFile$ /i "%ScriptPath%\files\myproduct.msi" ALLUSERS=1
```

Another possibility is that the installation starts a second process and stops before the second process is finished. So from the point of view of the opsi-winst script, the task is finished while in fact the second process is still working (installing / uninstalling).
In this case, you may use the modifier `/WaitSeconds <seconds>` , or `/WaitForProcessEnding "program.exe" /TimeOutSeconds "<seconds>"`, in the WinBatch section so that the script waits for the end of the second process.

Another more complex way to solve the problem is to create a temporary administrative user account and use this account for the program installation. For a detailed description on how to do this, please refer to the opsi-winst manual chapter 8.3 *Script for installation in the context of a local administrator* and use the template *opsi-template-with-admin*.

7.1.11.4 Working with MSI-packages

With Windows 2000, Microsoft launched its own installation concept based on the Microsoft Installer Service "MSI". Since then, many setup programs have become MSI compliant.

To be MSI compliant means to provide a package with installation instructions for the MSI. Usually this is a file named *product.msi*.

In practice, the setup.exe of a product contains a *product.msi* file and an additional control program for the installation. The control program unpacks the *product.msi* and pops up a window that asks if it is allowed to start the installation. If installation has been approved, then the control program checks whether or not MSI is installed, and if so passes *product.msi* to MSI. If no MSI is found, then the control program tries to install MSI.

If you were to interrupt the installation at that point, you will often find the unpacked MSI-package in a temporary directory.

For example, this package can be used for an unattended installation with the statement:

```
msiexec /i "%ScriptPath%\Product.msi" /qb-! ALLUSERS=1 REBOOT=ReallySuppress
```

7.1.11.5 Customization after a silent/unattended Installation

After a successful silent installation, some customizing might be useful. The opsi-winst is a powerful tool to do that job. First, find out what patches have to be applied. For example, that could mean analyzing which registry settings are affected by the GUI customizing tools.

You can use the tools shown in Section 7.1.11.7.

Some other often used tools are:

- sysinternals
- regshort

7.1.11.6 Integration with Automated Answers for the setup Program

Another fast way of integration is to provide an automated answer file for the setup process. The answer file contains pre-defined answers. To be more precise, the answer file is used by a control tool, which waits for the setup to come up with the interactive windows. The control tool then passes input to these windows as defined in the answer file. As a control tool we recommend *AutoIt*. The AutoIt program and the documentation can be found at: http://www.hiddensoft.com/autoit3.

AutoIt provides a lot of commands to control the setup process. Also, several error states can be handled (if known in advance) with the *[ADLIB]* section in the script.

There is, however, a fundamental challenge in using AutoIt:
The AutoIt script must provide input for every window that might pop up during installation. So if any unexpected window pops up, which isn't handled in the [ADLIB] section, AutoIt provides no input for this window and the installation stops at that point while waiting for input. This input could be done interactively by a user, and then the script can take over again and handle the next windows.

Another situation that may cause failure of an AutoIt installation:
The user can interfere with the installation if the mouse and keyboard are not disabled. Therefore we regard *unattended* or *silent* setup as a more stable solution.

A combination of both might do a good job:
The *silent*-setup does the main installation and the AutoIt script handles special conditions that might occur.

If you use the opsi option of running the installation on another desktop than the current desktop, or if the current desktop is locked, then you will find that some autoit functions do not work properly under these conditions.

Therefore you should avoid using the following autoit commands in *opsi-winst* scripts:

- winwait()
- winactivate()
- Send()

Because these commands are so widely used, we provide substitutes: **winwait()**
should be replaced by the function
`opsiwinwait($title, $text, $maxseconds, $logname)`
which is defined as:

```
Func opsiwinwait($title, $text, $maxseconds, $logname)
        Local $exists = 0
        Local $seconds = 0
        Local $mylog
        $mylog = FileOpen($logname, 1)
        While ($seconds <= $maxseconds) and ($exists = 0)
                $exists = WinExists($title , $text)
                FileWriteLine($mylog,"win: "  & $title & " ; " & $text & " exists result (1=exists): " & $exists )
                $seconds = $seconds + 1
```

```
                sleep(1000)
        WEnd
        FileClose($mylog)
EndFunc
```

The parameters are:

- `$title` the title of the window
- `$text` a part of the readable text in the window
- `$maxseconds` the timeout in seconds
- `$logname` the name of the log file

Send()
should be replaced by the function
`opsiControlClick($title, $text, $id, $maxseconds, $logname)`
respectively by
`opsiControlSetText($title, $text, $id,$sendtext, $maxseconds, $logname)`
which are defined as:

```
Func opsiControlClick($title, $text, $id, $maxseconds, $logname)
        Local $result = 0
        Local $seconds = 0
        Local $mylog
        $mylog = FileOpen($logname, 1)
        While ($seconds <= $maxseconds) and ($result = 0)
                $result = ControlClick($title , $text,$id)
                FileWriteLine($mylog,"answer for " & $title & " ; " & $text & " id: " & $id & " sended: result (1=\
    success) : " & $result)
                $seconds = $seconds + 1
                sleep(500)
        WEnd
        FileClose($mylog)
EndFunc

