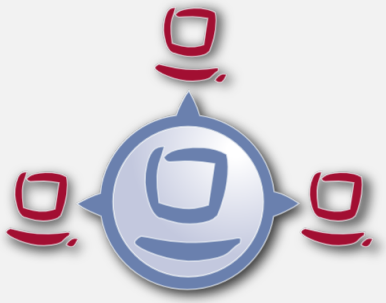

opsi Getting Started opsi-Version 4.0.7

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

Stand: 17. Dezember 2018

Inhaltsverzeichnis

Kapitel 1

Copyright

Kapitel 2

Einführung

Diese Anleitung beschreibt detailliert die Installation und Inbetriebnahme eines opsi-servers, ausgehend von den zur Verfügung gestellten Installationspaketen bis zur Testinstallation eines Clients.

Die dargestellte Netzwerkkonfiguration ist exemplarisch und bezieht sich auf ein Netz u.a. ohne konkurrierende DHCP-Server (z.B. ein isoliertes Testnetz, in das der opsi-server und seine Clients für die ersten Versuche gestellt werden können).

Wir empfehlen Ihnen dringend, erste Versuche mit opsi in einem Testnetz zu machen, das getrennt von anderen DHCP-Servern ist, welches Sie aber zeitweise an Ihr Hauptnetz ankoppeln können, um z.B. Aktualisierungen und Pakete aus dem Internet laden zu können

Für die Einbindung in bestehende Netze können Sie ggf. auf Beratungsleistungen durch uib zurückgreifen.

2.1 Die Schritte der Installation und Inbetriebnahme

Die Installation und Inbetriebnahme eines opsi-servers erfolgt in vier Schritten:

1. Grundinstallation des Servers
2. Anpassung des Servers: Konfiguration des Netzwerks, Passwortvergabe, Aktualisierung des Servers
3. Download und Installation der notwendigen opsi-Produkte für die Clients
4. Vervollständigen der Betriebssystem-Basispakete für Windows von den Original-Medien

Danach kann bereits ein Client automatisch installiert werden.
Für die Grundinstallation existieren mehrere Varianten, die je nach Interesse und Vorliebe genutzt werden können. Die Verfahrensweisen bei den Varianten der Grundinstallation sind in Abschnitt Kapitel 4 der vorliegenden Anleitung geschildert.

2.2 Hardwarevoraussetzungen

Für den opsi-server in realer Hardware wird benötigt:

- Intel-x86-kompatibler PC
- mindestens 2GB RAM
- eine Festplatte mit mindestens 60 GB Kapazität

Im Testbetrieb sind die Anforderungen an die Leistungsfähigkeit der Maschine nicht besonders hoch. Für den Produktivbetrieb kann es nötig werden, den Server an die Anforderungen anzupassen.

Bei der Verwendung der vorkonfigurierten VMware-Maschine muss ein angemessener Wirtsrechner genutzt werden. Es sollte mindestens ein Dualcore Prozessor und 4GB RAM vorhanden sein. Für eine Teststellung kann problemlos eine weitere VMware-Maschine im selben Wirtsrechner als Client dienen.

2.3 Software und Konfigurationsvoraussetzungen

Folgende Voraussetzungen müssen erfüllt sein, damit Sie opsi installieren bzw. einsetzen können:

- **gültiger DNS Domainname**
  Ihr DNS Domainname muss mindestens aus einer Domain und einer Topleveldomain bestehen. Anders ausgedrückt: der volle (Full qualified) Domainname muss mindestens einen Punkt enthalten. Weiterhin muss die Topleveldomain aus mindestens zwei Zeichen bestehen.
  Erlaubt sind z.B.: *domain.local*, *uib.de*, *subdomain.domain.de*.
  Nicht erlaubt ist z.B. *mydomain.d*, denn das wäre nur ein Zeichen in der Topleveldomain.
  Nicht erlaubt ist z.B. *mydomain*, denn das wäre nur eine Topleveldomain.
  Siehe auch:
  Wikipedia: Domain
  Wikipedia:Hostname - Richtlinien

- **gültige DNS Hostnamen**
  Die Hostnamen (auch der Clients) müssen den Richtlinien entsprechen. Dazu gehört auch, dass sie z.B. keine Unterstriche enthalten dürfen.
  Siehe auch:
  Wikipedia:Hostname - Richtlinien

- **Korrekte Namensauflösung für den Server**
  Prüfen Sie den Eintrag für den opsi-server in der Datei */etc/hosts*,
  oder aber die Ausgabe von

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

  Das Ergebnis sollte beispielsweise so aussehen:
  192.168.1.1 server.domain.tld server
  Sieht das Ergebnis nicht so aus (enthält z.B. *127.0.0.1* oder *localhost*), dann müssen Sie Ihre */etc/hosts* oder Namensauflösung zunächst korrigieren.

Anmerkung
Die Namen müssen den Vorgaben eines DNS-Systems entsprechen, aber ein DNS-Server wird für den Betrieb von opsi nicht benötigt.

Anmerkung
opsi benötigt kein *Active Directory* oder ähnliches. Eine Integration ist möglich, wird aber nicht vorausgesetzt.

Die benötigten Network Ports finden Sie hier: Abschnitt 4.8

Kapitel 3

opsi Support Matrix (opsi läuft auf welchen Servern)

Im folgenden finden Sie eine Übersicht auf welchen Plattformen opsi als Server läuft.

3.1 Unterstützte Distributionen für 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

Kapitel 4

opsi-server Installation

4.1 opsi-server-Grundinstallation

In diesem Abschnitt werden verschiedene Varianten der Realisierung des *opsi-servers* dargestellt. Wenn alle Schritte klappen, erhalten Sie ein Serversystem, das bereit für die endgültige Konfiguration und Inbetriebnahme ist. Sie haben also die Wahl und können die Abschnitte überspringen, die die nicht gewählten Wege zum opsi-server beschreiben.

Wir empfehlen zur Evaluierung von opsi, den Weg über die vorinstallierte VMware-Maschine zu wählen.
Die Grundregel ist: wenn Sie der Reihe nach die Befehle in den

```
hervorgehobenen Feldern
```

ausführen (z.B. per cut&paste aus diesem Dokument holen), dann sollte die Installation auch klappen.

Bei Problemen wenden Sie sich bitte an: https://forum.opsi.org

4.1.1 Inbetriebnahme der von uib vorkonfigurierten virtuellen Maschine

Da die Anforderungen an die Rechengeschwindigkeit eher niedrig sind, lässt sich der *opsi-server* auch problemlos als virtuelle Maschine installieren. Für ESX, VMware und Virtualbox haben wir bereits eine entsprechende Maschine eingerichtet. Die Dateien stehen im Internet zur Verfügung. Zum Betrieb genügt ein kostenfreier VMware-Player oder Virtualbox. Sie können diese Maschine aber auch in VMware-Server oder ESX betreiben.

4.1.1.1 Auspacken und erster Start

VMware

Sofern Sie bereits über einen Gastrechner verfügen, auf dem die VMware-Vollversion oder ein VMware-Player installiert ist, erledigen Sie die Grundinstallation des *opsi-servers* mit wenigen Mausklicks:

- Laden Sie die aktuelle opsi-ServerVM von hier: http://uib.de/de/opsi/opsi-testen-download/
- Entpacken Sie den Zip-File, das Verzeichnis `opsivm` wird erzeugt.
- Starten Sie den VMware-Player.
  Öffnen Sie "Open a Virtual Maschine", suchen Sie in dessen Dateiauswahldialog das Verzeichnis `opsivm` und darin die Datei `opsivm.ovf`. Gegebenenfalls müssen Sie ganz unten noch die anzuzeigenden Dateitypen auf ovf ändern. Sie können den Server jetzt unter einem eigenen Namen importieren. Der virtuelle Server kann gestartete werden.

ESXi-Server

- Laden Sie die aktuelle opsi-ServerVM von hier: http://uib.de/de/opsi/opsi-testen-download/
- Entpacken Sie den Zip-File, das Verzeichnis `opsivm` wird erzeugt
- Starten Sie den vSphere Client.
  Erstellen Sie sich einen neuen Rechner über *Datei* / *OVF-Vorlage bereitstellen...* und die Beantwortung der Folge-Dialoge.

Virtualbox

- Laden Sie die aktuelle opsi-ServerVM von hier: http://uib.de/de/opsi/opsi-testen-download/
- Entpacken Sie den Zip-File, das Verzeichnis `opsivm` wird erzeugt
- Starten Sie Virtualbox.
  Über den Menüpunkt *Datei* / *Appliance importieren* können Sie die Datei `opsivm.ovf` laden.

Allgemein

Den VMware-Player können Sie für alle gängigen Betriebssysteme kostenfrei bei vmware.com beziehen. Er lässt sich in der Regel problemlos installieren, sofern die Ausstattung des Wirtsrechners, insbesondere mit Speicher, den Bedarf mehrerer parallel laufender Betriebssysteme abdeckt.

4.1.1.2 Sprachauswahl

Nach dem Start des Systems müssen Sie die gewünschte Sprache auswählen:

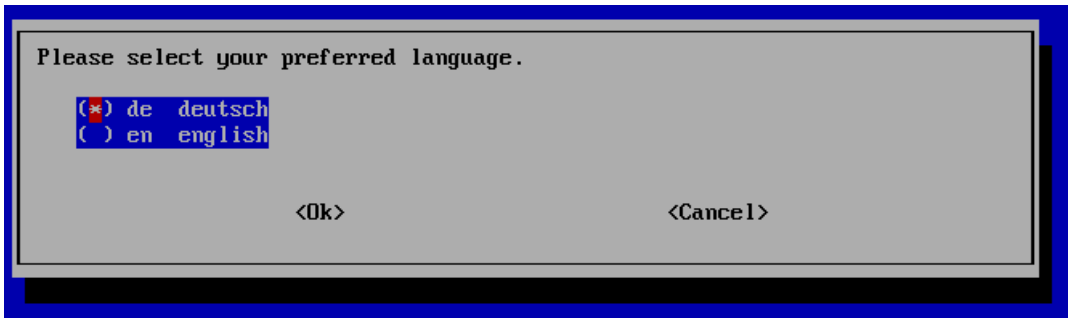

Abbildung 4.1: Sprachauswahl

4.1.1.3 Erster Start

Zur Arbeit mit dem opsi-server ist es von sehr großem Vorteil, wenn dieser direkt mit dem Internet verbunden ist. Zur Netzwerkkonfiguration wird beim ersten Start der VMware-Appliance automatisch das Skript `1stboot.py` aufgerufen. Wenn Sie das System anders aufgesetzt haben oder einen neuen Anlauf nehmen wollen, können sie `1stboot.py` bzw. `/usr/local/bin/1stboot.py` auch auf der Kommandozeile aufrufen.

Die Logdatei von `1stboot.py` ist `/var/lib/1stboot/1stboot.log`.

Warnung
`1stboot.py` eignet sich nicht, um einen konfigurierten opsi-server nachträglich umzubenennen!

Abbildung 4.2: Startmaske

Sie werden dann zur Eingabe von Informationen zur Konfiguration des Netzwerkes aufgefordert. Beantworten Sie die Fragen.

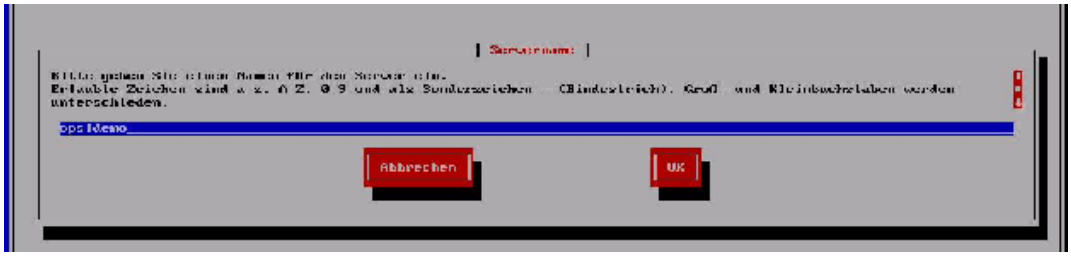

Abbildung 4.3: Eingabemaske

Im Folgenden werden Sie gefragt nach:

Servername
Name diese Servers (ohne Domain) z.B. `opsidepot`

Domain
DNS-Domain (nicht Windows-Domain, muss einen Punkt enthalten) z.B. `opsi.local` oder `meinefirma.local`

IP-Adresse
Adresse dieses Servers z.B. `192.168.1.50`

Netzmaske
Netzmaske dieses Servers z.B. `255.255.255.0`

Windows Domain
Name der Windows Domain (nicht DNS-Domain)

Länderkennung
Für die Erstellung des SSL-Zertifikats: Kennung der Nation in 2 Großbuchstaben, z.B. `DE`

Bundeslandkennung
Für die Erstellung des SSL-Zertifikats: Kennung des Bundeslandes, z.B. `RPL`

Stadt
Für die Erstellung des SSL-Zertifikats: Stadt, z.B. `Mainz`

Firma
Für die Erstellung des SSL-Zertifikats: Firma, z.B. `uib gmbh`

Abteilung
Für die Erstellung des SSL-Zertifikats (Optional)

Mail Adresse
Für die Erstellung des SSL-Zertifikats(Optional): Mailadresse

Gateway
IP-Adresse des Internetgateways, z.B. `192.168.1.1`

Proxy
Soweit zum Internetzugriff benötigt, die Angaben zum Proxy: z.B. `http://myuser:mypass@192.168.1.5:8080`

DNS-Server
IP-Adresse des Nameservers, z.B. `192.168.1.1`

Mailrelay
IP-Adresse des Mailservers z.B. `192.168.1.1`

Tftpserver
Als *TFTP server* geben Sie in der Regel die IP-Nummer des Servers (=*IP-Adresse*) ein.

Passwort für root
Das Passwort für den lokalen Administrator-Benutzer.

Passwort für adminuser
Das Passwort für den lokalen opsi-Administrator.

Nach Abschluss des Programms `1stboot.py` wird die VMware-Maschine, sofern Sie automatisch gestartet war, auch automatisch neu gebootet.

4.1.1.4 Zweiter Start

Nach dem Neustart bzw. nach Fertigstellen der Netzwerkkonfiguration loggen Sie sich als adminuser mit dem von Ihnen vergebenen Passwort ein.

Sie befinden sich direkt auf der graphischen Oberfläche des *opsi-servers* (für diese wird ein Ressourcen schonender Windowsmanager verwendet). Zur Begrüßung erscheint ein „Firefox“-Browser-Fenster mit dem Verweis auf das vorliegende Handbuch und weiteren Hinweisen.

Wenn die Meldung erscheint, dass keine Netzwerkverbindung verfügbar ist, kann dies mit der speziellen Start-Konfiguration der VMware-Appliance zusammenhängen. Vor einer weiteren Fehlersuche sollten Sie zunächst probieren, den Server nochmals zu rebooten (z.B. mit dem Ausschaltknopf in der Bedienleiste unten auf der graphischen Oberfläche).

Abbildung 4.4: Graphische Startoberfläche des opsiservers

Sobald die Netzwerkkonfiguration funktioniert, können Sie auch remote auf den opsi-server zugreifen:

- **Per ssh**
  (in Linux-Systemen stets vorhanden, unter Windows mit putty)
  kommen Sie auf die Kommandozeile des Servers. Als Benutzernamen verwenden Sie root, Sie authentifizieren sich mit dem Root-Passwort.

4.1.1.5 Terminalfenster

Im Folgenden müssen einige Befehle auf der Kommandozeile eingegeben werden. Bei anderen ist die Eingabe der Befehle auf der Kommandozeile der schnellere Weg zum gewünschten Ergebnis.

Ein Fenster zur Text-Eingabe von Befehlen, d.h. ein Terminalfenster, erhält man auf verschiedenen Wegen:

- Remotezugriff per ssh auf den opsi-server (s. vorheriger Abschnitt)
- Öffnen eines Terminalfensters in der graphischen Oberfläche (direkt auf dem opsi-server) durch Klicken auf das Terminal-Icon in der Icon-Leiste der graphischen Oberfläche.
- Öffnen eines Terminalfensters in der graphischen Oberfläche (direkt auf dem opsi-server) durch Rechtsklick in der Fläche und Auswahl von „Terminal“. Dazu hilfreich: die graphische Oberfläche hat mehrere Arbeitsflächen, erreichbar durch die Auswahl-Schaltflächen in der linken oberen Bildschirmecke.

Besonders vorteilhaft ist es, Befehle aus den Anleitungen, z.B. diesem Handbuch, direkt per Kopieren & Einfügen in ein Terminalfenster zu übertragen, soweit die entsprechende Anwendungsumgebung dies unterstützt.

Beispiele aus Konfigurationsdateien sind in den Dokumentationen wie folgt formatiert:

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

Befehle sind folgendermaßen hervorgehoben:

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

In *<spitzen Klammern>* werden Namen dargestellt, die durch ihre Bedeutung ersetzt werden müssen.
Beispiel: Der Fileshare, auf dem die opsi Softwarepakete liegen, wird *<opsi-depot-share>* genannt und liegt auf einem realen Server in der Regel in */var/lib/opsi/depot*.
Das Softwarepaket: `<opsi-depot-share>/ooffice` liegt dann tatsächlich unter `/var/lib/opsi/depot/ooffice`.

4.1.1.6 Überprüfen und ggf. korrigieren der Netzwerkanbindung

Wenn die Netzwerkkonfiguration korrekt ist und der Rechner Anbindung an das Internet hat, können Sie mit dem Browser im Startfenster bereits auf eine beliebige Adresse im Internet zugreifen.

Sofern nicht alles funktioniert, öffnen Sie am besten ein Terminalfenster (möglicherweise geht es dann noch nicht remote, sondern nur auf der Server-Oberfläche), und prüfen die Netzwerkanbindung mit den üblichen, hier nicht zu erklärenden Checks.

Sie können im Terminalfenster auch das Kommando

```
1stboot.py
```

aufrufen und die Netzwerkkonfiguration neu eingeben.

Ein Systemneustart wird dann durch den Befehl

```
reboot
```

erzwungen.

Wenn die Netzwerkanbindung funktioniert, setzen Sie die Konfiguration des opsi-servers fort, um dann mit ersten Installationstest beginnen zu können.

4.1.1.7 Aktualisierung des opsi-Servers

Bringen Sie den opsi-Server auf den aktuellen Stand, indem Sie das Icon *Update Manager* auf dem Desktop doppelt klicken und dann auf *Jetzt installieren* gehen.

4.1.1.8 Installieren der Standard opsi-Produkte

Installieren Sie die Standard opsi-Produkte durch Doppelklick auf das Icon *opsi-product-updater FirstRun*. Bitte geben Sie das Passwort für den adminuser ein. Hierdurch werden automatisiert die aktuellen opsi-Pakete, incl. Templates für Windows- und Linux-Betriebssysteminstallationen, von http://download.uib.de/opsi4.0/products/ geholt und auf dem Server installiert. Für weitere Informationen siehe auch Abschnitt 4.5.

4.1.1.9 Start der Management-Oberfläche

Zur Beschreibung der Management-Oberfläche gehen Sie zu Abschnitt 4.7.

Damit ist die grundlegende Server-Konfiguration beendet.

Sie können nun fortfahren mit

- Kapitel 5
- Kapitel 6

4.1.2 Installation auf einem Debian / Ubuntu System

Wichtig
Prüfen Sie hier ob Ihre Server Betriebssystem Version von opsi unterstützt wird: Kapitel 3
Beachten Sie bitte die Software und Konfigurationsvoraussetzungen: Abschnitt 2.3.

In diesem Abschnitt wird davon ausgegangen, dass Sie mit dem Debian-Paketsystem vertraut sind (Informationen zu diesem Thema finden Sie in den einschlägigen Büchern, in den man-pages oder im Debian GNU/Linux Anwenderhandbuch).

Bitte beachten Sie, das ein opsi-server im Verzeichnis */var/lib/opsi* einen empfohlenen freien Speicher von mindestens 16 GB benötigt.

Für die folgenden Installations-Aufrufe muss das Programm aptitude installiert sein:

```
apt-get install aptitude
```

Wir empfehlen zunächst folgende Installationen:

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

Weiterhin muss Samba installiert sein:

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

Nun sollten Sie den MySQL-Server installieren, damit Sie MySQL als Backend z.B. für die Inventarisierungsdaten und Lizenzmanagement verwenden können. (Die Verwendung von MySQL für Inventariserungsdaten ist kostenlos):

```
aptitude install mysql-server
```

Prüfen Sie den Eintrag für den opsi-server in der Datei `/etc/hosts` oder aber die Ausgabe von

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

Das Ergebnis sollte beispielsweise so aussehen:
192.168.1.1 server.domain.tld server
Sieht das Ergebnis nicht so aus (enthält z.B. *127.0.0.1* oder *localhost*) dann müssen Sie Ihre */etc/hosts* oder Namensauflösung zunächst korrigieren.

Um nun opsi zu installieren, erstellen Sie die Datei
/etc/apt/sources.list.d/opsi.list und tragen dort das opsi-Repository gemäß
Ihrem Betriebssystem ein:

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 ./
```

Führen Sie nun folgende Befehle aus, um die Signatur des Repositories zu importieren:

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 -
```

Alle:
Prüfen Sie ob der Import erfolgreich war:

```
apt-key list
```

sollte unter anderem enthalten:
`pub 2048R/D8361F81 2017-09-30 [verfällt: 2019-12-09] uid home:uibmz:opsi OBS Project <home:uibmz:opsi@build.opensuse.org>`

Führen Sie nun folgende Befehle aus um opsi auf dem Server zu installieren:

```
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
```

Falls Sie bei der Installation von opsi-tftp-hpa nach dem Tftp-Basisverzeichnis gefragt werden, beantworten Sie diese Frage mit */tftpboot*. Die Fragen nach Multicast Support können Sie mit *Nein* beantworten.

Bei der Installation des Paketes opsiconfd werden Sie nach Angaben zur Erstellung eines lokalen SSL-Zertifikates gefragt.

Bei der Installation des opsi-servers werden Sie gefragt, ob die smb.conf gepatcht werden darf. Beantworten Sie die Fragen mit *Ja*. Weiterhin werden Sie nach einem Passwort für den User pcpatch gefragt. Vergeben Sie ein Passwort (und beachten Sie den folgenden Abschnitt zum Ändern der Passwörter).

Wenn Sie das opsi Managementinterface *opsi-configed* direkt auf dem Server ausführen möchten, so benötigen Sie eine aktuelle Java Laufzeitumgebung.

Um diese zu installieren gehen Sie bitte wie folgt vor:

Debian: Fügen Sie in der `/etc/apt/sources.list` den Repository die branches `non-free` und `contrib` hinzu.

Das Ergebnis könnte zum Beispiel so aus sehen:

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

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

Wichtig: Setzten Sie *lenny* ein so muss da natürlich statt *squeeze lenny* stehen.

Um die Java Laufzeitumgebung zu installieren geben Sie nun ein:

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

Ubuntu Trusty: Um die Java Laufzeitumgebung zu installieren geben Sie nun ein:

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

Ubuntu Xenial: Um die Java Laufzeitumgebung zu installieren geben Sie nun ein:

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

Debian 8 (Jessie) Besonderheiten: Das bootimage hat Probleme die `opsi_depot`-Freigabe per mount.cifs zu mounten. Um dieses Problem zu verhindern, muss entweder ID mapping in Samba entsprechend konfiguriert sein oder der Dienst `winbind` darf nicht laufen. Falls `winbind` nicht benötigt wird, wird empfohlen diesen zu deaktivieren.

Start von winbindd deaktivieren:

```
systemctl disable winbind
```

oder

```
insserv -r winbind
```

Für eine funktionierende Konfiguration von ID mapping geben Sie einen beschränkten Zuweisungsbereich in `smb.conf` und starten anschließend Samba neu.

Um ID mapping zu konfigurieren, können Sie das Folgende in die Sektion `[global]` Ihrer `smb.conf` einfügen.

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

Da Sie opsi auf einer existierenden Maschine eingespielt haben, gehen wir davon aus, dass Ihre Netzwerkkonfiguration korrekt ist.
Machen Sie daher mit dem Punkt Abschnitt 4.2.3 weiter.

4.1.3 Installation auf einem Univention Corporate Server (UCS)

Wichtig
Prüfen Sie hier ob Ihre Server Betriebssystem Version von opsi unterstützt wird: Kapitel 3
Beachten Sie bitte die Software und Konfigurationsvoraussetzungen: Abschnitt 2.3.

Die Installation auf einem Univention Corporate Server ist sowohl über das dort vorhandene App Center als auch auf klassischem Wege über die von uib gepflegten Repositories möglich.

Anmerkung
Aktuell werden Installationen auf einem System mit 32bit- und 64bit-Architektur untestützt.
In Zukunft wird der Support für 32bit eingestellt, weshalb wir nur noch den Einsatz von 64bit empfehlen.

4.1.3.1 Installation im Univention App-Center

Über das Univention App-Center steht eine automatisierte Installation des opsi-Servers zur Verfügung. Die Installations-App für opsi befindet sich in der UCS-Management-Oberfläche in der Kategorie "System". Über das App-Center kann opsi nur auf den UCS-Servern Master und Backup installiert werden. Falls Sie ein Update einer vorhandenen opsi-Umgebung durchführen möchten, lesen Sie bitte auch das darauf folgende Kapitel.
Bei der Installation geschieht folgendes:

- Es wird die Grundlage für die MySQL-Installation gelegt.
- Die Ignore-List des Samba4-Connectors wird so angepasst, dass der User pcpatch entfernt wird.
- Das MySQL-Backend wird für die Nutzung der Inventarisierungsdaten konfiguriert (Anpassung der dispatch.conf) falls dies der erste opsi-Server in dieser Umgebung ist.
- Mit opsi-Bordmitteln wird die Samba-Konfiguration überprüft, um sicherzustellen, dass die für opsi benötigten Samba-Shares richtig angelegt sind.
- Zur Installation des opsi-Servers werden automatisch weitere Pakete installiert: opsi4ucs, opsi-atftpd, p7zip-full, cabextract, mysql-server.

Dazu wird bei der Installation selbständig geprüft, welche UCS-Systemrolle der Server hat.

- Der *opsi-product-updater* wird aufgerufen zum Einspielen der minimalen opsi-Produkte. Nach dem Download werden die Pakete auf dem Server installiert.
  Falls ein existierender Configserver entdeckt wird, so wird *opsi-product-updater* so konfiguriert, dass er die Pakete von diesem Server bezieht.

Bitte beachten Sie, dass anschließend kein automatischer Transfer von Clients nach opsi stattfindet. Mehr Informationen unter Abschnitt 4.1.3.6.

Die opsi-Installation auf einem UCS-Server über das *Univention* App-Center ist damit abgeschlossen. Weiter geht es mit: Abschnitt 4.6

4.1.3.2 Update vorhandener opsi-Installation von einem UCS 3 auf UCS 4 (über App-Center)

Eine der wichtigsten Änderungen, die mit dem Update für opsi 4.0.5 gekommen ist, ist die Unterstützung der Gruppe *opsifileadmins*. Diese Gruppe ersetzt in UCS zukünftig die Gruppe *pcpach*. Opsifileadmins wurde zwar schon mit der Unterstützung von UCS 3.0 eingeführt, es wurde allerdings nur auf Installationen eingerichtet, die mit Samba 4 und den Univention Directory Services (Samba4-AD) installiert wurden. Auf allen anderen Varianten und Rollen, wurde bisher die Gruppe als *pcpatch* angelegt.
Da dieser Umstand nicht nur im Installationshandling ein Problem darstellt, sondern auch Probleme bei möglichen Migrationen (besonders von Samba3 auf Samba4) verursachen kann, wird seit opsi 4.0.5 die Gruppe *pcpatch* als *opsifileadmins* angelegt.

Warnung
Um diese Anforderungen sauber in das Integrationspaket implementieren zu können, wird die Gruppe *pcpatch*, falls vorhanden, automatisch über das Integrationspaket in *opsifileadmins* umbenannt. Dies geschieht über das join-Skript. Falls Ihr Configserver auf einem Master oder Backup betrieben wird, wird das Join-Skript automatisch ausgeführt.

Der Hauptgrund für diese drastische Maßnahme ist, dass die manuelle Umbenennung dieser Gruppe nicht trivial ist, da es sich um eine primäre Gruppe handelt. Wir empfehlen vor dem Einspielen dieses Updates zu überprüfen, ob Ihre Gruppe noch pcpatch heißt. Wenn dem so ist, sollte das Update - angefangen beim Configserver - auf allen Depotservern zeitnah eingespielt werden, da ansonsten der Betrieb der Multidepot Umgebung gestört wird. Wenn Ihre Gruppe schon opsifileadmins heißt, sollten keine Beeinträchtigungen auftreten. Dennoch wird empfohlen, nach dem Einspielen der Updates alle opsi-Server auf Funktion zu überprüfen.

4.1.3.3 Manuelle opsi-Installation unter UCS (ohne App-Center)

Beachten Sie bitte Abschnitt 2.3.

Notwendige Vorbereitungen:

- Der Befehl

```
hostname -f
```

  muss einen fully qualified domainname zurück liefern, in dem mindestens zwei Punkte vorkommen, z.B. *opsiserver.domain.local*.

- Der Befehl

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

  muss die IP-Adresse der Netzwerkschnittstelle zurück liefern, zu der sich die Clients verbinden sollen. Gibt der Befehl die Adresse *127.0.0.1* oder *127.0.0.2* aus, muss die Datei */etc/hosts* korrigiert werden.

- Samba muss konfiguriert sein. Für den Einsatz auf einem Server in der Rolle Member muss `univention-samba` anstatt `univention-samba4` zum Einsatz kommen.
- Soll die Maschine als Configserver fungieren, sollte `mysql-server` installiert sein.
- Soll die Maschine auch als DHCP-Server eingesetzt werden, muss der Daemon dhcpd konfiguriert und am Laufen sein.

Die Installation von opsi ist möglich auf den Rollen Master, Backup, Slave und Member. Für die Installation auf einem Member beachten Sie unbedingt Abschnitt 4.1.3.4!

Die folgende Dokumentation geht von einer Installation auf dem Master mit samba4 aus.

Achtung
Wird die Installation auf einem *Slave* ausgeführt, muss dieser bereits gegenüber dem Master gejoint und während der Grundinstallation Samba 4 installiert worden sein.
Die UCS-Konfiguration wird überlicherweise auf dem *Master* vorgenommen, während die Installation und Konfiguration von opsi auf dem *Slave* geschieht.

Die klassischen Installationsvariante mit dem Benutzer: `pcpatch` mit der primären Gruppe `pcpatch` kann unter UCS nicht eingehalten werden. Da Samba4 den grundlegenden Restriktionen von Active-Directory unterliegt, sind Gruppen mit der gleichen Bezeichnung wie User (wie in Unix/Linux üblich) nicht mehr erlaubt. Aus diesem Grund wurde für UCS 3 die neue Konfigurationsdatei `/etc/opsi/opsi.conf` eingeführt, über welche gesteuert wird, wie die Gruppe für den Samba-Zugriff auf die Freigaben bestimmt wird. Seit UCS 3 wird über diese Datei der Gruppenname `pcpatch` umbenannt und heißt von nun an: `opsifileadmins`. Das bedeutet, dass die User, die Zugriffsrechte für die Freigaben von opsi erhalten müssen (opsi-Paketierer) nicht Mitglied der Gruppe `pcpatch`, sondern Mitglied der Gruppe `opsifileadmins` sein müssen. Diese Besonderheit gilt nur für UCS und unterscheidet sich von den anderen Distributionen und in weiterführenden Kapiteln der opsi-Dokumentationen. Unter UCS wird der User pcpatch als vollwertiger Domänenbenutzer angelegt. Nähere Informationen über diese neue Konfigurationsdatei können Sie dem Handbuch entnehmen.

- Um opsi zu installieren, tragen Sie das entsprechende Repository ein:*

UCS 4.2:

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

Nun muss noch der Schlüssel des Repositories ins System importiert werden:

UCS 4.2

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

Für die Installation führen Sie nun folgende Befehle aus:

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

Ist die Rolle des Zielsystems eine andere als Master oder Backup, muss nun noch das opsi4ucs Join-Skript ausgeführt werden:

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

Unter der URL `https://<servername>:4447` finden Sie nun einen Link zur opsi Management Oberfläche.

Abschliessend muss noch im UDM unter dem Punkt Freigaben für die `opsi_depot`-Freigabe unter Erweiterte Einstellungen → Erweiterte Samba-Einstellungen: Die Option: `follow symlinks` auf `yes` gesetzt werden. Das gleiche sollte man am besten auch für die `opsi_depot_rw`-Freigabe erledigen, damit später die Treiberintegration keine Probleme verursacht. Sollte sich das `/var/lib/opsi/depot`-Verzeichnis auf einer extra Partition oder Festplatte befinden, muss man noch zusätzlich für diese Freigaben die Option wide links auf *yes* setzen.