Func opsiControlSetText($title, $text, $id,$sendtext, $maxseconds, $logname)
        Local $result = 0
        Local $seconds = 0
        Local $mylog
        $mylog = FileOpen($logname, 1)
        While ($seconds <= $maxseconds) and ($result = 0)
                $result = ControlSetText ($title , $text,$id, $sendtext)
                FileWriteLine($mylog,"answer for " & $title & " ; " & $text & " id: " & $id & " set: " & $sendtext & " \
    sended: result (1=success) : " & $result)
                $seconds = $seconds + 1
                sleep(500)
        WEnd
        FileClose($mylog)
EndFunc
```

The parameters are:

- `$title` the title of the window
- `$text` a part of the readable text in the window
- `$id` the numerical ControlId of the button or edit field
- `$sendtext` the text to insert to a edit field
- `$maxseconds` the timeout in seconds
- `$logname` the name of the log file

Therefore, you should use the program `Au3info.exe` to get the *ControlId* needed by these commands. Please use the numerical *ControlId*, because the other variants do not seem to work properly:

Below is an example from a script.
In this script we produce a log file from the autoit activities, which may be integrated in the *opsi-winst* log file with the following commands:

```
includelog %opsiLogDir% + "\au3.log" "500"
```

Example:

```
[ExecWith_autoit_confirm]
Func opsiwinwait($title, $text, $maxseconds, $logname)
        Local $exists = 0
        Local $seconds = 0
        Local $mylog
        $mylog = FileOpen($logname, 1)
        While ($seconds <= $maxseconds) and ($exists = 0)
                $exists = WinExists($title , $text)
                FileWriteLine($mylog,"win: "  & $title & " ; " & $text & " exists result (1=exists): " & $exists )
                $seconds = $seconds + 1
                sleep(1000)
        WEnd
        FileClose($mylog)
EndFunc

Func opsiControlClick($title, $text, $id, $maxseconds, $logname)
        Local $result = 0
        Local $seconds = 0
        Local $mylog
        $mylog = FileOpen($logname, 1)
        While ($seconds <= $maxseconds) and ($result = 0)
                $result = ControlClick($title packet
        FileClose($mylog)
EndFunc

Func opsiControlSetText($title, $text, $id,$sendtext, $maxseconds, $logname)
        Local $result = 0
        Local $seconds = 0
        Local $mylog
        $mylog = FileOpen($logname, 1)
        While ($seconds <= $maxseconds) and ($result = 0)
                $result = ControlSetText ($title , $text,$id, $sendtext)
                FileWriteLine($mylog,"answer for " & $title & " ; " & $text & " id: " & $id & " set: " & $sendtext & " \
    sended: result (1=success) : " & $result)
                $seconds = $seconds + 1
                sleep(500)
        WEnd
        FileClose($mylog)
EndFunc

; exact title match
Opt("WinTitleMatchMode", 3)
$mylog = FileOpen("%opsiLogDir%\au3.log", 2)
FileWriteLine($mylog,"auto-it started - waiting for the window")
FileClose($mylog)

opsiwinwait("InstallShield Wizard" , "Wollen Sie wirklich", 200, "%opsiLogDir%\au3.log")
        opsiControlClick("InstallShield Wizard" , "Wollen Sie wirklich", 6, 5, "%opsiLogDir%\au3.log")
opsiwinwait("InstallShield Wizard" , "Deinstallation ist abgeschlossen", 400, "%opsiLogDir%\au3.log")
        opsiControlClick("InstallShield Wizard" , "Deinstallation ist abgeschlossen", 1, 5, "%opsiLogDir%\au3.log")

Sleep(500)
;and good bye
Exit
```

see also:
http://www.autoitscript.com/wiki/FAQ#Why_doesn.27t_my_script_work_on_a_locked_workstation.3F

http://www.autoitscript.com/autoit3/docs/
http://www.autoitscript.com/autoit3/docs/intro/controls.htm
http://www.autoitscript.com/autoit3/docs/functions.htm

7.1.11.7 Analyze and Repackage

When a software developer builds a setup for deployment, the developer usually knows about the required components of the software that have to be installed. But if somebody has a black box as a setup, then they need first to analyze what the setup does. This can be done by monitoring the setup activities with the appropriate tools (e.g. monitoring files and registry access) or by comparing the system states before and after installation.

To analyze the before or after states, there are several tools. For Example:

- InstallWatch Pro
- appdeploy-repackager

7.1.11.8 How to uninstall Products

To uninstall a software product from a computer, you need an *uninstall* script to perform the deletion. The fundamental difficulty in software deletion is deciding what exactly has to be removed. Not all of the files that came with a software package can be deleted afterwards. Sometimes a package comes with standard modules that are also referred to by other programs. Often only the software manufacturer himself knows what parts have to be removed. The manufacturer's setup might offer an unattended uninstall option which can be embedded in the opsi uninstall script. Otherwise opsi-winst provides several commands for software deletion:

Using an uninstall routine If the product manufacturer provides an option for software deletion, you must checked whether or not it can be run unattended (or in silent mode). If it requires some user interaction, an AutoIt script combined with the uninstall routine might do the job. The uninstall statement can be embedded in a [WinBatch] section of the opsi-winst script:

```
[WinBatch_start_ThunderbirdUninstall]
"%SystemRoot%\UninstallThunderbird.exe" /ma
```

When using an uninstall program, always run a test to see if all of the files have been deleted and the computer is still in a stable state.

Products that are installed by MSI normally come with an uninstall option, which is usually the program `msiexec.exe` combined with the parameter `/x`. The parameter `/qb-!` is for the unattended mode (or without user interaction). So here is an example of an unattended uninstall command:

```
msiexec.exe /x some.msi /qb-! REBOOT=ReallySuppress
```

Instead of the package name, you could also use the GUID (Global Unique ID) with `msiexec.exe`. This GUID identifies the product in the system, which can be found in the registry directory *HKLM\Software\Microsoft\Windows\CurrentVersion\Uninstall*

A request using the GUID looks like this:

```
msiexec.exe /x {003C5074-EB37-4A75-AC4B-F5394E08B4DD} /qb-!
```

If none of these methods are available or sufficient, the uninstall can be done using a opsi-winst script as described below:

Useful opsi-winst commands for uninstall If a product has been installed by opsi-winst functions, or if there is no uninstall routine for the product, the complete uninstall has to be done by a opsi-winst script. opsi-winst comes with some powerful uninstall functions. This chapter provides a brief overview of the uninstall functions, and more detailed information can be found in the opsi-winst handbook.

Basic uninstall means deleting one or more files from the file system. This command can be executed from a opsi-winst files section:

```
delete -f <file name>
```

or to delete a directory including sub directories:

```
delete -sf <dir name>\
```

The parameter *f* means *force* or to delete the files even if they are marked as *read only* and the parameter *s* means including the *subdirectories*. A file or directory can be deleted from all user profiles using the option */AllNTUserProfiles* (see opsi-winst manual for details).

Directories containing files with the attribute *hidden* or *system* can be deleted by using a *DosInAnIcon*-section:

```
[DosInAnIcon_deleteDir]
rmdir /S /Q "<List>"
```

To stop a running process before deletion use the 'killtask` command with the process' name (look at the task manager for process name):

```
KillTask "thunderbird.exe"
```

If the product or part of it, runs as a service, you will have to stop the service before deleting the files. One way to do so, is to set the service state to inactive in the registry and restart the computer. Or to stop the service by using the command *net stop*, which doesn't need a reboot:

```
net stop <servicename>
```

Deleting DLL files also requires special attention, since DLLs could also be used by other products. There is no general way of handling this.

To delete registry entries with opsi-winst you can use the command DeleteVar. This command deletes entries from the currently open key:

```
DeleteVar <VarName>
```

To delete a registry key with all sub keys and registry variables, you can use the opsi-winst command DeleteKey:

```
DeleteKey [HKLM\Software\Macromedia]
```

7.1.11.9 Known Issues with the 64 Bit Support

The opsi installer opsi-winst is a 32 bit program. There are no known problems when installing 32 bit software on a 64 bit system using opsi-winst. For the installation of 64 bit software, some constants (like *%ProgramFilesDir%*) give wrong values.

New versions of opsi-winst have special commands to handle these problems. So read the opsi-script manual for these issues.

7.2 Creating an opsi Package

In opsi, the new software is integrated into the system as a package. This package contains the installation files, the opsi-winst installation script, and any meta data.

The advantages of this format are essentially:

- Simplified menu driven handling using the program `opsi-newprod`.
- Holding all meta data in one file, which is easy to edit.

- Optional menu driven installation of the package, with optional default overriding.
- Information about the package will be saved; including product version, package version, and customer extensions. The package information is stored in the installation directory, and all the information can be seen in the package name and the opsi-configeditor. This means that different package versions can be easily handled (product life cycle management).
- For creating and unpacking products, no root privileges are required. Privileges of the group *pcpatch* are sufficient.

The package itself is merely a Gzip compressed cpio archive. This archive includes three directories:

- `CLIENT_DATA`
  holds the files which are to be copied into the product directory (`/var/lib/opsi/depot/<productid>`).
- `OPSI`
  The file named `control` holds the product meta data (like the product dependencies). The files `preinst` and `postinst` will be executed before and after the installation. Any customer extensions might be added here.

7.2.1 Create, Pack, and Unpack a New Product

In order to create a new opsi package, you must login to the server and do some things at the command line. To be able to do this from windows you may use putty.exe: (http://www.chiark.greenend.org.uk/~sgtatham/putty/-download.html).

The essential commands to create and install packages are:

- `opsi-newprod`
- `opsi-makeproductfile`
- `opsi-package-manager -i <{opsi-product}-file>`

The privileges of the group *pcpatch* are required to create a new product.

Opsi makes use of parallel compression provided by `pigz` if installed. This requires a minimum version 2.2.3 or any higher version. If a sufficient version is installed, opsi will automatically use it for (de-)compression of products. Please keep in mind that archives created by `gzip` or `pigz` can profit from the bandwidth preserving synchronization via `rsync` but they are not bit-compatible. This will become relevant if you have been using `gzip` before to create your packages and synchronized these packages to other depots. If you now use `pigz` for compression an sync, it will transmit more than the expected differences. This is the case for the first synchronization after a switch of the used compression program. Any further synchronization will then again only transmit the differences. It is possible to explicitly disable the usage of `pigz` on your server by setting the value for `use_pigz` under the section `packages` in the file `/etc/opsi/opsi.conf` to `False` as shown below:

```
[packages]
use_pigz = False
```

You should create products in the directory `/home/opsiproducts`. This directory is also available as share on *opsi_workbench*. The group *pcpatch* has to be owner of the directory and the directory permissions are 2770 (*set group ID* bit is set for group pcpatch).

Caution
On distributions from the SUSE family this directory is located at `/var/lib/opsi/workbench`.

7.2.1.1 Create with opsi-newprod

Warning
Do not use any country-specific symbols (umlaut), since the actual country code might vary for different code tables.

To start creating a new product, change directories to the product directory, and start the creation of the new product by entering the command `opsi-newprod`. The next question will ask you about the type of product you want to create. Choose the type *localboot* for products which should be installable by *opsi-client-agent*/*opsi-winst*. The product type *netboot* is used for products which are activated as a bootimage (like OS installation)

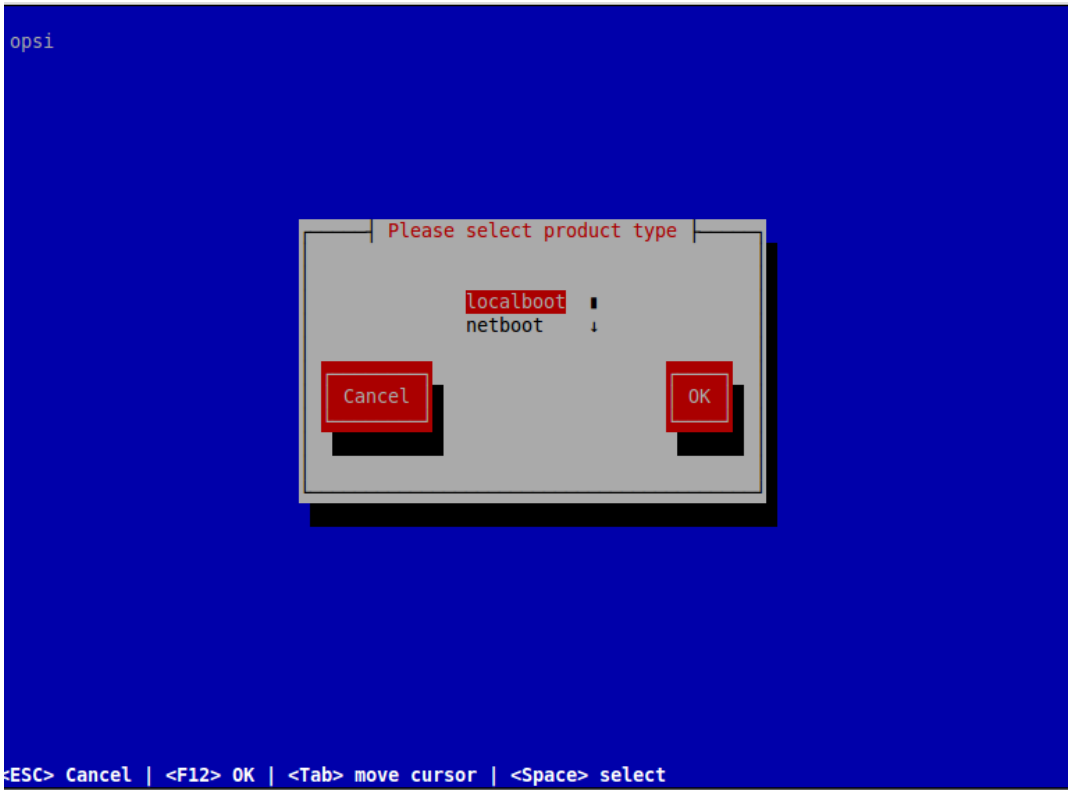

Figure 7.8: Choose the product type: localboot

Confirm your choice with tab (or F12). Next, fill in the basic product parameters. At the top of the window there is an explanation for the current input field.

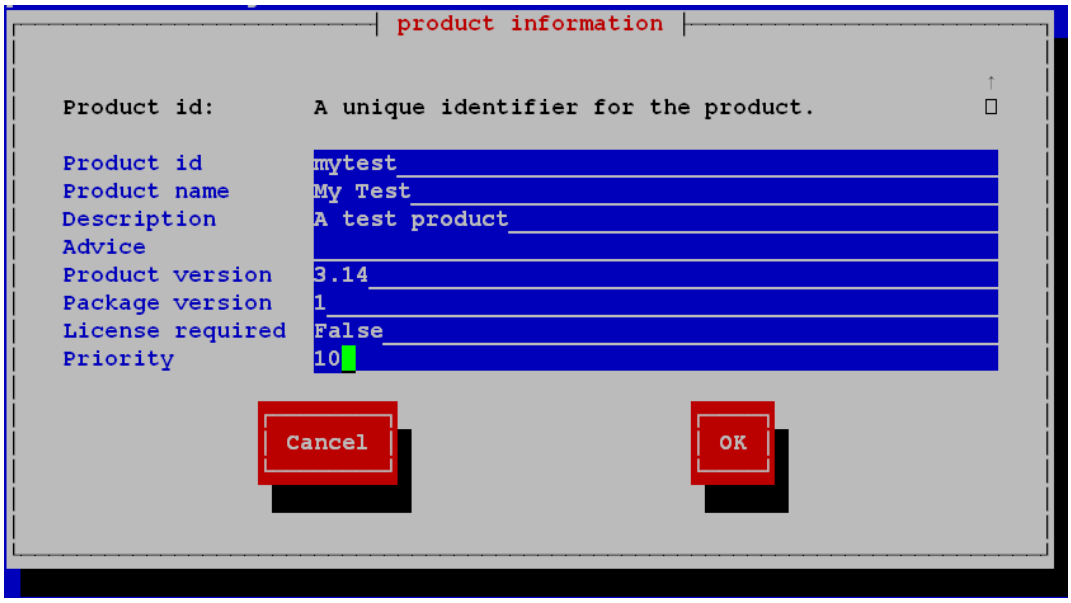

Figure 7.9: Input of the product information

Product Id
is a distinct short name for the product, independent from the product version (we recommend to use only plain ASCII letters and -, no white space, no special characters)

Product name
is the full name of the product

Description
is an additional description of the product.

Advice
is some additional information on how to handle the product (a note).

Product version
is the version of the packed software (max 32 chars).

Package Version
is the version of the package for the product version. For example, this helps to distinguish between packages with the same product version but with modified *opsi-winst* scripts.