Um sicher zu stellen, dass alle Einstellungen von opsi nun korrekt übernommen wurden, sollte man noch folgende Befehle ausführen:

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

Bei samba4 handelt es sich um einen zentralen Dienst, daher wird dies nicht automatisch vom Paket ausgeführt, sondern muss nachträglich manuell durchgeführt werden. Nach dem Neustart von samba kann es zu einer kleinen Verzögerung beim Zugriff auf die neuen Freigaben kommen, wir bitten dies zu berücksichtigen.

Da es keinen direkten Kontakt zwischen Univention-LDAP und dem opsi-Backend gibt, müssen alle Clients erst im LDAP über udm angelegt werden und danach nochmal mit opsi im opsi-System mit allen Informationen (insbesondere der MAC-Addresse) angelegt werden. Das Löschen des Clients im Univention-LDAP sorgt nicht dafür, dass der Client unter opsi auch gelöscht wird. Unter Abschnitt 4.1.3.6 wird eine Lösung dafür beschrieben.

Da Sie opsi auf einer existierenden Maschine eingespielt haben, gehen wir davon aus, dass Ihre Netzwerkkonfiguration korrekt ist.
Machen Sie daher mit dem Punkt Abschnitt 4.2.3 weiter.

Warnung
Die verwendeten Unix-Befehle in den folgenden Kapiteln beziehen sich auf Debian-Systeme, Sie müssen sie evtl. durch die entsprechenden UCS-Befehle ersetzen.

4.1.3.4 Hinweise zur opsi-Installation auf einem UCS-Server in der Rolle *Member*

Warnung
Da der Einsatz von opsi auf Member-Servern besonderen Einschränkungen unterliegt, empfehlen wir Anfängern den Einsatz von opsi auf einer anderen Rolle.

Die Installation von opsi auf einem Server in der Rolle Member ist möglich. Eine automatisierte Installation über das Univention App Center kann jedoch aktuell nicht vorgenommen werden.

Nach einer Installation muss sicher gestellt werden, dass der für den Depotzugriff verwendete Benutzer mitsamt Domäne aufgeführt ist. Dazu muss der Hostparameter `clientconfig.depot.user` kontrolliert werden. Ist die Domäne `backstage`, so muss der Wert `backstage\pcpatch` lauten. Lautet er `memberserver\pcpatch`, so muss der Wert korrigiert werden.

Das Setzen des Kennworts für den Benutzer `pcpatch` über `opsi-admin` scheitert auf Grund der fehlenden AD-Schreibberechtigungen eines Member-Servers. Daher muss die Änderung des Kennworts des technischen Benutzers `pcpatch` **zusätzlich** auf einem Server mit Schreibrechten im AD vorgenommen werden - ein Master, Backup oder Slave.

4.1.3.5 Konfiguration des PXE-Boot für Betriebssysteminstallationen

Wenn der PXE-Boot für die Betriebssysteminstallation eingesetzt werden soll, muss der DHCP-Dienst auf dem entsprechenden UCS-System umkonfiguriert werden. Hier gibt es zwei Besonderheiten, die UCS von den anderen unterstützten Distributionen unterscheidet.

- Die Konfiguration wird nicht automatisch vorgenommen, da bei einer Installation von opsi in eine produktive UCS-Infrastruktur in der Regel schon diese Konfigurationen existieren.
- Ist der opsi-atftpd so konfiguriert, dass er nicht wie sonst das Verzeichnis `/tftpboot` als Basisverzeichnis nutzt, sondern `/var/lib/univention-client-boot`. Alle wichtigen Dateien vom opsi-linux-bootimage werden deshalb vom /tftpboot in das Basisverzeichnis verlinkt. Dies hat den Nebeneffekt, dass die DHCP-Option: Filename `pxelinux.0` statt `linux/pxelinux.0` lauten muss.

Um die oben genannte Konfigurationen um zu setzen, muss im UCS-System eine Richtlinie erstellt werden. Diese Einstellung ist abhängig von den schon vorhandenen Richtlinien und müssen entsprechend umgesetzt werden. Wenn opsi auf einem UCS-Testsystem installiert wurde und keine Richtlinien entsprechend existieren, sollte erst geprüft werden ob der DHCP-Dienst installiert ist, wenn nicht muss das noch nachgeholt werden. Wenn der DHCP-Dienst schon installiert wurde, ist der einfachste Weg die Richtlinie in der UMC-Webschnittstelle (Univention Management Console) vom UCS-Server zu erstellen. Dazu wählt man den die Kategorie "Domäne" und darunter das Modul DHCP-Server. Als nächstes muss man den Service wählen (Bei einem Testsystem gibt es in der Regel nur einen Eintrag). In der folgenden Detailansicht wählt man den Menüpunkt Richtlinien. Die Richtlinie, die hier benötigt wird ist eine DHCP-Boot Richtlinie. Bei der Richtlinienkonfiguration wählt man nun den Defaulteintrag `cn=default-settings` (sollte der einzige Eintrag sein) und wählt bearbeiten. Unter Grundeinstellungen - DHCP Boot für die Option Bootserver die IP vom opsi-server ein und bei Boot-Dateiname muss `pxelinux.0` eingetragen werden.

Warnung
Wenn man die Richtlinie wie oben beschrieben setzt, gilt das für jedes Gerät, welches per DHCP von diesem Server bedient wird. Deshalb noch mal der Hinweis, dass diese Beschreibung nur für Evaluierer gedacht ist, die im Zuge von opsi auch ucs Testen. Bei Installationen in produktive UCS-Umgebungen sollte man diese Richtlinie nicht auf diese Weise setzen.

Optional lassen sich diese Einstellungen auch auf der Konsole mit dem udm-Kommando erledigen. Nähere Informationen dazu entnehmen Sie bitte der entsprechenden UCS-Dokumentation.

4.1.3.6 LDAP-Daten nach opsi überführen

In einer opsi4ucs-Installation müssen die Windows-Clients erst im UDM angelegt werden und erst im folgenden Schritt im opsi-configed erstellt werden. Änderungen von Clients im UDM werden nict an opsi weitergegeben. Wenn ein Client zum Beispiel im LDAP eine neue Mac-Addresse bekommt, wird dies im opsi-System nicht bemerkt. Wenn man nun versucht ein Netboot-Produkt für diesen Client auf setup zu setzen, würde opsi die Bootkonfiguration mit der falschen Mac-Addresse im Bootsystem hinterlegen.

Die Lösung hierzu ist die Erweiterung *opsi-directory-connector*. Weitere Informationen hierzu enthält das Handbuch.

4.1.4 Installation auf openSUSE

Wichtig
Prüfen Sie hier ob Ihre Server Betriebssystem Version von opsi unterstützt wird: Kapitel 3
Beachten Sie bitte die Software und Konfigurationsvoraussetzungen: Abschnitt 2.3.

Notwendige Vorbereitungen:

- Der Befehl

```
hostname -f
```

  muss einen fully qualified domainname zurück liefern, in dem mindestens zwei Punkte vorkommen, z.B. *opsiserver.domain.local*.

- Der Befehl

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

  muss die IP-Adresse der Netzwerkschnittstelle zurück liefern, zu der sich die Clients verbinden sollen. Gibt der Befehl die Adresse *127.0.0.1* oder *127.0.0.2* aus, muss die Datei */etc/hosts* korrigiert werden.

- Samba muss installiert und konfiguriert sein.
- Soll die Maschine als Configserver fungieren, sollte `mariadb-server` installiert sein.
- Soll die Maschine auch als DHCP-Server eingesetzt werden, muss der Daemon dhcpd konfiguriert und am Laufen sein.

Hinzufügen des opsi-SUSE-Repositories per zypper:

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
```

Nach dem Hinzufügen der Repositories kann die Installation gestartet werden:

```
zypper refresh
  Wollen Sie den Schlüssel (a)bweisen, ihm (t)emporär oder (i)mmer vertrauen? [a/t/i/?] (a): i
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
```

Bitte stellen Sie sicher, daß Ihre Firewall Konfiguration die folgenden Ports erlaubt: tftp Port (69/UDP) und die opsi ports (4447/TCP und 4441/TCP).

Falls Sie die Netzwerkkonfiguration mit Hilfe der Tools yast oder autoyast vorgenommen haben kann es sein, dass dieses Tool einen Eintrag in die Datei `/etc/hosts` einen Eintrag nach folgendem Muster angelegt hat:

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

Fall Sie opsi die Konfiguration des DHCP Servers überlassen wollen muss dieser Eintrag auf die öffentliche IP-Adresse geändert werden, über die der Server erreichbar ist.

Da Sie opsi auf einer existierenden Maschine eingespielt haben, gehen wir davon aus, dass Ihre Netzwerkkonfiguration korrekt ist.
Machen Sie daher mit dem Punkt Abschnitt 4.2.3 weiter.

Warnung
Die verwendeten Unix-Befehle in den folgenden Kapiteln beziehen sich auf Debian-Systeme. Sie müssen sie durch die entsprechenden Suse-Befehle ersetzen.

4.1.5 Installation auf Suse Linux Enterprise Server (SLES)

Wichtig
Prüfen Sie hier ob Ihre Server Betriebssystem Version von opsi unterstützt wird: Kapitel 3
Beachten Sie bitte die Software und Konfigurationsvoraussetzungen: Abschnitt 2.3.

Notwendige Vorbereitungen:

- Der Befehl

```
hostname -f
```

  muss einen fully qualified domainname zurück liefern, in dem mindestens zwei Punkte vorkommen, z.B. *opsiserver.domain.local*.

- Der Befehl

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

  muss die IP-Adresse der Netzwerkschnittstelle zurück liefern, zu der sich die Clients verbinden sollen. Gibt der Befehl die Adresse *127.0.0.1* oder *127.0.0.2* aus, muss die Datei */etc/hosts* korrigiert werden.

- Samba muss installiert und konfiguriert sein.
- Soll die Maschine als Configserver fungieren, sollte `mariadb-server` installiert sein.
- Soll die Maschine auch als DHCP-Server eingesetzt werden, muss der Daemon dhcpd konfiguriert und am Laufen sein.

Hinzufügen des opsi-SLES-Repositories per zypper:

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
```

Nach dem Hinzufügen der Repositories kann die Installation gestartet werden:

```
zypper refresh
  Wollen Sie den Schlüssel (a)bweisen, ihm (t)emporär oder (i)mmer vertrauen? [a/t/i/?] (a): i
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
```

Falls Sie die Netzwerkkonfiguration mit Hilfe der Tools yast oder autoyast vorgenommen haben kann es sein, dass dieses Tool einen Eintrag in die Datei `/etc/hosts` einen Eintrag nach folgendem Muster angelegt hat:

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

Fall Sie opsi die Konfiguration des DHCP Servers überlassen wollen muss dieser Eintrag auf die öffentliche IP-Adresse geändert werden, über die der Server erreichbar ist.

Da Sie opsi auf einer existierenden Maschine eingespielt haben, gehen wir davon aus, dass Ihre Netzwerkkonfiguration korrekt ist.
Machen Sie daher mit dem Punkt Abschnitt 4.2.3 weiter.

Warnung
Die verwendeten Unix-Befehle in den folgenden Kapiteln beziehen sich auf Debian-Systeme. Sie müssen sie durch die entsprechenden Suse-Befehle ersetzen.

4.1.6 Installation auf einem RedHat Enterprise Linux (RHEL)

Wichtig
Prüfen Sie hier ob Ihre Server Betriebssystem Version von opsi unterstützt wird: Kapitel 3
Beachten Sie bitte die Software und Konfigurationsvoraussetzungen: Abschnitt 2.3.

Notwendige Vorbereitungen:

- Der Befehl

  ```
  hostname -f
  ```

  muss einen fully qualified domainname zurück liefern, in dem mindestens zwei Punkte vorkommen, z.B. *opsiserver.domain.local*.

- Der Befehl

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

  muss die IP-Adresse der Netzwerkschnittstelle zurück liefern, zu der sich die Clients verbinden sollen. Gibt der Befehl die Adresse *127.0.0.1* oder *127.0.0.2* aus, muss die Datei */etc/hosts* korrigiert werden.

- xinetd installieren:

```
yum install xinetd
```

- Samba und Datenbankserver installieren:

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

- Samba und Datenbankserver konfigurieren:

```
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
```

- Soll die Maschine auch als DHCP-Server eingesetzt werden, muss der Daemon dhcpd konfiguriert und am Laufen sein.

Beim Red Hat Network registrieren:

```
rhn_register
```

Hinzufügen des RHEL Repositories:

Für 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
```

Nach dem Hinzufügen der Repositories kann die Installation gestartet werden:

```
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
```

Es kann vorkommen, dass eine Nachfrage zum Importieren des GPG-Schlüssel des Repositories wie die nachfolgene Meldung während der Installation gezeigt wird:

```
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
```

Bitte beantworten Sie diese Nachfrage mit *y*.

Bitte stellen Sie sicher, dass Ihre iptables- und SELinux-Konfiguration die folgenden Ports erlaubt: tftp Port (69/UDP) und die opsi ports (4447/TCP und 4441/TCP).

Da Sie opsi auf einer existierenden Maschine eingespielt haben, gehen wir davon aus, dass Ihre Netzwerkkonfiguration korrekt ist.
Machen Sie daher mit dem Punkt Abschnitt 4.2.3 weiter.

Warnung
Die verwendeten Unix-Befehle in den folgenden Kapiteln beziehen sich auf Debian-Systeme. Sie müssen sie durch die entsprechenden RHEL-Befehle ersetzen.

4.1.7 Installation auf einem CentOS Server

Wichtig
Prüfen Sie hier ob Ihre Server Betriebssystem Version von opsi unterstützt wird: Kapitel 3
Beachten Sie bitte die Software und Konfigurationsvoraussetzungen: Abschnitt 2.3.

Notwendige Vorbereitungen:

- Der Befehl

  ```
  hostname -f
  ```

  muss einen fully qualified domainname zurück liefern, in dem mindestens zwei Punkte vorkommen, z.B. *opsiserver.domain.local*.

- Der Befehl

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

  muss die IP-Adresse der Netzwerkschnittstelle zurück liefern, zu der sich die Clients verbinden sollen. Gibt der Befehl die Adresse *127.0.0.1* oder *127.0.0.2* aus, muss die Datei */etc/hosts* korrigiert werden.

- xinetd installieren:

  ```
  yum install xinetd
  ```

- Samba und Datenbankserver installieren:

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

  Unter CentOS 7 wurde mysql-server durch mariadb-server ersetzt. Sie können folgenden Befehl für die Installation verwenden:

- Samba und Datenbankserver konfigurieren:

```
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
```

- Soll die Maschine auch als DHCP-Server eingesetzt werden, muss der Daemon dhcpd konfiguriert und am Laufen sein.

Hinzufügen des CentOS Repositories:

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
```

Nach dem Hinzufügen der Repositories kann die Installation gestartet werden:

```
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
```

Es kann vorkommen, dass eine Nachfrage zum Importieren des GPG-Schlüssel des Repositories wie die nachfolgene Meldung während der Installation gezeigt wird:

```
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-6/repodata/repomd.xml.key
Is this ok [y/N]: y
```

Bitte beantworten Sie diese Nachfrage mit *y*.

Bitte stellen Sie sicher, dass Ihre iptables- und SELinux-Konfiguration die folgenden Ports erlaubt: tftp Port (69/UDP) und die opsi ports (4447/TCP und 4441/TCP).

Da Sie opsi auf einer existierenden Maschine eingespielt haben, gehen wir davon aus, dass Ihre Netzwerkkonfiguration korrekt ist.
Machen Sie daher mit dem Punkt Abschnitt 4.2.3 weiter.

Warnung
Die verwendeten Unix-Befehle in den folgenden Kapiteln beziehen sich auf Debian-Systeme, Sie müssen sie durch die entsprechenden CentOS-Befehle ersetzen.

4.2 Aktualisieren und Konfigurieren des opsi-servers

4.2.1 Proxy-Eintrag in apt-Konfiguration

Sofern für Ihren Internet-Zugang erforderlich, passen Sie die Datei `/etc/apt/apt.conf` an Ihre Netzwerkgegebenheiten an (richtigen Proxy eintragen oder Zeile auskommentieren / löschen). Eine Datei können Sie editieren z. B. mithilfe des Programms „midnight commander“:

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

4.2.2 Aktualisierung des opsi-servers

Bringen Sie den opsi-server auf den letzten Stand, in dem Sie nacheinander in einem Terminalfenster die folgenden Kommandos aufrufen:

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

Tipp
Sollte beim Update nachgefragt werden, ob die `smb.conf` überschrieben werden soll, muss man dies bestätigen. Sollte die `smb.conf` vorher schon geändert worden sein, sollte man den default beibehalten und später die Dateien miteinander abgleichen. Sollte diese Nachfrage schon mit Nein beantwortet worden sein, kann man dies später auf dem opsi-server durch ausführen von `opsi-setup --auto-configure-samba` nachholen.