License required
is only relevant to netboot products.

Priority
controls the installation sequence. Possible Values are between 100 (at the very beginning) and -100 (at the end). Note: product dependencies also have influence on the installation sequence. See the opsi manual for more information.

After the product information is completed, fill in which action scripts should be provided:

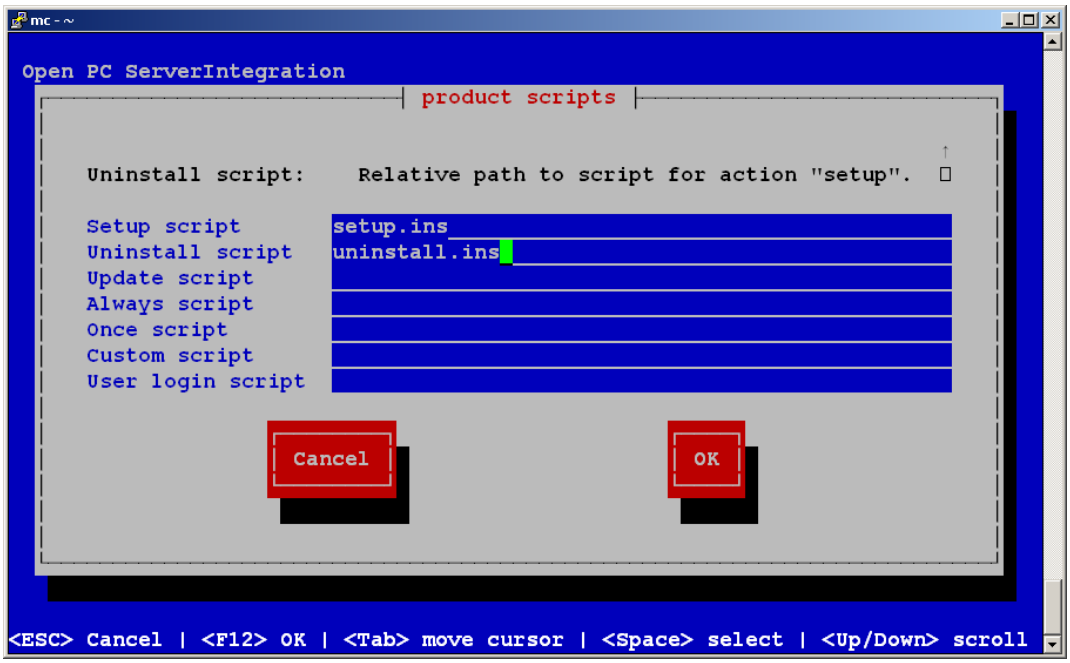

Figure 7.10: Input of the opsi-winst script names for different actions

After editing the product information you should mention the script you want to use for different activities.

Usually the **Setup script** is named **setup.ins**

Usually the **Uninstall script** is named **uninstall.ins**

An **Update-Script** will be used for minor changes on existing big installations. If this product is switched to the required action *setup*, then the update script will be automatically executed after the setup script.

An **Always-Script** will be executed at the beginning of every activity of *opsi-client-agent* (e.g. on every boot).

A **Once-Script** has the resulting state **not_installed**. It is a very special kind of script, and you should only use it if you really know what you are doing.

A **Custom-Script** doesn't change the resulting state. It is a very special kind of script, and you should only use it if you really know what you are doing.

A **userLoginScript** is used to modify the user's profile after the user logs into the system. It only works with the opsi extension *User Profile Management*, which is described at the *User Profile Management* chapter in the opsi-manual.

Type	resulting state	resulting action
setup	installed	none
uninstall	not_installed	none
update	installed	none
always	installed	always
once	not_installed	none
custom	*unchanged*	*unchangend*
User login	*unchanged*	*unchanged*

The next step is to define one or more product dependencies. If there are no product dependencies, select *No*.

Figure 7.11: Create product dependency: No/Yes

To create a product dependency, enter the following data (help is available at the top of the window):

Dependency for Action
Which product action shall the dependency create, or when should the dependency be checked (setup, uninstall ...).

Required product id
Product id of the required product.

Required action
Select the required action (if any) for the required product. Actions can be as *setup*, *uninstall*, *update*. If no *required action* is set, a *required installation status* must be set

Required installation status
Select the required status of the required product (if any). Usually this is *installed*. So the required product will be installed if it isn't installed on the client yet. If no *required installation status* is set, a *required action* must be set

Requirement type
This is regarding the installation order. If the required product has to be installed before the installation of the actual product, this is set to *before*. If it has to be installed after the actual product, set *requirement type* to *after*. Leave it blank if the installation order doesn't matter.

Note
The possibility to define uninstall actions or dependencies is broken. After defining a product dependency, you will be asked if you want to create another product dependency. If you choose *Yes*, then the procedure for defining a product dependency is repeated. If you choose *No*, then you will be asked to define some product properties, which means defining additional switches for product customization.

Note
The installation sequence results from a combination of product dependencies and product priorities. For details on how this is done, and what you can configure, see the opsi-manual.

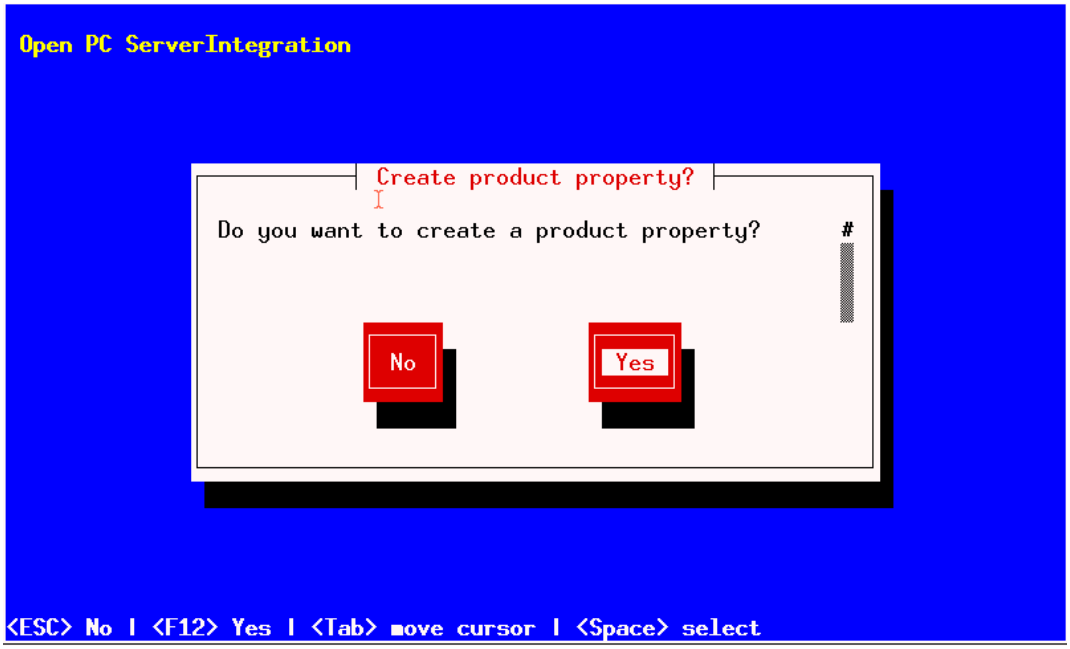

Figure 7.12: A(nother) product property to create?

If you answer *Yes*, you will have to describe the product properties.