4.2.3 Backend-Konfiguration

Opsi unterstützt zur Datenhaltung unterschiedliche Backends.

Im Wesentlichen sind dies:

- file (Datenhaltung in Dateien)
- mysql (Datenhaltung in einer MySQL-Datenbank)

Anmerkung
Einige Distributionen verwenden *MariaDB* anstatt *MySQL*.
Das *mysql*-Backend funktioniert auch mit MariaDB.

Achtung
MySQL-Server verwendet seit Version 5.7 den vorher optionalen *strict mode* nun standardmäßig. Dies führt zu einem Fehlschlag des Befehls `opsi-setup --configure-mysql`. Dementsprechend sollte vor dem Befehlsaufruf die Datei `/etc/mysql/mysql.conf.d/mysqld.cnf` editiert werden.
In der `[mysqld]` Sektion muss nun folgende Zeile eingefügt werden:
`sql_mode=NO_ENGINE_SUBSTITUTION`

Danach muß der Dienst `mysql` neu gestartet werden: `service mysql restart`

Anmerkung
Die Verwendung des mysql-Backend für Inventariserungsdaten ist kostenlos.
Mehr Informationen zur Freischaltung finden Sie hier.

Daneben gibt es noch für spezielle Zwecke die Backends:

- **opsipxeconfd** (der Dienst für den opsi pxe-boot)
- **dhcpd** (zur Kommunikation mit dem dhcp-Server auf dem opsi-server)
- **jsonrpc** (zur Weiterleitung aller Anfragen an einen anderen Server)

Um (wie empfohlen) das mysql-Backend verwenden zukönnen (z.B. für die Inventarisierung), so muss dieses nun mit dem Befehl

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

initialisiert werden.

Es wird davon ausgegangen, dass MySQL eingerichtet wurde und die Zugangsdaten eines Datenbank-Administrators bekannt sind. Gezielte Informationen zu Installation und Einrichtung entnehmen Sie bitte den Handbüchern Ihrer Distribution.

Der Befehl wird nach den Zugangsdaten zum Datenbanksystem fragen, um eine Datenbank und einen Benutzer mit entsprechenden Berechtigungen für diese Datenbank für opsi anzulegen.

Eine Beispiel-Sitzung:

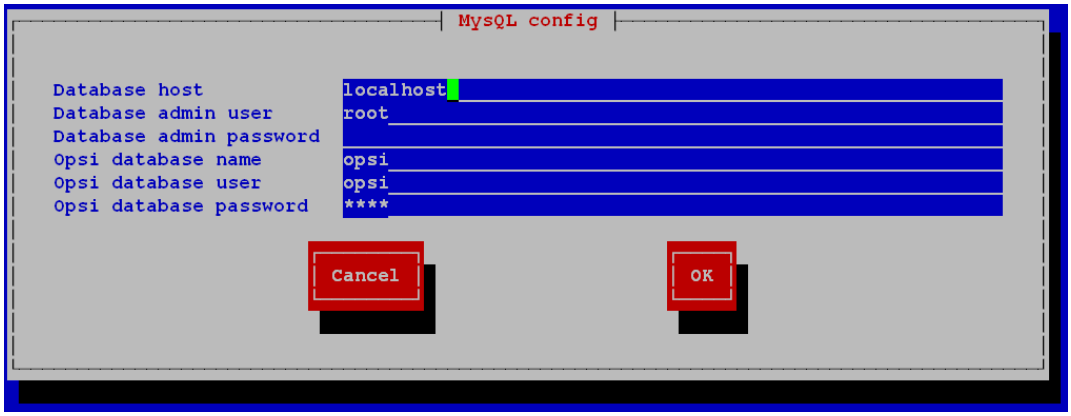

Abbildung 4.5: opsi-setup --configure-mysql: Eingabemaske

```
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
```

Abbildung 4.6: opsi-setup --configure-mysql: Ausgabe

Bei den Abfragen können außer beim *Database Admin Password* alle Vorgaben mit Enter bestätigt werden. Das *Database Admin Password* ist auf der opsi-VM *linux123* ansonsten das, was Sie bei der Installation des mysql-servers vergeben haben.

Unterschiedliche Daten können in unterschiedlichen Backends gehalten werden. Über bestimmte Vorgänge müssen mehrere Backends informiert werden. Hierzu werden die opsi-Methoden den Backends zugeordnet. Dies geschieht in der Datei
`/etc/opsi/backendManager/dispatch.conf`.

Hier ein Beispiel:

```
# = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = =
# =      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
```

In dieser Datei sind oben Erläuterungen und Beispielkonfigurationen angegeben. In den nicht auskommentierten Zeilen steht vorne der Name der opsi-Methoden (mit wildcard *) und nach einem Doppelpunkt die hierfür zuständigen Backends. Bei jedem Methodenaufruf wird anhand dieser Liste geprüft, welche Backends aufgerufen werden müssen. Dabei wird die erste Zeile genommen die zu dem Methoden Namen passt. Die letzte Zeile (.*) passt auf jeden Methoden Namen.

Die Defaulteinstellung bei Installation der Pakete ist das file-Backend als Haupt-Backend.

Achtung
Achten Sie darauf, dass alle verwendeten Backends in der Zeile beginnend mit *backend_.** aufgeführt werden.

Bei Inbetriebnahme des Servers (auch ohne Änderung der Datei) und nach jeder Änderung dieser Datei sollten Sie folgende Befehle ausführen:

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

Wer welche Methoden verwenden darf, ist in der Datei */etc/opsi/backendManager/acl.conf* festgelegt.

4.2.4 Samba-Konfiguration anpassen und Ändern von Passwörtern

Um sicherzustellen, dass die für opsi erforderlichen Samba-Shares verfügbar sind, führen Sie bitte den folgenden Befehl aus:

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

Auf dem System ist ein Pseudo-User *pcpatch* eingerichtet. Die PCs melden sich zwecks Installation von Softwarepaketen als dieser User an und haben dann Zugriff auf die Installationsdateien auf den hierfür vorgesehenen Shares. Der User *pcpatch* muss mit korrektem Passwort – gleichzeitig als System-User, als Samba-User und als opsi-User – eingerichtet werden.

Rufen Sie in einem Terminalfenster das Programm opsi-admin mit der Option zum Setzen des pcpatch-Passwortes (in einem für opsi, Unix und Samba).

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

Nach „Abschicken“ des Befehls werden Sie zur Passworteingabe aufgefordert.

4.2.5 Überprüfung der Java-Konfiguration

Zur Verwaltung des opsi-servers und der angeschlossenen Clients dient das Programm opsi-configed. Es kommuniziert über HTTPS mit dem opsi-Server und kann daher auf jedem Rechner verwendet werden, der eine entsprechende Verbindung aufbauen kann. Dieses Programm benötigt auf den Rechnern, auf denen es ausgeführt wird, eine Java-Runtime-Umgebung mindestens in Version 7 bzw., wie die interne Versionszählung lautet, Version 1.7

Wenn der opsi-server eine graphische Oberfläche installiert hat, können Sie den opsi-configed auch direkt auf dem Server aufrufen; dies ist in der virtuellen Maschine vorbereitet. Kann oder soll der opsi-configed nicht auf dem Server ausgeführt werden, können Sie dieses Kapitel überspringen.
Kontrollieren Sie, ob Java in der benötigten Version installiert ist, indem Sie in einem Terminal

```
java -version
```

aufrufen.

Sollte hier nicht mindestens Java Version *1.7.0* angezeigt werden, so müssen Sie eine aktuellere Version aktivieren.

Sie können in einem Terminalfenster schauen, was update-alternatives zeigt und die Einstellung anpassen:

```
update-alternatives --config java

Es gibt 2 Auswahlmöglichkeiten für die Alternative java (welche /usr/bin/java bereitstellen).

* 0            /usr/lib/jvm/java-6-openjdk-i386/jre/bin/java   1061      Auto-Modus
  1            /usr/lib/jvm/java-6-openjdk-i386/jre/bin/java   1061      manueller Modus
  2            /usr/lib/jvm/java-7-openjdk-i386/jre/bin/java   1051      manueller Modus


Drücken Sie die Eingabetaste, um die aktuelle Wahl[*] beizubehalten,
oder geben Sie die Auswahlnummer ein: 2
```

Wenn, wie hier die Ausgabe angezeigt hat, mehrere Versionen installiert sind, können Sie auch einfach die ungeeigneten älteren entfernen:

```
apt-get remove openjdk-6-jre
apt-get remove openjdk-6-jre-lib
```

Falls Java 7 (oder höher) noch nicht installiert war, installieren Sie eine aktuelle Version:

```
apt-get install openjdk-7-jre
```

Danach sollte

```
java -version
```

mindestens Java Version 1.7.0 anzeigen.

Noch ein Hinweis: Achten Sie bei der virtuellen Maschine auf eine ausreichende Auflösung des virtuellen Bildschirms. Für den opsi-configed ist es sinnvoll, eine Auflösung ab 1024x768 zu verwenden. Für die Verbesserung der Graphik- und Maustreibersituation bei einer höheren Auflösung, hilft es, die VMware Tools bei einer VMware-Maschine bzw. die Gasterweiterungen bei einer VirtualBox-Maschine zu installieren.

4.2.6 User einrichten und Gruppen opsiadmin und pcpatch pflegen

Im folgenden wird als Beispiel der neue Benutzer *adminuser* so eingerichtet, wie Sie ihn sich einrichten sollten.

Zunächst wird der User erstellt:

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

Wir vergeben nun Passwörter für Unix:

```
passwd adminuser
```

und für Samba

```
smbpasswd -a adminuser
```

Achtung
Verwenden Sie in den Passwörtern kein § da dies bei der Verbindung zum opsi-Service nicht erlaubt ist.

Nun wird die Gruppenmitgliedschaft eingerichtet und getestet mit der Befehlsfolge:

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

Der getent-Befehl sollte dann so etwas ausgeben wie:
opsiadmin:x:1001:opsiconfd,adminuser

Alle User, die Produkte packen (opsi-makeproductfile), installieren (opsi-package-manager) oder Konfigurationsdateien manuell bearbeiten wollen, müssen zusätzlich in der Gruppe pcpatch sein:

```
usermod -aG pcpatch adminuser
```

Der Test

```
getent group pcpatch
```

ergibt
pcpatch:x:992:adminuser

Damit Mitglieder der Gruppe pcpatch den Befehl `sudo opsi-set-rights` nutzen können führen Sie bitte aus:

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

Dann kann `opsi-set-rights`
(macht das selbe wie `opsi-setup --set-rights`), nicht nur als root, sondern auch per sudo von Mitgliedern der Gruppe pcpatch (bzw. opsi-file-admins) aufgerufen werden.
Beispiel:

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

Root darf dies alles ohnehin und muss daher nicht explizit in die Gruppe aufgenommen werden.

4.3 DHCP-Konfiguration

Wichtig:

Eine korrekt funktionierende Namensauflösung und DHCP ist für das Funktionieren von opsi essentiell. Um die Installation zu vereinfachen, ist die von uib bereitgestellte VM schon mit einem DHCP-Server ausgestattet. Auf der anderen Seite ist im Produktivbetrieb in der Regel ein DHCP-Server schon vorhanden, der weiter genutzt werden soll. Daher werden im folgenden beide Alternativen beschrieben.

4.3.1 Alternative: DHCP auf dem opsi-server

Vorkonfigurierte VM: In der vorkonfigurieren opsi VM ist bereits ein DHCP-Server installiert.
Der DHCP-Server auf der opsi-server VM ist so konfiguriert, das er keine freien leases hat, also keine IP-Nummern an unbekannte Rechner vergibt. Wenn Sie im opsi-configed einen Client erzeugen, müssen Sie daher IP-Nummer und MAC-Adresse angeben, da diese in die */etc/dhcp/dhcpd.conf* eingetragen und danach der DHCP Dienst neu gestartet wird.

Eigene Installation: Wenn Sie den opsi-Server als DHCP-Server verwenden möchten, müssen Sie daher das entsprechende Paket manuell nachinstallieren

z.B. mit

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

Nach der Installation muß die Konfigurationsdatei noch angepasst werden mit dem Befehl:

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

Damit der in der Konfiguration-Datei */etc/opsi/backends/dhcpd.conf* aufgeführte Befehl zum Neustarten des DHCP-Servers vom opsiconfd ausgeführt werden darf, ist ein Eintrag unter /etc/sudors notwendig. Dieser wird über den Aufruf

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

erstellt.

Gegebenenfalls wären noch die Berechtigungen der DHCPD Konfigurationsdatei zu kontrollieren, die zB. so aussehen sollten:

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

4.3.2 Alternative: externer DHCP-Server

Vorkonfigurierte VM: Wenn Sie die opsi-VM verwenden dann können Sie den DHCP-Server deinstallieren: Dazu führen Sie folgende Befehle aus:

```
aptitude remove dhcp3-server
```

oder

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

Eigene Installation: Bei einer eigenen Installation wird seit opsi 4.0.3 nicht mehr automatisch ein DHCP-Server installiert.

Nun müssen Sie den externen DHCP-Server so konfigurieren, dass er ein PXE-Boot über den opsi-server ermöglicht. Wenn Ihr DHCP-Server auf einem Linux läuft, sind folgende Einträge in der Konfigurationsdatei des dhcpd (z.B. */etc/dhcp3/dhcpd.conf*) für die Clients notwendig:

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

Wobei <ip of opsi-server> durch die IP-Nummer des opsi-servers zu ersetzen ist.

Läuft der opsi-Server auf openSUSE oder SLES, so ist `filename=opsi/pxelinux.0`.
Läuft der opsi-Server auf UCS, so ist `filename=pxelinux.0`.

Bei einem Windows-Server sind die entsprechenden Einträge *Startserver (Option 66)* und *Startfile (Option 67).*

Wenn Sie im opsi-configed einen Client erzeugen, müssen Sie die MAC-Adresse angeben, aber keine IP-Nummer.

4.3.3 Überprüfung/Anpassung Backendkonfiguration für DHCP-Nutzung

Je nachdem ob der interne oder ein externer DHCP-Server verwendet wird, muss die Konfiguration von opsi angepasst werden.

In der Datei */etc/opsi/backendManager/dispatch.conf* ist festgelegt, welche Backends von opsi zum Einsatz kommen (file, mysql).

In den Zeilen *backend_. und* host_. wird u.a. gesteuert, ob der opsi-server auch die lokale DHCP-Konfiguration – also die Zuweisung von Internet-Adressen zu den Hardware-Adressen der Netzwerkkarten – mit übernimmt. Dies muss so eingerichtet sein, wenn für die opsi-clients die DHCP-Einträge durch die opsi-Konfigurationsaufrufe erzeugt werden sollen. Der entsprechende Eintrag mit file Backend muss dann z.B. lauten:

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

Wenn der opsi-server den DHCP-Dienst nicht bereitstellen soll (weil ein anderer Server im lokalen Netz diese Aufgabe übernimmt und auch für die opsi-clients gepflegt wird), so wird das Backend dhcpd nicht benötigt:

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

Nach Anpassung der Backendkonfiguration muss die Konfiguration initialisiert und der opsiconfd neu gestartet werden:

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

4.4 Konfiguration der Namensauflösung

Für die Installation der Software auf den Clients vor dem Login müssen die Clients wissen, wie sie den Server erreichen.

opsi kennt inzwischen auch eine Reihe von Push Funktionalitäten wie z.B. *on_demand* Installationen, Nachrichten versenden, Remote-Control Software starten, Session Informationen abrufen.

Für all diese Funktionen muss der Server die Clients erreichen können und dazu muss er die gültige IP-Nummer des Clients zu ermitteln. Wie dies am besten geschieht hängt von der konkreten Konfiguration von DNS und DHCP ab. Die Zahl der möglichen Varianten ist hier sehr groß.

Daher seien hier zwei typische Extreme aufgeführt:

1. Die Clients sind nicht im DNS und haben dynamisch zugewiesene wechselnde IP-Nummern.
2. Die IP-Nummern aller laufenden Clients lassen sich immer korrekt beim DNS abfragen.