The product properties are client specific, and have names (keys) which can hold different values. These values can be evaluated by the *opsi-winst* script, and result in installing different options at installation time.

First we have to decide if our property is a text value (*unicode*) or a logical value e.g. true/false (*boolean*). If you are not sure choose *unicode*.

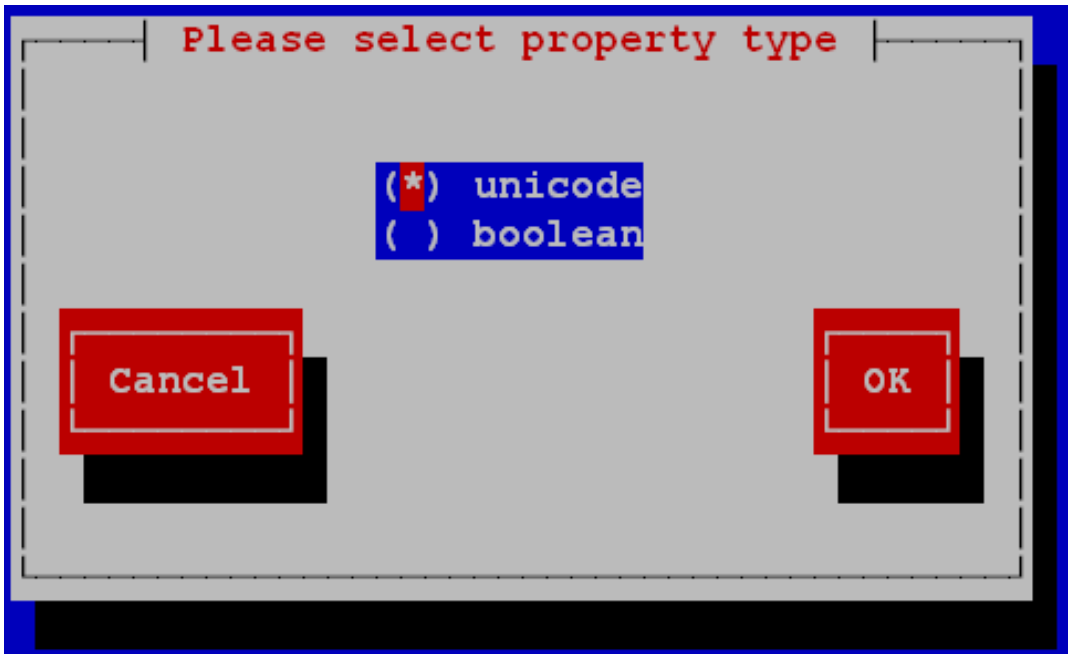

Figure 7.13: Choose the data type of the property

Next, a description for the switch needs to be specified. This description will be shown in the opsi-configed as a help

text. Next, you can define the set of values for the switch (separated by comma). If this is left blank, then any value is allowed for the switch.

Note
If a values contains a backslash \ it has to be doubled.
An example showing how a path would be defined: `C:\\temp`

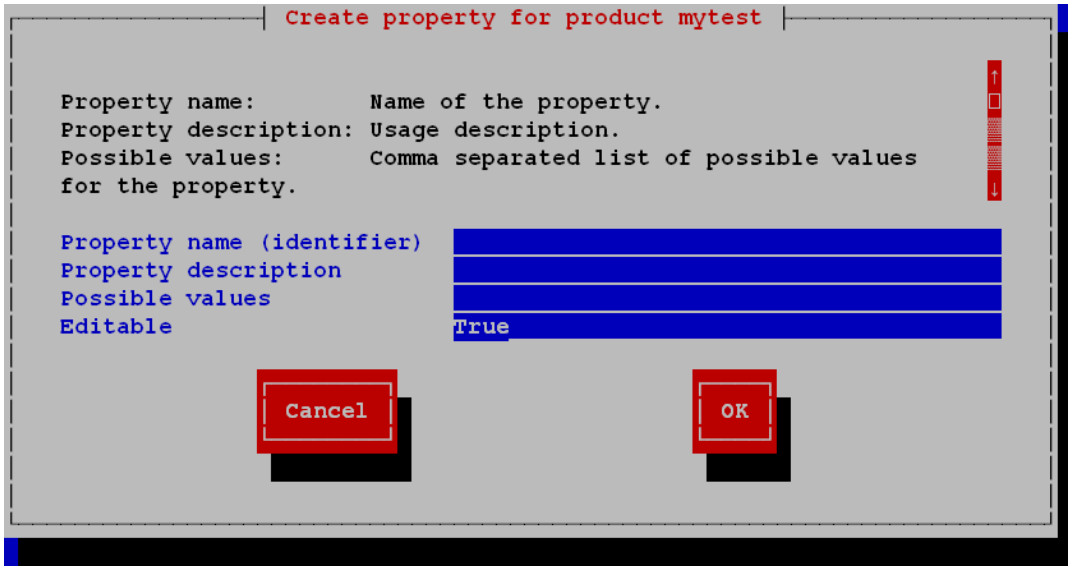

Figure 7.14: Description of the product properties

Next, you can decide if the product property has a default value (switch).

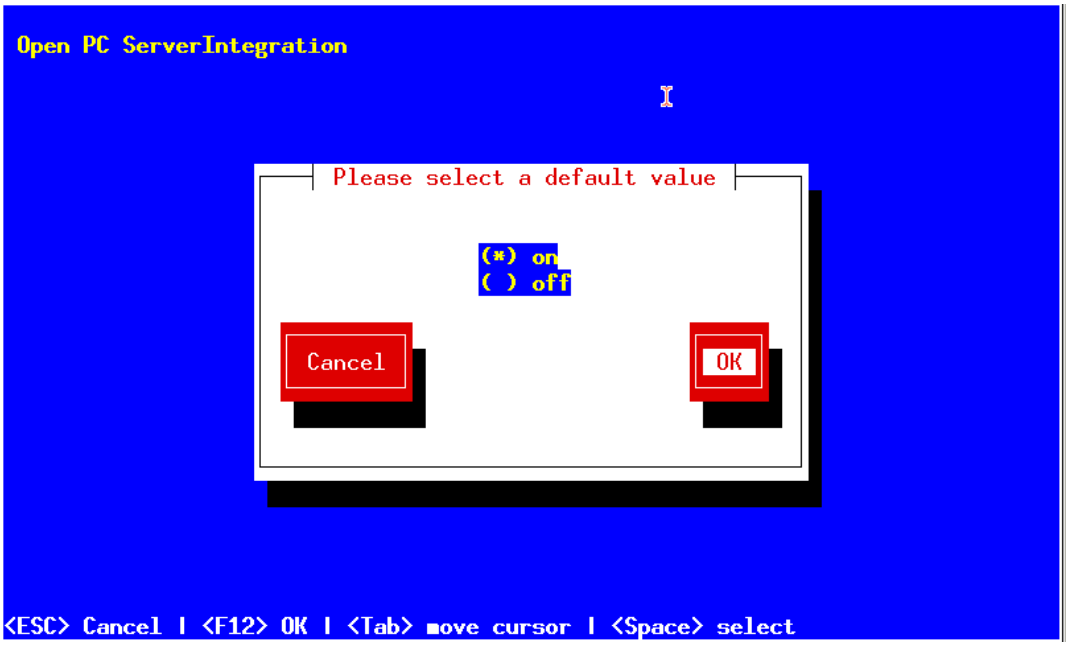

Figure 7.15: Default value of the product property

If you choose *boolean* as the data type, then the description will contain only the *Property name* and *Property description*.

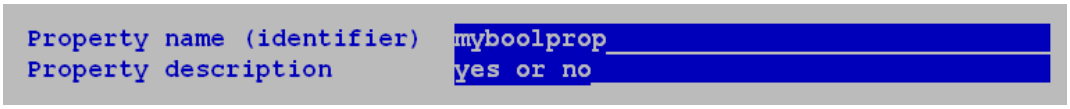

Figure 7.16: Description of a boolean property

After defining a product property, you will be asked if you want to create another product property. If you choose *Yes*, then the procedure of defining a property will be repeated. If you choose *No*, then you will be asked for name and email of the product maintainer. This data will be written on the changelog.

Figure 7.17: Input of the maintainer data

Finally, the basic definitions for the new product are done.

Using the list command (`ls`), you can see the directory structure as described above. Change to the `OPSI` folder and list the content. The `control` file now contains the data you just defined, and you can load the file into an editor to view or change the entries.

Example of a control file:

```
[Package]
version: 1
depends:
incremental: False

[Product]
type: localboot
id: mytest
name: My Test
description: A test product
advice:
version: 3.14
priority: 10
licenseRequired: False
productClasses:
setupScript: setup.ins
uninstallScript:
updateScript:
alwaysScript:
onceScript:
customScript:
userLoginScript:

[ProductDependency]
action: setup
requiredProduct: javavm
requiredStatus: installed

[ProductProperty]
type: unicode
name: mytextprop
multivalue: False
editable: True
description: hint
```

```
values: ["off", "on"]
default: ["off"]

[ProductProperty]
type: bool
name: myboolprop
description: yes or no
default: False

[Changelog]
mytest (3.14-1) testing; urgency=low

  * Initial package

 -- jane doe <j.doe@opsi.org>  Mi, 14 Jul 2010 12:47:53 +0000
```

For the next step, you will have to copy the product *opsi-winst* script, and any necessary data files (i.e. program-installation-executable.exe), into the `CLIENT_DATA` folder.

So if the script you have written is currently at `c:\test`, just mount the share *\\<opsiserver\opsi_workbench* e.g. to *w:*, and then copy the complete content of `c:\test` to the directory *CLIENT_DATA*.

7.2.1.2 Build the Package with opsi-makeproductfile

Now you may build the package. Change to the root directory of the product (maybe `/home/opsiproducts/myproduct/`, and enter *opsi-makeproductfile*. The product package will be built. The package (`<package name>`) will be a file that has a format similar to `/home/opsiproducts/<myproduct>/<myproduct_ProductVersion-PackageVersion>.opsi`.

Finally, install the package. The resulting package can be installed on the opsi-server with the command `opsi-package-manager -i <package name>`.

`opsi-makeproductfile` can be started with different options:

```
#  opsi-makeproductfile --help

Usage: opsi-makeproductfile [-h] [-v|-q] [-F format] [-l log-level] [-i|-c custom name] [-I required version] [-t temp \
    dir] [source directory]
Provides an opsi package from a package source directory.
If no source directory is supplied, the current directory will be used.
Options:
   -v          verbose
   -q          quiet
   -l          log-level 0..9
   -n          do not compress
   -F          archive format [tar|cpio], default: cpio
   -h          follow symlinks
   -I          incremental package
   -i          custom name (add custom files)
   -c          custom name (custom only)
   -C          compatibility mode (opsi 3)
   -t          temp dir
```

It is recommended to create the packages with a corresponding md5 checksum file. This file is used amongst others by opsi-product-updater to check after a file transfer to ensure package integrity. Such a file can be created with `-m`.

When transferring packages to opsi-depot-server zsync can be used to only transfer differences between different packages. To be able to use this method a special `.zsync` file is required. Such a file can be created with `-z`.

If you are running into the problem that the creation of a package fails because of insufficient free space in `/tmp` you can use the option `-t` to specify a different temporary folder.

If there is already a package file with the same version information, opsi-makeproductfile will ask for overwrite confirmation:

```
Package file '/home/opsiproducts/mytest/mytest_3.14-1.opsi' already exists.
Press <O> to overwrite, <C> to abort or <N> to specify a new version:
```

Choosing *o* will overwrite, *c* abort, and *n* will ask for new version information.

The created opsi-package can be installed at the opsi-server with the command:
`opsi-package-manager -i <packagename>`

More information about the opsi-package-manager can be found in the opsi-manual.

Chapter 8

More Information

More detailed information can be found in the opsi-manual (http://download.uib.de/opsi_stable/doc/opsi-manual-stable-en.pdf).

If you need help during your evaluation of opsi, then more help can be found at https://forum.opsi.org

For production-level installations, we offer commercial support: http://uib.de/en/support-training/support/