Um den opsi-server nun an die unterschiedlichen Gegebenheiten anpassen zu können gibt es zwei Konfigurationen die Sie ändern können:

- Der Eintrag `resolveHostAddress` in der Datei `/etc/opsi/backends/hostcontrol.conf`
  Steht diese Option auf *True*, wird bei einem Verbindungsaufbau vom opsi-server zu einem opsi-client die IP-Adresse des Clients bevorzugt über die Namensauflösung ermittelt. Um die im Backend von opsi hinterlegte IP-Adresse zu bevorzugen ist die Option auf *False* zu setzen.
- Der Eintrag `update ip` in der Datei `/etc/opsi/opsiconfd.conf`
  Steht dieser Eintrag auf *yes*, so wird wann immer der opsi-server von einem Client eine IP-Adresse empfängt (z.B. bei jedem Kontakt die der Client auf nimmt) die IP-Datenbank des opsi-servers aktualisiert. Der Default ist *yes*.

Für die oben aufgeführte Variante 1, ist es wahrscheinlich schlau `resolveHostAddress` auf *False* zu setzen und `update ip` auf *yes*.

Für die oben aufgeführte Variante 2, ist die bessere Konfiguration `resolveHostAddress` auf *True* zu setzen und `update ip` auf *no*.

Welche Kombination bei Ihnen am besten passt, müssen Sie anhand Ihrer Gegebenheiten selbst ermitteln.

Wenn Sie an diesen Konfigurationen etwas geändert haben, so reloaden den opsiconfd:

```
service opsiconfd restart
```

4.5 Einspielen der minimalen opsi-Produkte

Holen Sie sich die aktuellen notwendigen opsi-Pakete im opsi-Paketformat.

Die Pakete finden Sie unter http://download.uib.de/opsi4.0/products/ in den Verzeichnissen netboot/ und localboot/, für Linux-Clients zusätzlich unter opsi-linux/.

Nach dem Download müssen Sie die Pakete auf dem Server mit dem Befehl `opsi-package-manager -i <paketname>.opsi` installiert werden

Sie können das interaktiv für jedes einzelne Paket tun. Wir empfehlen Ihnen dies automatisiert zu tun. Hierzu gibt es das Werkzeug `opsi-product-updater`, welches wie in */etc/opsi/opsi-product-updater.conf* konfiguriert, automatisch die aktuellen Pakete vom opsi Repository holt und auf dem Server installiert.

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

Sollte der `opsi-product-updater` Befehl scheitern, so muss evtl. ein Proxy in der Konfigurationsdatei eingetragen werden unter

```
[repository_uib]
proxy =
```

Bitte beachten Sie, dass OS-Installationsprodukte wie z.B. win10, nach der Installation nicht sofort einsatzbereit sind. Die Installation muss noch durch die Installationsdateien des entsprechenden Installationsmediums ergänzt werden (siehe: Abschnitt 6.4).

Wenn Sie wollen, können Sie noch weitere opsi-Produkte von download.uib.de herunterladen und auf die gleiche Weise auf Ihrem opsi-server installieren.

4.6 Einspielen / Überprüfen der Freischaltdatei

Auch wenn opsi Opensource ist, so gibt es einige Zusatz-Komponenten, die im Rahmen eines Kofinanzierungsprojektes erstellt wurden und evtl. noch nicht Opensource bzw. noch nicht kostenlos sind. Sobald die Entwicklungskosten eingenommen sind, werden auch diese Module Opensource bzw. kostenlos sein. Um bis dahin die Verwendung dieser Module den zahlenden Kunden und zu Evaluierungszwecken zu gestatten, gibt es die Freischaltdatei `/etc/opsi/modules`, welche durch eine elektronische Signatur vor unautorisierter Veränderung geschützt ist. Ist diese Datei nicht vorhanden, so funktionieren nur die *freien* Module von opsi.

Um zu Evaluierungszwecken eine zeitlich befristet gültige Freischaltdatei zu erhalten, wenden Sie sich an info@uib.de. Im Rahmen einer Beteiligung an den entsprechenden Kofinanzierungsprojekten erhalten Sie eine unbefristet gültige Freischaltdatei. Diese können Sie mit root-Rechten nach /etc/opsi kopieren.

Führen Sie danach den folgenden Befehl aus:

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

Kontrollieren Sie die Freischaltung mit einer der folgenden Methoden:

Im opsi-configed können Sie sich über den Menüpunkt Hilfe/opsi-Module den Status Ihrer Freischaltung anzeigen lassen.

Mit der Methode `backend_info` können Sie mit `opsi-admin` überprüfen, welche Module freigeschaltet sind. (Hinweis: Geben Sie die weder die Datei noch die Ausgabe dieses Befehls öffentlich weiter, zumindest nicht ohne die Signatur zu löschen).

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

Beispiel-Ausgabe:

```
{
"opsiVersion" : "4.0.1",
"modules" :
          {
          "customer" : "uib GmbH",
          "vista" : true,
          "vpn" : true,
          "license_management" : true,
          "expires" : "never",
          "valid" : true,
          "multiplex" : true,
          "signature" : "DIES-IST-KEINE-ECHTE-SIGNATUR",
          "treeview" : true,
          "mysql_backend" : true
          }
}
```

Wir weisen darauf hin, dass die modules-Datei nur für zusätzliche Funktionalitäten benötigt wird und diese nicht für den „normalen“ Betrieb von opsi benötigt wird.

4.7 Start der Management-Oberfläche (opsi-configed)

Opsi bietet mit dem opsi-configed ein komfortables Management Interface.

Sie können es auf mehrere Weisen starten:

- Rufen Sie in einem Browser die folgende Adresse auf: `https://<opsidepotserver>:4447/configed.jnlp`
  Das Management-Interface wird über Java Web Start geladen.
  Damit es funktioniert, muss auf dem aufrufenden Rechner Java in Version 1.7 oder neuer installiert sein.
- Alternativ können Sie auf der graphischen Oberfläche Ihres opsi-servers mit Klick auf die rechte Taste das Kontextmenü öffnen und *opsi config editor* auswählen.
  In diesem Fall muß auf dem opsi-Server die Java Laufzeitumgebung installiert sein.

Die Bedienung des Management-Interfaces ist weitgehend selbsterklärend. Daher hier nur ein Hinweis: Änderungen im opsi-Management Interface müssen gespeichert werden, bevor Sie wirksam werden und Veränderungen der Daten müssen vom Server über *Daten neu laden* geholt werden.

Sie finden eine ausführliche Anleitung im opsi-Handbuch.

4.8 Von opsi benötigte Netzwerk-Ports

Dies ist eine Übersicht über verwendete Ports und Netzwerk-Protokolle.

- opsi-server Webservice: TCP 4447
  Client zum Server, Depot zum Server (bidirektional, Verbindungen via localhost).

- opsi-client Webservice: TCP 4441
  Server zum Client, Verbindung vom Client an sich selbst via localhost.
- opsi-client Webservice: TCP 4442
  Verbindung vom Client an sich selbst via localhost.
- opsi-client Notifier: TCP 45000 - 65536
  Verbindung vom Client an sich selbst via localhost.
  Ein zufälliger Port aus dem gegebenen Bereich wird ausgewählt.
- TFTP: UDP 69
  Client zu Server.
- CIFS/SMB: UDP 137 / UDP 138 (netbios) / TCP 139 / TCP 445
  Client zu Server (bidirektional).
  Abhängig von der Version des Client-Betriebssystems.
- WEBDAV: TCP 80
- WINEXE: TCP 139
- SSH (optional): TCP 22
- DNS: TCP 53
- WakeOnLan (WOL): UDP 12287
  Server zum Client.
- HTTP: TCP 80
  Z.B. Um Server updates von http://download.opensuse.org/ zu laden
- HTTPS: TCP 443
  Um updates von https://download.uib.de zu laden (opsi-package-updater)

Kapitel 5

Integration vorhandener Windows-Clients in opsi

Um vorhandene Windows-Clients in opsi aufzunehmen, muss auf diesen der opsi-client-agent installiert werden. Dies kann auf mehrere Arten durchgeführt werden. Nach dem Sie wie im Folgenden beschrieben den opsi-client-agent installiert haben, erscheint der Client auch in der Client-Liste des opsi-configed.

Danach können vorhandene opsi-Produkte auf diesen Clients installiert werden.

5.1 Installation des opsi-client-agent

5.1.1 Verwendung von service_setup.cmd

Diese Methode dient zur Installation auf einzelnen Rechnern. Für ein Massen-Rollout siehe weiter unten.

5.1.1.1 service_setup.cmd auf Windows NT6

1. Loggen Sie sich mit administrativen Rechten auf dem Client ein.
2. Mounten Sie den share `\\<opsiserver>\opsi_depot` auf einen Laufwerksbuchstaben.
3. Starten Sie das Script `opsi-client-agent\service_setup.cmd`
   Starten Sie da Script nicht elevated (also rechte Maustaste: *als Administrator*) sonst kann das Script evtl. nicht gestartet werden, da eine elevated Prozess kein Zugriff auf ein Netzlaufwerk hat.
4. Das Skript kopiert die notwendigen Dateien in ein temporäres lokales Verzeichnis und startet dann zur eigentlichen Installation opsi-script (winst32.exe) elevated. Daher bekommen Sie an dieser Stelle evtl. eine UAC Anfrage.
5. Das Skript nimmt per opsi-Webservice Kontakt zum Server auf um serverseitig den Client zu erzeugen und den pckey zu erfahren. Dies erfolgt zunächst mit der in der config.ini eingetragenen user/password Kombination. Schlägt dies fehl, so erscheint ein Login-Fenster mit Service-URL (opsi-configserver), Benutzername und Passwort. Hier wird ein Benutzer benötigt, der Mitglied der Gruppe *opsiadmin* ist.

Achtung
Der Client rebootet nach der Installation.

5.1.1.2 service_setup_NT5.cmd auf Windows NT5

1. Loggen Sie sich mit administrativen Rechten auf dem Client ein.
2. Mounten Sie den share `\\<opsiserver>\opsi_depot` auf einen Laufwerksbuchstaben.
3. Starten Sie das Script `opsi-client-agent\service_setup_NT5.cmd`
4. Das Skript kopiert die notwendigen Dateien in ein temporäres lokales Verzeichnis und startet dann zur eigentlichen Installation opsi-script (winst32.exe).
5. Das Skript nimmt per opsi-Webservice Kontakt zum Server auf um serverseitig den Client zu erzeugen und den pckey zu erfahren. Dies erfolgt zunächst mit der in der config.ini eingetragenen user/password Kombination. Schlägt dies fehl, so erscheint ein Login-Fenster mit Service-URL (opsi-configserver), Benutzername und Passwort. Hier wird ein Benutzer benötigt, der Mitglied der Gruppe *opsiadmin* ist.

Achtung
Der Client rebootet nach der Installation.

5.1.2 Verwendung von opsi-deploy-client-agent

Das `opsi-deploy-client-agent` Skript verteilt den opsi-client-agent direkt vom opsi-server auf die Clients. Voraussetzung hierfür sind bei den Clients:

- ein offener c$ share
- ein offener admin$ share
- ein administrativer account

Auf dem Server muss das Programm `winexe` vorhanden sein. Eine statisch gelinkte winexe in Version 0.90 ist Teil des opsi-client-agent. Für die Installation auf Clients mit einer Windows-Version neuer als Windows 7 muss *winexe* **neuer** als Version 1.0 vorhanden sein.

Winexe ist teilweise über die Paketquellen der verwendeten Distribution verfügbar. Für einige Distributionen können Sie unter http://download.opensuse.org/repositories/home:/uibmz:/opsi:/opsi40-testing/ eine aktuellere Version finden, welche auch das Deployment zu aktuellen NT6-Clients ermöglicht. Falls Sie für Ihre Distribution keine Pakete finden oder selbst eine aktuelle Version des Programms erstellen wollen, so finden Sie weitere Informationen dazu auf der Projektseite unter http://sourceforge.net/projects/winexe/.

Das Skript erzeugt serverseitig den Client, kopiert die Installations-Dateien und Konfigurationsinformationen, wie bspw. den pckey, auf den Client und startet dort die Installation.

Das Kopieren der Installationsdateien kann auf zwei Wegen geschehen. Die erste Variante wird mittels *mount* C$ auf dem Server verfügbar machen und dort die Daten zur Installation hin kopieren. Die zweite Variante wird den c$-Share des Clients mittels *smbclient* auf dem Server einhängen und dann die Installationsdateien dorthin kopieren.

Mit dem `opsi-deploy-client-agent` Skript kann auch eine ganze Liste von Clients bearbeitet werden. Dazu können entweder beliebig viele Clients als letzter Parameter übergeben werden oder mit der Option *-f* die Clients aus einer Datei eingelesen werden. Bei der Verwendung einer Datei, muss in jeder Zeile ein Client stehen.

Das Script kann mit IP-Adressen, Hostnamen und FQDNs arbeiten. Es wird versuchen automatisch zu erkennen welche Art von Adresse übergeben wurde.

Mit dem `opsi-deploy-client-agent` Skript kann auch eine ganze List von Clients bearbeitet werden. Das Skript findet sich unter */var/lib/opsi/depot/opsi-client-agent*
Führen Sie das Script mit *root* Rechten aus. Falls das Script nicht ausführbar ist, so können Sie dieses Problem mit dem folgenden Befehl beheben:
`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-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 existierender Produkte

5.2.1 Verwendung von opsi Standard Produkten: opsi-configed

Zu den Standard Produkten gehört das Produkt `opsi-configed` welches das opsi Managementinterface als Anwendung auf einem Rechner installiert. Da diese Anwendung eine Java-Anwendung ist, wird die benötigte Java-VM gleich mit installiert.

Wählen Sie im *opsi-configed*, Modus `Client-Konfiguration`, unter dem Reiter `Clients` den betreffenden Client aus.

Wenn noch nicht geschehen, aktualisieren Sie den Datenbestand des opsi-configeds mittels `Datei/Daten neu laden` bzw. Anklicken des entsprechenden Icons.

Wechseln Sie zum Reiter `Produktkonfiguration`, klicken Sie in die Spalte `Angefordert` für das Produkt `opsi-configed`, daraufhin öffnet sich eine Liste/Dropdown-Menü und dort wählen Sie die Aktion `setup`.
Sie können beobachten, das für das Produkt `javavm` `Angefordert` ebenfalls auf `setup` wechselt.

Der Haken in der Icon-Menüleiste sollte seine Farbe auf Rot wechseln. Wenn Sie ihn anklicken, werden die neuen Einstellungen zum opsi-server übermittelt, im Anschluss ist seine Farbe wieder grün.

Booten Sie dann den Client. Er sollte jetzt den `opsi-client-agent` starten und die Produkte `javavm` und `opsi-configed` installieren. Nach Abschluß der Installation sollten sie im `Starmenü` den Punkt `opsi-configed` finden.

5.2.2 Inventarisierung mit dem localboot-Produkten hwaudit und swaudit

Wählen Sie im *opsi-configed*, Modus `Client-Konfiguration`, unter dem Reiter `Clients` den betreffenden Client aus.

Wenn noch nicht geschehen, aktualisieren Sie den Datenbestand des opsi-configeds mittels `Datei/Daten neu laden` bzw. Anklicken des entsprechenden Icons.

Wechseln Sie zum Reiter `Produktkonfiguration`, klicken Sie in die Spalte `Angefordert` für das Produkt `hwaudit`, daraufhin öffnet sich eine Liste/Dropdown-Menü und dort wählen Sie die Aktion `setup`. Wiederholen Sie das für das Produkt `swaudit`.

Der Haken in der Icon-Menüleiste sollte seine Farbe auf Rot wechseln. Wenn Sie ihn anklicken, werden die neuen Einstellungen zum opsi-server übermittelt, im Anschluss ist seine Farbe wieder grün.

Booten Sie dann den Client. Er sollte jetzt den `opsi-client-agent` starten und die Produkte `hwaudit` und `swaudit` installieren. Bei `hwaudit` und `swaudit` wird Hard- bzw Softwareinformationen erhoben und zum opsi-server übermittelt. Die gesammelten Informationen werden unter den Tabs `Hardwareinformationen` bzw. `Software-Inventur` angezeigt.

5.2.3 Hardware-Inventarisierung mit dem netboot-Produkt hwinvent

Sofern Sie bereits einen Client eingerichtet haben und das Produkt hwinvent installiert ist, können Sie bereits etwas Nützliches mit opsi tun:

Wählen Sie im opsi-configed, Modus Client-Konfiguration, unter dem Reiter Client-Auswahl den betreffenden Client aus. Wenn noch nicht geschehen, aktualisieren Sie den Datenbestand des opsi-configeds mittels Datei/Daten neu laden bzw. Anklicken des entsprechenden Icons. Wechseln Sie zum Reiter Netboot-Produkte, gehen Sie in das Feld "Anstehende Aktion" des Produkts "hwinvent" und wählen Sie in der dort angebotenen Liste die Aktion "setup". Der Haken in der Icon-Menüleiste sollte seine Farbe auf Rot wechseln. Wenn Sie ihn anklicken, werden die neuen Einstellungen zum opsi-server übermittelt, im Anschluss ist seine Farbe wieder grün.

Booten Sie dann den Client. Er sollte jetzt per PXE über das Netz ein Linux-Image ziehen, das die Hardware des PCs scannt und dann den Rechner rebootet (wenn der Rechner nicht ansonsten schon eingerichtet war, kommt im Anschluss korrekterweise die Meldung, dass auf der Platte kein Betriebssystem installiert ist).

Das Ergebnis des Hardware-Scans hat der PC zum opsi-server übermittelt. Es ist unter dem Reiter "Hardware-Informationen" einzusehen.

Anmerkung
Sollte nach dem Laden des Bootimages der Bildschirm schwarz bleiben oder die Netzwerkkarte nicht (korrekt) funktionieren, so muss für diese konkrete Hardware evtl. die Startparameter des Bootimages angepasst werden.
Dies können Sie im *opsi-configed* im Tab *Hostparameter* am Eintrag *opsi-linux-bootimage.append* tun. Details hierzu finden Sie im opsi-manual im Kapitel *Netboot Produkte*.

Kapitel 6

Installation eines neuen Windows PC über opsi (OS-Installation)

6.1 Anlegen eines neuen opsi-Clients über die Management Oberfläche

Als Client-PC eignen sich reale oder virtuelle Rechner mit mindestens 1024 MB RAM, die über eine Netzwerkkarte mit Netzwerkboot-Unterstützung verfügen: D.h., sie unterstützen das PXE-Protokoll zum Laden von Boot-Systemen via Netzwerk. Der Netzwerkboot ist ggf. im Bios-Menü zu aktivieren bzw. an die erste Stelle der Bootoptionen zu rücken.

Für einen ersten Tests empfehlen wir eine VMware-Appliance, die einen „nicht installierten Rechner“ abbildet und im VMware-Player laufen kann. Diese könne Sie z.B. bei download.uib.de herunterladen (http://download.uib.de/vmware_pxeclient.zip).

Diese virtuelle Hardware wird gut von den Windows-Standardtreibern unterstützt, wenn Sie später eine Testinstallation von Windows durchführen. Zur Installation von Windows auf neueren realen Rechnern müssen Sie möglicherweise vorab zusätzliche Treiber integrieren.

Den Client können Sie jetzt mit dem opsi-configed beim opsi-server registrieren. Wählen Sie den Menü-Punkt *OpsiClient / Neuen opsi-Client erstellen* und geben Sie ein:

- IP-Namen
- DNS Domain (falls abweichend von der Vorgabe)
- Beschreibung
- IP-Nummer (benötigt, falls kein DNS zur Namensauflösung für diesen Client verwendet werden kann)
- MAC-Adresse (benötigt, falls der opsi-server DHCP-Server ist oder PXE boot mit dem Client durchgeführt werden soll).

Nach Eingabeabschluss wird der Client dem opsi-server bekanntgemacht und gleichzeitig in der DHCP-Konfiguration als PXE-Client angelegt.

Ein Client kann auch auf der Kommandozeile per opsi-admin erzeugt werden:

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

z.B.:

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

Die Liste der eingerichteten opsi-Clients kann jederzeit im opsi-configed Modus „Client-Konfiguration“ unter dem Reiter Client-Auswahl eingesehen werden.

6.2 Hardware-Inventarisierung mit dem netboot-Produkt hwinvent

Sofern Sie bereits einen Client eingerichtet haben und das Produkt hwinvent installiert ist, können Sie bereits etwas Nützliches mit opsi tun:

Wählen Sie im opsi-configed, Modus Client-Konfiguration, unter dem Reiter Client-Auswahl den betreffenden Client aus. Wenn noch nicht geschehen, aktualisieren Sie den Datenbestand des opsi-configeds mittels Datei/Daten neu laden bzw. Anklicken des entsprechenden Icons. Wechseln Sie zum Reiter Netboot-Produkte, gehen Sie in das Feld "Anstehende Aktion" des Produkts "hwinvent" und wählen Sie in der dort angebotenen Liste die Aktion "setup". Der Haken in der Icon-Menüleiste sollte seine Farbe auf Rot wechseln. Wenn Sie ihn anklicken, werden die neuen Einstellungen zum opsi-server übermittelt und die Farbe des Hakens wechselt im Anschluss wieder zu grün.

Booten Sie dann den Client. Er sollte jetzt per PXE über das Netz ein Linux-Image ziehen, das die Hardware des PCs scannt und dann den Rechner rebootet (wenn der Rechner nicht ansonsten schon eingerichtet war, kommt im Anschluss korrekterweise die Meldung, dass auf der Platte kein Betriebssystem installiert ist).

Das Ergebnis des Hardware-Scans hat der PC zum opsi-server übermittelt. Es ist unter dem Reiter "Hardware-Informationen" zu besichtigen.

Anmerkung
Sollte nach dem Laden des bootimages der Bildschirm schwarz bleiben oder die Netzwerkkarte nicht (korrekt) funktionieren, so muss für diese konkrete Hardware evtl. die Startparameter des bootimages angepasst werden.
Dies können Sie im *opsi-configed* im Tab *Hostparameter* am Eintrag *opsi-linux-bootimage.append* tun. Details hierzu finden Sie im opsi-manual im Kapitel *Netboot Produkte*.

6.3 Anlegen eines neuen opsi-Clients mit Hilfe der opsi-client-bootcd

Auf der Downloadseite von uib finden Sie unter http://download.uib.de/opsi4.0/ verschiedene iso-Images der opsi-client-boot-cd. Laden Sie sich das neueste herunter und brennen es auf eine CD. Booten Sie den Rechner von der CD. Sie sollten dann folgendes Bild sehen:

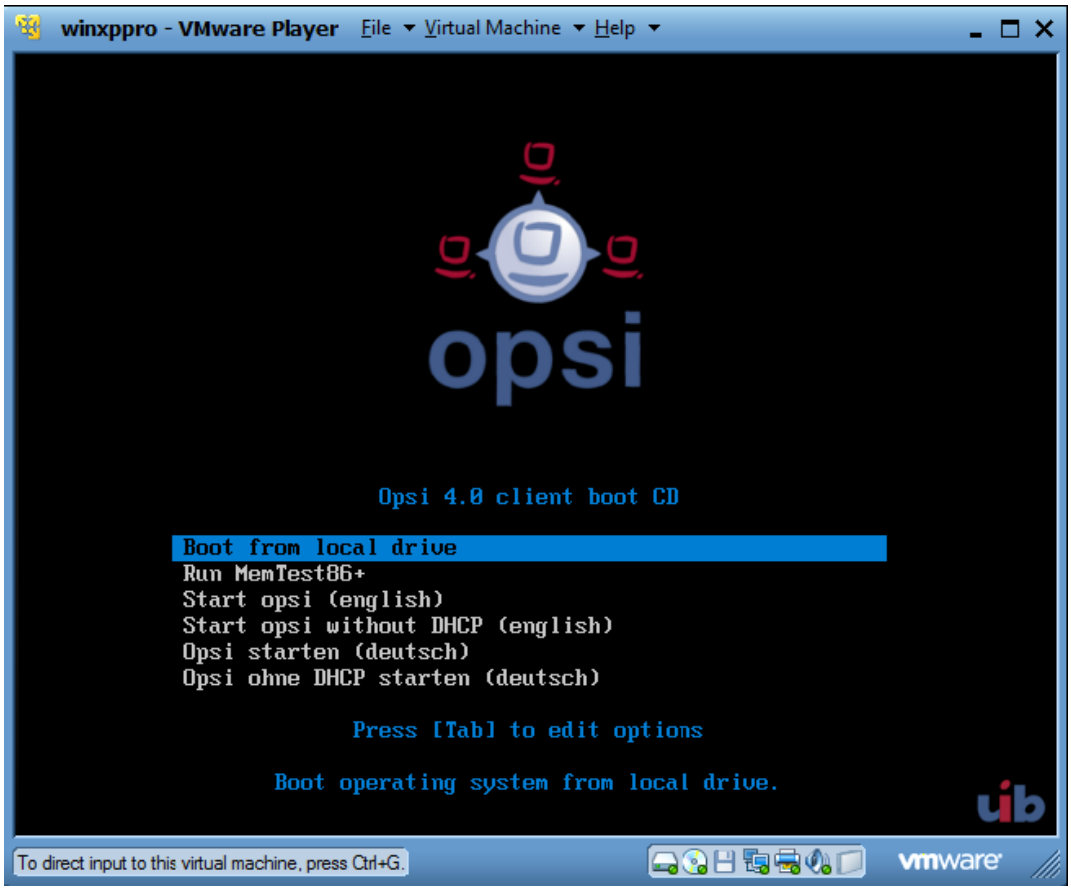

Abbildung 6.1: Startbild opsi-client-boot-cd

Wählen Sie *Opsi starten*. Nach einer Weile wird folgender Bildschirm erscheinen. Wenn Ihr DHCP-Server IP-Nummern an unbekannte Rechner vergibt ist die Maske schon weitgehend ausgefüllt. Ansonsten müssen Sie es von Hand tun. In der Regel müssen Sie mindestens den Hostnamen vergeben.

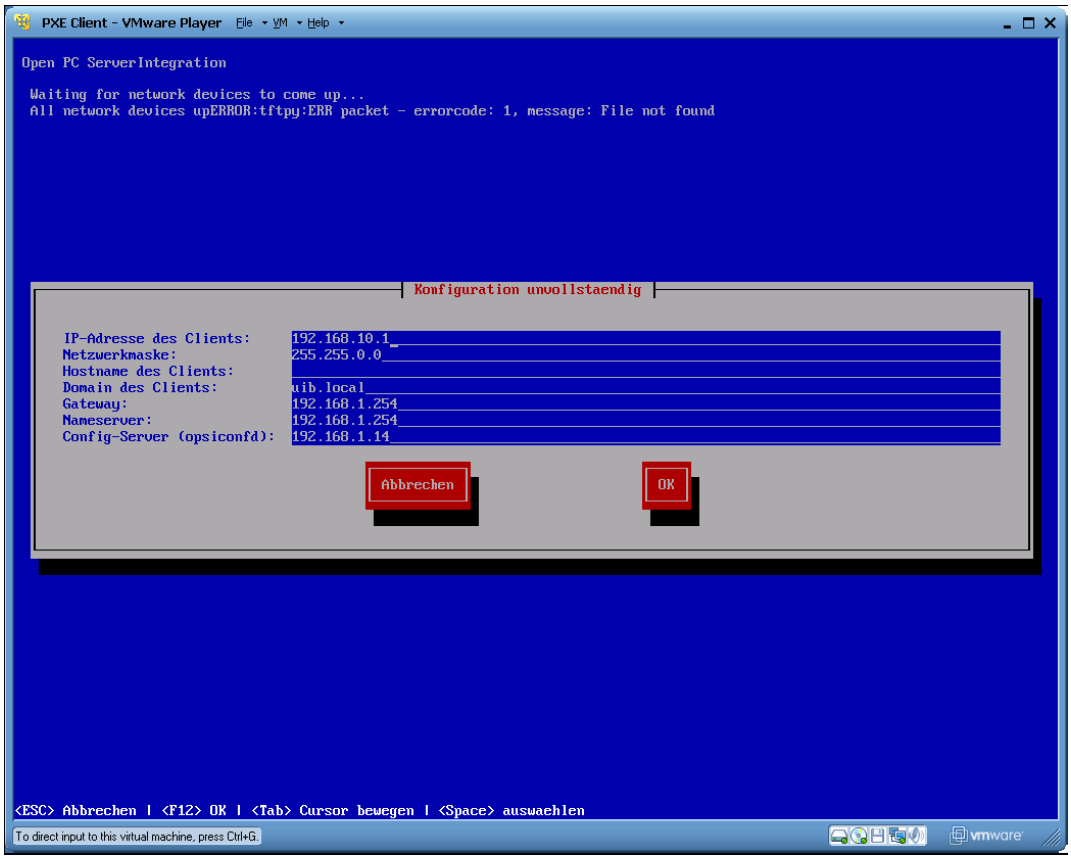

Abbildung 6.2: bootimage/boot-cd Konfigurationsmakse

Wählen Sie dann *ok*.

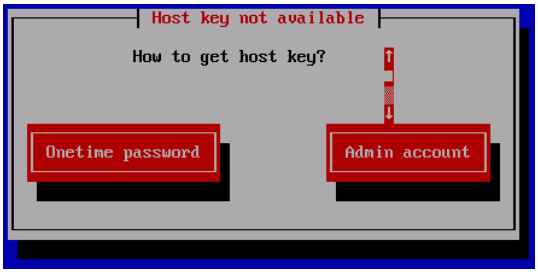

Abbildung 6.3: bootimage/boot-cd Auswahl Erstellungsmethode

Wählen Sie dann *Admin account*. Sie erklären damit, dass der Client sich selbst beim opsi-server anmelden und erstellen soll. Dieser Vorgang muss natürlich autorisiert werden.

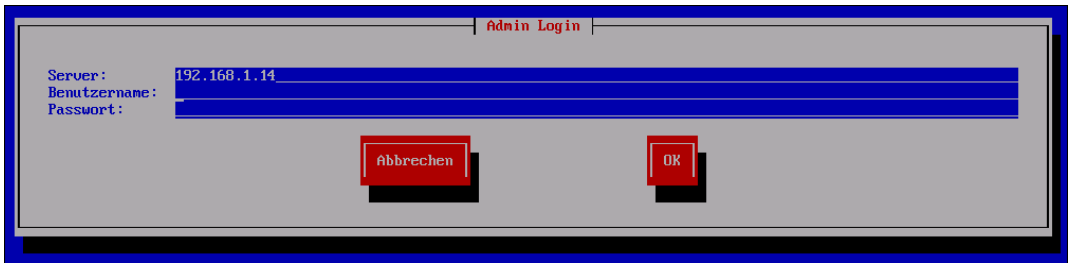

Abbildung 6.4: bootimage/boot-cd Authentifizierungsmaske

Abbildung 6.5: bootimage/boot-cd netboot Produktauswahl

Sie können jetzt direkt das zu installierende Betriebssystem (oder z.B. hwinvent) auswählen.

6.4 Betriebssysteminstallation: Vervollständigen der Basispakete für Windows

Zu den zum Download empfohlenen Paketen gehören Pakete wie win7-x64, win2012-r2, und win81-x64 / win10-x64. Diese dienen zur Installation der Windows-Betriebssysteme. Wir stellen sie nur als Basispakete (Framework) zur Verfügung, die Dateien zur Automatisierung der Betriebssysteminstallation enthalten, jedoch nicht die Dateien des Betriebssystems selbst.

Zur automatischen Windows-Betriebssysteminstallation müssen Sie Ihre vorhandenen Original-Windows-Installationsdateien kopieren (und ggf und den Windows-Lizenzschlüssel auf dem Server ablegen).

6.5 NT 6 Familie: Win10 / Win8.1 / Win7 / 2008R2

Um diese Betriebssysteme installieren zu können, wird ein sogenanntes WinPE als *Live-System* genutzt. Sie können es mit Hilfe eines opsi - Paketes erstellen (ADK 8.1,10), oder von Hand, wenn Sie möchten. Grundsätzlich ist die Windows-Version des PE unabhängig von der zu installierenden Windows - Version. Wichtig ist vor allem die Verfügbarkeit von Platten- und Netzwerkkarten - Treibern an dieser Stelle. Microsoft empfiehlt für 32-Bit Installationen ein 32-Bit PE, bzw. für 64-Bit ein 64-Bit PE.

"Um eine 64-Bit-Version von Windows zu installieren, müssen Sie eine 64-Bit-Version von Windows PE verwenden. Um eine 32-Bit-Version von Windows zu installieren, müssen Sie eine 32-Bit-Version von Windows PE verwenden."
http://technet.microsoft.com/de-de/library/cc766093.aspx

In jedem Fall benötigen Sie ein "Assessment and Deployment Kit" (ADK, Win8.1 bzw 10), bzw dessen Vorgänger "Windows Automated Installation Kit" (Windows AIK; bis Windows 7), welches Sie auf ein unterstütztes Windows Betriebssystem (bevorzugt 64 Bit) installieren.:

- Windows 10 / 8.1 ADK

Installieren Sie das ADK (nach Möglichkeit auf einer 64Bit- Maschine) in den vorgeschlagenen Pfad unter `Program Files (x86)`. Wählen Sie nur die "Windows-Vorinstallationsumgebung (Windows PE)" aus; Abhängigkeiten werden automatisch ausgewählt.
* WAIK Windows 7 Version vom 06.08.2009 in deutscher Sprache. Sie können das ISO brennen oder z.B. mit Daemon Tools mounten und dann installieren.

6.5.1 Erstellen eines PE

Die einfachste Methode setzt voraus, dass Sie bereits einen Windows - Computer haben, auf dem sowohl der opsi-client-agent installiert ist, als auch das Windows ADK (Win8.1,Win10). Der manuelle Weg wird anschließend Abschnitt 6.5.1.2 beschrieben.

6.5.1.1 Erstellen eines PE mit opsi

- Setzen Sie das localboot- Produkt `opsi-winpe` für diesen Client auf `once`, wählen Sie ggf. in den Produkt-Properties rechts unten `x86` statt `x64` aus, und speichern Sie (Rechtsklick > Speichern)
- (sofern das opsi-Produkt `opsi-winpe` nicht vorhanden ist, installieren Sie es auf dem opsi - Server mittels `opsi-product-updater -i -p opsi-winpe -vv`)
- veranlassen Sie ein Installations - Event für diesen Client (zB Rechtsklick > on-demand , oder Reboot)
- Nach erfolgreichem Lauf dieser Aktion verschieben oder kopieren Sie den Inhalt des entstandenen Verzeichnisses auf dem Client `C:\winpe_<ARCH>\media\` in das bereits existente Verzeichnis des zu installierenden Betriebssystems `\\opsiserver\opsi_depot_rw\<Betriebssystem>\winpe\`
- Führen Sie nun zuletzt folgenden Kommandozeilen - Befehl auf dem neuen opsi-Server aus. Fertig.

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

6.5.1.2 Manuelles Erstellen eines PE

Bitte die kleinen Pfad-Unterschiede zwischen WAIK und ADK beachten und der entsprechenden Sektion folgen. Die Befehle für 32- und 64-Bit sind nahezu identisch, nur müssen die Einträge **<ARCH>** mit entweder **x86** , **amd64** oder **ia64** ersetzt werden.

6.5.1.3 Anpassen des WinPE (WAIK / Windows 7)

Alles als Administrator aus der Eingabeaufforderung mit erweiterten Rechten ausführen (Start ⇒ Programme ⇒ Zubehör ⇒mit rechter Maustaste auf "Eingabeaufforderung" ⇒ Ausführen als . . . ⇒ Administrator)

- WinPE kopieren:

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

- Image mounten/einhängen:

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

- Startnet.cmd ersetzen:

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

(Hinweis: `c:\opsi\startnet.cmd` wird vom opsi-linuxbootimage in der `setup.py` erstellt nebst dem oben entfernten wpeinit-Aufruf).

- Image wieder aushängen:

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

- Verschieben des fertigen WinPE - (von dort werden mehrere Files auf den Server verschoben).

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

- Den Inhalt von `C:\winpe\ISO` nach `/var/lib/opsi/depot/win7/winpe` bzw. `/var/lib/opsi/depot/win2008/winpe` kopieren.
  Rechte anpassen (z.B.):

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

6.5.1.4 Anpassen des WinPE (ADK / Windows 8 und 10)

Führen Sie > Start > Programme > Windows Kits > Windows ADK > "Umgebung für Bereitstellungs- und Imageerstellungstools" aus; eine Eingabe-Aufforderung erscheint (in dieser sind in der Folge benötigte Umgebungsvariablen gesetzt)

- WinPE kopieren:

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

- Image mounten/einhängen:

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

- Startnet.cmd ersetzen:

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

(Hinweis: `c:\opsi\startnet.cmd` wird vom opsi-linuxbootimage in der `setup.py` erstellt nebst dem oben entfernten wpeinit-Aufruf).

- Image wieder aushängen:

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

- Den Inhalt von `C:\winpe\media` nach `/var/lib/opsi/depot/win7/winpe` bzw. `/var/lib/opsi/depot/win2008/winpe` kopieren.
  Rechte anpassen (z.B.):

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

6.5.2 Erweiterung eines PE

Manchmal ist es sehr nützlich, das Windows PE zu erweitern. Besonders bei Dell-Hardware gibt es spezielle Netzwerk- und Storage-Treiber, die speziell für den PE-Einsatz empfohlen werden. Folgende Anleitung beschreibt, wie man ein PE mit Treibern erweitern kann. Diese Anleitung funktioniert nur mit Windows 7. (Windows Vista beinhaltet nicht das benötige dism - Deployment Image Servicing and Management.) Weiterhin setzt diese Anleitung vorraus, dass die Schritte im vorigen Kapitel zum erstellen des PE ausgeführt wurden.

Anmerkung
Windows Automated Installation Kit wird für folgende Schritte nicht benötigt.

Herunterladen von Dell-PE-Treibersammlung. Bitte beachten, dass für Windows 7 die WINPE 3.0 Treiber benötigt werden. Die heruntergeladene CAB-Datei muss nun lokal ausgepackt werden. Dies kann man mit 7zip oder aber mit dem Kommandozeilentool expand erledigen. Für die Übersichtlichkeit wird empfohlen ein Verzeichnis wie zum Beispiel dell-driver auf C: an zu legen und die CAB-Datei dort aus zu packen.

- Nun wird zunächst das Image untersucht. Als Administrator die Eingabeaufforderung starten (Start ⇒ Programme ⇒ Zubehör ⇒mit rechter Maustaste auf "Eingabeaufforderung" ⇒ Ausführen als ... ⇒ Administrator) und folgenden Befehl eingeben:

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

Aus der Ausgabe des Befehls braucht man für den nächsten Schritt den Index. Da in der Regel ein winpe aus einem einzigen Image besteht, sollte in der Regel der Index 1 immer funktioneren.

- Mit folgendem Befehl wird das Image gemountet:

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

- Nun wird das PE mit den dafür ausgepackten Treibern erweitert:

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

Für 32-Bit muss das `x64` durch `x86` ersetzt werden. Die Driver-CAB von Dell beinhaltet die Treiber für beide Architekturen.

Anmerkung
Wenn man nur einen Treiber integrieren möchte, kann man die Option `/Recurse` weglassen und statt ein Verzeichniss direkt die inf-Datei des Treibers angeben. Weiterhin ist es möglich mit dem Parameter `/ForceUnsigned` auch nicht signierte Treiber in ein PE zu integrieren.

- Zum Schluss wird das Image wieder ausgehängt und die Änderungen übernommen:

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

- Das Verzeichnis `C:\winpe\ISO` als Verzeichnis `winpe` nach `/var/lib/opsi/depot/win7/` bzw. `/var/lib/opsi/depot/win2008` kopieren.
  Rechte anpassen (z.B.):

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

6.5.3 unattend.xml

Die Steuerdatei für die unattended Installation ist die `unattend.xml`, welche unter `/var/lib/opsi/depot/win7/custom` zu finden ist. Mögliche Modifikationen an dieser Datei sollten in diesem Verzeichnis und nicht im opsi Verzeichnis gemacht werden.

Die von uns mitgelieferte `unattend.xml` enthält die Aktivierung des Administrator Accounts mit dem Passwort *nt123*.

Dokumente zur Unattend.xml finden sich nach Installation des WAIK in `c:\Program Files\Windows\Waik\docs\chms`

6.5.4 Treiber-Integration

Die Treiber-Integration verläuft analog Windows 7/8.1/10: Die Treiber werden unter `/var/lib/opsi/depot/<product-id>/drivers/drivers` abgelegt. Danach wird im Ordner `/var/lib/opsi/depot/<product-id>/` das Script `./create_driver_links.py` aufgerufen.

Zu beachten ist, dass nur signierte Treiber verwendet werden können. Bei der Verwendung von Treiber-Paketen wie z.B. von driverpacks.net ist darauf zu achten, dass Sie die Pakete für Windows 7/8.1/10} verwenden.

6.5.5 Bereitstellung der Installationsmedien

Kopieren der Installations-DVD nach
`/var/lib/opsi/depot/<productid>/installfiles` und passen Sie Rechte/Eigentümer an:

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

6.5.6 Log-Dateien der unattended-Installation

- `c:\Windows\Panther\setupact.log`:
  Log bis Ende Setup-Phase 4 (läuft unter WinPE)
- `c:\Windows\Panther\setupact.err`:
  Fehler-Log bis Ende Setup-Phase 4 (läuft unter WinPE)
- `c:\Windows\Panther\UnattendGC\setupact.log`:
  Log ab Specialize-Phase
- `c:\Windows\Panther\UnattendGC\setupact.err`:
  Fehler-Log ab Specialize-Phase
- `c:\Windows\System32\Winevt\Logs\*`
- `c:\Windows\ntbtlog.txt` (nur nach aktivierter Startprotokollierung)

6.6 Windows-Produktschlüssel

Wenn Sie über das Modul opsi-Lizenzmanagement verfügen, können Sie die Windows Lizenzschlüssel über das Lizenzmanagement Modul verwalten. Lesen Sie dazu das Lizenzmanagement Handbuch bzw. das entsprechende Kapitel im opsi-Handbuch.

Haben Sie kein Lizenzmanagement oder wollen dieses nicht verwenden, gehen Sie wie folgt vor.

Wenn Sie bereits opsi-Clients eingerichtet haben, können Sie im opsi-Konfigurationseditor einen Windows-Produktschlüssel per Client eintragen:

- einen Client auswählen

- zum Tab Netboot-Produkte wechseln
- dort zB das Produkt win7-x64 auswählen
- rechts in der Schalter-Liste in die Property-Zeile productkey gehen
- in das Value-Feld den Schlüssel eintragen
- das Feld verlassen, die Änderungen speichern.

Oder Sie arbeiten auf der Kommandozeile. Mit Angabe eines opsiservers: *<opsiserver.domain.local>* werden gemeinsame Werte für alle Clients am Depot abgefragt/gesetzt:

Die vorgegebenen Werte der Produkt-Properties erfahren Sie mit (passen Sie gegebenfalls das angegebene Produkt an und tauschen Sie *<opsiserver.domain.local>* mit der FQDN Ihres opsiservers aus):

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

Um die Vorgabewerte abzuändern können Sie wie folgt vorgehen: Lesen Sie die Daten aus und schreiben Sie das Ergebnis in eine Datei. Diese kann nun verändert und das Ergebnis zurückgeschrieben werden.

Auslesen und wegschreiben (passen Sie ggf. das angegebene Produkt an und tauschen Sie *<opsiserver.domain.local>* mit der FQDN Ihres opsiservers aus):

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

Editieren Sie die entstandene Datei `/tmp/property_config.json` und ändern dabei die Einträge für *"values"* der gewünschten Properties. Die veränderte Datei lesen Sie wieder ein mit dem Befehl:

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

Kontrollieren Sie das Ergebnis mit einem Aufruf von (passen Sie ggf. das angegebene Produkt an und modifizieren Sie *<opsiserver.domain.local>* mit der FQDN Ihres opsiservers):

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

6.7 Start der Windows-Installation

Zum Starten einer Windows-Installation wählen Sie nun im opsi-configed den betreffenden Client aus, setzen unter dem Karteireiter *Netboot-Produkte* für die gewünschten Betriebssystem (z.B. win7-x64) die Aktion auf *setup* und klicken auf den roten Haken (der wieder grün wird).

Der Client sollte jetzt beim Booten ein Linux Bootimage übers Netz ziehen, in dem Sie nochmal die PC-Neu-Installation bestätigen müssen. Dann sollte alles automatisch weiter laufen, bis schließlich die Logon-Aufforderung des installierten Windows auf dem Bildschirm steht.

Anmerkung
Sollte nach dem Laden des Bootimages der Bildschirm schwarz bleiben oder die Netzwerkkarte nicht (korrekt) funktionieren, so muss für diese konkrete Hardware evtl. die Startparameter des Bootimages angepasst werden.
Dies können Sie im *opsi-configed* im Tab *Hostparameter* am Eintrag *opsi-linux-bootimage.append* tun. Details hierzu finden Sie im opsi-manual im Kapitel *Netboot Produkte*.

Achtung
Vorsicht bei Clients mit einer über 2TB großen Festplatte. In einem nicht UEFI-System beträgt die maximale Partitionsgröße 2 Terabyte. Sofern eine größere Partition angelegt werden soll schlägt die Installation fehl. Dies ist technisch durch die standard Partitionstabelle bedingt. Sie müssen die Festplatte in zwei Partitionen aufteilen. Dies können Sie über die Product-Properties steuern. Oder Sie erwerben das UEFI-Modul, wodurch diese technische Limietierung entfällt.

6.8 Aufbau der Produkte zur unattended Installation

Diese Kapitel betrifft die Windows-Netboot-Produkte.

6.8.1 Übersicht des Verzeichnisbaums

```
<productid>-
          |-i386/                              NT5 only: Installations files
          |-installfiles/                      NT6 only: Installations files
          |-winpe/                             NT6 only
          |-opsi/                              Scripte und Templates by opsi.org
          |  |-$oem$/                                 NT5 only: $oem$ gemaess MS
          |  |-postinst.d/                            Scripte nach OS-Install by opsi.org
          |  !-unattend.(txt/xml).template            Template by opsi.org
          |-custom/                            Scripte und Templates by customer
          |  |-$oem$/                                 NT5 only: $oem$ gemaess MS by customer
          |  |-postinst.d/                            Scripte nach OS-Install by customer
          |  !-unattend.(txt/xml)                     unattend.txt by customer
          |-drivers/                           drivers Verzeichnis
       user   |  |-drivers/                     drivers Verzeichnis
          |  |-pciids/                         Symlinkbaum zu Treibern
          |  |-vendors/                        Symlinkbaum zu Treibern
          |  |-classes/                        Symlinkbaum zu Treibern
          |  |-usbids/                         Symlinkbaum zu Treibern
          |  |-hdaudioids/                     Symlinkbaum zu Treibern
          |  |-pci.ids                         PCI-IDs DB
          |  !-usb.ids                         USB-IDs DB
          |-setup.py                           Installationsscript
          |-<productid>_<version>.control      Meta Daten (nur zur Info)
          |-<productid>.files                  Dateiliste (automatisch erstellt)
          |-create_driver_links.py             Script zur Treiberverwaltung
          |-show_drivers.py                    Script zur Treiberverwaltung
          !-extract_driver_pack.py             Script zur Treiberverwaltung
```

6.8.2 Die Dateien

- `setup.py`
  Dies ist das Installationsscript, welches vom Bootimage ausgeführt wird.
- `<productid>_<version>.control`
  enthält die Metadaten des Produkts, so wie sie vom Paketierer bereitgestellt wurden. Die Datei liegt hier nur zu Informationszwecken, d.h. Änderungen an dieser Datei haben keinerlei Auswirkungen auf das System.
- `<productid>.files`
  Diese Datei wird automatisch erzeugt und sollte nicht verändert werden.
- `create_driver_links.py`
  `show_drivers.py`
  `extract_driver_pack.py`
  Dies sind Scripte zur Treiberintegration, die im Kapitel Vereinfachte Treiberintegration in die automatische Windowsinstallation, näher erläutert werden.

6.8.3 Verzeichnis installfiles / winpe

- `installfiles`
  Enthält ab Windows7 (NT 6.x / 7) den Inhalt der Installations-CD.
- `winpe`
  Enthält ab Windows7 (NT 6.x / 7) ein bootbares winpe Image.

6.8.4 Verzeichnis opsi / custom

Diese beiden Verzeichnisse enthalten Scripte und Konfigurationsdateien zur Steuerung der Betriebssysteminstallation. Während der Installation wirken diese Verzeichnisse zusammen, indem die Dateien aus custom Vorrang haben.

Das Verzeichnis opsi enthält Dateien, die mit Updates jederzeit überspielt werden können. Hier sollten also keine Änderungen vorgenommen werden. Für Anpassungen können Sie Änderungen im Verzeichnis custom vornehmen, welches bei Updates nicht überspielt wird.

Das Unterverzeichnis `postinst.d` enthält Scripte, welche nach der eigentlichen Installation des Betriebssystems über die `postinst.cmd` gestartet werden, um z.B. den opsi-client-agent zu installieren. Die Scripte werden dabei in alphabetischer Reihenfolge abgearbeitet. Um die Reihenfolge zu verdeutlichen, fangen die Dateinamen mit einer zweistelligen Nummer an (`10_dhcp.cmd`). Wollen Sie hier Erweiterungen vornehmen, so können Sie im Verzeichnis custom/postinst.d Scripte mit Nummern zwischen den vollen 10ern ablegen (`13_myscript.cmd`). Die vollen 10er sind für die Pflege durch opsi.org/uib reserviert. Das Script `99_cleanup.cmd` ist das letzte und endet mit einem Reboot.

6.8.5 Verzeichnis drivers

Dieses Verzeichnis dient der Treiberintegration und ist im folgenden Kapitel beschrieben.

6.9 Vereinfachte Treiberintegration in die automatische Windowsinstallation

Administriert man einen Pool von PCs, die Geräte besitzen, deren Treiber nicht in der Windows-Standardinstallation enthalten sind, so ist es meist sinnvoll, diese Treiber direkt in die Installation zu integrieren. Bei Netzwerkgeräten kann dies teilweise sogar unumgänglich sein, denn ein startendes Windows ohne Netzwerkkarte ist für den Administrator nicht ohne weiteres erreichbar.

Opsi unterstützt Sie durch eine Automatisierung der Treibereinbindung und vereinfacht so die Bereitstellung der Treiber. Dabei müssen die Treiber nur in dem korrekten Verzeichnis abgelegt werden. Durch den Aufruf eines Scripts werden dann die Treiberverzeichnisse durchsucht und ein Katalog erstellt, anhand dessen das Bootimage automatisch die richtigen Treiber erkennen und einbinden kann. Dabei können sowohl Standard-Treiber, USB-Treiber, HD-Audio-Treiber wie auch Treiber für Festplattencontroller (Textmode Treiber) abgelegt und automatisch eingebunden werden.

Damit die Treiber sofort bei der Windowsinstallation mit installiert werden, müssen Sie in einer bestimmten Form auf dem Server hinterlegt werden. Hierzu sind Treiberverzeichnisse geeignet, die eine **.inf*-Datei enthalten, die den Treiber für das Windows-Setupprogramm beschreibt. Irgendwelche in `setup.exe`, **.zip* oder anders verpackten Treiber sind hier unbrauchbar. Mit dem Programm *double driver* (http://www.boozet.org/dd.htm) können Sie von einem installierten Rechner die Treiber im geeigneten Format extrahieren.

Es stehen mehrere Ebenen zur Bereitstellung von Treibern zur Verfügung:

- Allgemeine Treiber Pakete
- Treiber die zu Ihrer Hardware gehören aber nicht speziell zu geordnet sind
- Treiber die manuell Rechnern zu geordnet sind
- Treiber die über die Felder <vendor>/<model> der Inventarisierung automatisch den Rechnern zu geordnet werden.

Wie diese unterschiedlichen Ebenen verwendet werden können ist im folgenden beschrieben:

6.9.1 Allgemeine Treiber Pakete

Wenn die Hardwareausstattung sehr heterogen ist, kann es sinnvoll sein mit allgemeinen Treiberpaketen zu arbeiten. Allgemeine Treiber legen Sie ab unter `./drivers/drivers`.
Solche allg. Treiber Pakete finden Sie unter DriverPacks.net.
Laden Sie die gewünschten Treiber Pakete in ein temporäres Verzeichnis herunter und entpacken die Treiberpakete mit:

```
./extract_driver_pack.py <pfad zu dem temporären Verzeichnis mit den komprimierten driverpacks>
```

Hiermit werden die Treiber entpackt und in das Verzeichnis `./drivers/drivers/` abgelegt.
Nachteil dieser Pakete ist, das sich hier auch Treiber finden welche zwar von der Beschreibung zu Ihrer Hardware passen aber nicht unbedingt mit Ihrer Hardware funktionieren.
Treiber welche im Verzeichnis `./drivers/drivers/` liegen, werden anhand der PCI-Kennungen (bzw. USB- oder HD_Audio-Kennung) in der Beschreibungsdatei des Treibers als zur Hardware passend erkannt und in das Windows Setup mit eingebunden.

6.9.2 Treiber die zu Ihrer Hardware gehören aber nicht speziell zu geordnet sind

Haben Sie nur wenige unterschiedliche Hardware zu unterstützen, so können Sie die Treiber bei den Herstellern suchen.
Zusätzliche bzw. geprüfte Treiber gehören in jeweils eigene Verzeichnisse (Name und Tiefe der Verzeichnisstruktur egal) unterhalb des Verzeichnisses
`./drivers/drivers/preferred`.
Treiber welche im Verzeichnis `./drivers/drivers/preferred` liegen, werden gegenüber den Treibern in `./drivers/drivers/` bevorzugt anhand der PCI-Kennungen (bzw. USB- oder HD_Audio-Kennung) in der Beschreibungsdatei des Treibers als zur Hardware passend erkannt und in das Windows Setup mit eingebunden.
Finden sich z.B. zu ein und derselben PCI-ID unterschiedliche Treiber unter `preferred`, so kann dies zu Problemen bei der Treiber Zuordnung führen. In diesem Fall ist eine direkte Zuordnung der Treiber zu den Geräten notwendig.

6.9.3 Treiber die manuell Rechnern zu geordnet sind

Zusätzliche Treiber, die unabhängig von ihrer Zuordnung bzw. Erkennung über die PCI- oder USB-IDs installiert werden sollen, gehören in jeweils eigene Verzeichnisse (Name und Tiefe der Verzeichnisstruktur egal) unterhalb des Verzeichnisses `./drivers/drivers/additional`. Über das Produkt-Property *additional_drivers* von können Sie einen oder mehrere Pfade von Treiberverzeichnissen innerhalb von `./drivers/drivers/additional` einem Client zu ordnen. Im Produkt-Property *additional_drivers* angegebene Verzeichnisse werden rekursiv durchsucht und alle enthaltenen Treiber eingebunden. Dabei wird auch symbolischen Links gefolgt. Dies können Sie nutzen, um für bestimmte Rechner-Typen ein Verzeichnis zu erstellen (z.B. dell-optiplex-815).

Wird in den über *additional_drivers* angegebenen Treiberverzeichnissen ein Treiber für ein vorhandenes PCI-Gerät (oder HD-Audio, USB) gefunden, so wird für dieses Gerät kein weiterer Treiber aus `drivers/preferred/` oder `drivers/` mehr eingebunden. Damit hat *additional_drivers* nicht nur die Funktion Treiber hinzuzufügen, welche über die normale Treibererkennung nicht gefunden würden. Darüberhinaus haben die Treiber welche dem client via *additional_drivers* zugeordnet werden auch Vorrang vor Treibern aus anderen Verzeichnissen (*additional_drivers* ist sozusagen auch *super-preferred*).

6.9.4 Treiber die über die Felder <vendor>/<model> der Inventarisierung automatisch den Rechnern zu geordnet werden.

Der im vorigen Abschnitt beschriebene Mechanismus der direkten zuordnung von Treibern zu Geräten, kann seit dem 2 Teil des Service Release opsi 4.0.2 automatisiert werden. Dazu wird in dem Verzeichnis `./drivers/drivers/additional/byAudit` nach einem Verzeichnisnamen gesucht der dem bei der Hardwareinventarisierung gefundenen *Vendor* entspricht. In diesem *Vendor* Verzeichnis wird nun nach einem Verzeichnisnamen gesucht, das dem bei der Hardwareinventarisierung gefundenen *Model* entspricht. Wird ein solche Verzeichnis gefunden, so wird diese Verzeichnis genauso behandelt, als wären sie über das Produktproperty *additional_drivers* manuell zugewiesen.

Dabei können seit opsi 4.0.5 die Treiber für einen opsi-client über den opsi-configed im Reiter Hardwareinventarisierung bereitgestellt werden (vgl.:opsi-Handbuch Automatisierte Treiberintegration).

Die Abarbeitung im opsi-linux-bootimage erfolgt in der Reihenfolge

- `<vendor>/<model> (<sku>)`

- ohne Fund greift der Fallback auf `<vendor>/<model>`
- ohne Fund greift der Fallback auf die Hauptplatine `<vendor>/<motherboard-model>`.

Einige Hersteller verwenden Modellbezeichnungen, die für diese Methode sehr ungünstig sind, da man einige Sonderzeichen wie / nicht in Datei- oder Verzeichnisnamen verwenden darf. Ein Beispiel dafür wäre als Modelbezeichnung: "5000/6000/7000". Ein Verzeichnis mit dieser Bezeichnung ist wegen den Sonderzeichen nicht gestattet. Seit der dritten Service Release opsi 4.0.3 werden deshalb folgende Sonderzeichen: < > ? " : | \ / * intern durch ein _ ersetzt. Mit dieser Änderung kann man oben genanntes schlechtes Beispiel als: "5000_6000_7000" anlegen und das Verzeichnis wird automatisch zu gewiesen, obwohl die Information in der Hardwareinventarisierung nicht der Verzeichnisstruktur enstprechen.

6.9.5 Struktur des Treiber Verzeichnisses und Ablage der Treiber:

```
/var/
  !-lib/
     !-opsi/depot/
        !-<productid>/
           !-drivers
              |-classes/               (Links auf Treiber über Geräteklassen)
              |-hdaudioids/            (Links auf HD-Audio Treiber)
              |-pciids/                (Links auf Treiber über PCI-Kennung)
              |-pci.ids                (PCI Datenbank)
              |-usbids/                (Links auf Treiber über USB-Kennung)
              |-usb.ids                (USB Datenbank)
              |-vendors/               (Links auf Treiber über Hersteller)
              !-drivers                (Platz für allg. Treiber Packs)
                 |-additional/         (Für manuell zugeordnete Treiber)
                    |-byAudit/         Modell spezifische Treiber welche
                       |-<vendor>              über die Hardwareinventarisierung
                          |-<model>            zugeordnet werden
                 |-buildin/            (Daten aus dem i386 Baum)
                 |-preferred/          (geprüfte Treiber)
                 |-exclude/            (ausgeschlossene Treiber)
                 !-mydriverpacks/      (Beispiel Treiber Pack)
```

6.9.6 Abarbeitung der unterschiedlichen Ebenen der Treiberintegration

Als oberste Priorität werden alle Treiber eingebunden, welche über das Property *additional_drivers* bzw. die über die Inventarisierungsdaten in `./drivers/drivers/additional/byAudit` gefunden werden. Im Rahmen der Einbindung von Treibern wird geprüft für welche der Hardware eines Geräts (anhand der PCI-,USB-,HDAudio-Kennungen) hierdurch ein Treiber bereit gestellt wurde. Nur für Geräte für die auf diese Weise noch kein Treiber bereitgestellt wurde wird über die nachfolgenden Methode ein Treiber gesucht.

Für Geräte denen nicht über *additional_drivers* (bzw. *byAudit*) ein Treiber zu geordnet wurde wird anhand der PCI Kennung (bzw. USB-, HDAudio-Kennung) ein passender Treiber gesucht und eingebunden.

Einbindung von Treiber bedeutet dabei:

- Der Treiber wird auf die lokale Festplatte nach `c:\drv\<num>` kopiert.
- Dem Windows Setup wird in der unattended Datei mitgeteilt, in den Verzeichnissen unterhalb von `c:\drv\` nach passenden Treibern zu suchen.

6.9.7 Treiber hinzufügen und prüfen

Nach jedem Hinzufügen eines Treibers oder jeden anderen Änderung im `./drivers/drivers` Verzeichnis (oder darunter) rufen Sie im Stammverzeichnis des netboot Produktes Verzeichnis folgenden Befehl auf, um die Rechte korrekt zu setzen:

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

Danach rufen Sie das Script `./create_driver_links.py` auf. Dieses durchsucht die Verzeichnisse unterhalb von *./drivers/drivers* und erzeugt eine Reihe von Links anhand deren die Zuordnung der Treiber zu bestimmter Hardware (PCI-IDs, USB-IDs, HD-Audio-IDs) zu erkennen ist. Die Treiber aus dem preferred Verzeichnis werden von dem Script bevorzugt verwendet.

Das `setup.py` Script des Bootimages untersucht die Hardware des zu installierenden Computers und identifiziert die notwendigen Treiber. Diese werden dann auf die Platte kopiert und die unattend.txt entsprechend gepatcht. Das Script `create_driver_links.py` durchsucht auch bei NT5 Produkten einmalig den *i386* Baum und extrahiert die Inf-Dateien der von Windows mitgelieferten Treiber nach *windows_builtin*. Sollten Sie am i386-Baum eine Änderung vornehmen (z.B. durch das Einspielen eines Servicepacks) so löschen Sie dieses Verzeichnis und führen `create_driver_links.py` erneut aus. Bei NT6 Produkten werden die Treiber welche sich im WinPE finden als *windows_builtin* erkannt.

Liegt zu einem Client eine Hardware-Inventarisierung vor, so kann über den Befehl:

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

ausgegeben werden, welche Treiber das Bootimage via PCI-IDs, USB-IDs, HD-Audio-IDs und *additional_drivers* (bzw. *byAudit*) zur Installation auswählen würde und zu welcher Hardware noch kein Treiber bereit steht.

Kontrollieren Sie die Ausgabe von `show_drivers.py` um zu prüfen ob die gewünschten Treiber eingebunden werden.

Es kann vorkommen, das Treiberverzeichnisse von Herstellern Treiber für unterschiedliche Betriebssystemversionen (z.B. Windows 7/8.1/10) oder Konfigurationen (SATA / SATA-Raid) enthalten. Das `create_driver_links.py` script kann das nicht unterscheiden. Wenn Sie die Vermutung haben, das ein verlinkter Treiber falsch ist, so verschieben Sie diesen Treiber in das Verzeichnis `drivers/exclude` und führen `create_driver_links.py` erneut aus. Treiber die in `drivers/exclude` liegen werden bei der Treiberintegration nicht berücksichtigt.

Beispiel einer `show_drivers.py` Ausgabe:

```
./show_drivers.py pcdummy

PCI-Devices
   [(Standardsystemgeräte), PCI Standard-PCI-zu-PCI-Brücke]
      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-Verbundgerät]
      /var/lib/opsi/depot/<productid>/drivers/drivers/preferred/brother_844x_pGerb
   [Microsoft, USB-Druckerunterstützung]
      /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
```

```
 ./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-Brücke - 244E
      Using build-in windows driver
      This driver will not be integrated, because same device already integrated in: '/var/lib/opsi/depot/<productid>n/\
    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]  (Standardsystemgeräte) : USB-Eingabegerät
      No driver - vendor directory '/var/lib/opsi/depot/<productid>/drivers/usbids/0461' not found
   [0461:4D20]  (Standardsystemgeräte) : USB-Eingabegerät
      No driver - vendor directory '/var/lib/opsi/depot/<productid>/drivers/usbids/0461' not found
   [058F:6366]  Kompatibles USB-Speichergerät : USB-Massenspeichergerät
      No driver - vendor directory '/var/lib/opsi/depot/<productid>/drivers/usbids/058F' not found
   [0461:0010]  (Standard-USB-Hostcontroller) : USB-Verbundgerät
      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
```

TIPPS

- Treiberverzeichnisname `NDIS1` sind Vista-Treiber ; `NDIS2` sind Win7-Treiber
- Manche Chipsatztreiber enthalten Beschreibungsdateien, welche sehr viel Hardware aufführen ohne hierzu tatsächlich Treiber zu liefern. Ein Beispiel hierfür ist z.B. die `cougar.inf` oder `ibexahci.inf` von Intel.

Wird ein solches *Pseudo-Treiber* Verzeichnis per *additional_drivers* (bzw. *byAudit*) zu geordnet, so führt dies dazu, das die hier aufgeführte Hardware von der weiteren suche nach Treibern im *preferred* Verzeichnis ausgeschlossen wird.

- SATA-Treiber und SATA-RAID Treiber beziehen sich auf die selbe PCI-Kennung. Ein SATA-RAID Treiber wird aber mit einem einzelplatten System nicht funktionieren.
- Kontrollieren Sie die Ausgabe von `./show_drivers.py` genau !

Kapitel 7

Einbindung eigener Software in die Softwareverteilung von opsi

Die Installation von Software erfolgt bei opsi durch den opsi-Clientagenten und insbesondere durch das Script gesteuerte Setup Programm opsi-winst. Daher muss zu jedem opsi-Produkt ein opsi-winst-Script erstellt werden. Danach werden dieses Script, die Installationsdateien und die Metadaten zu einem opsi-Produkt gepackt, welches sich schließlich auf dem opsi-server installieren lässt.

7.1 Ein kleines Tutorial zur Erstellung eines opsi-winst Scriptes

7.1.1 Einführung

Dieses Tutorial kann keine Schulung oder das Studium der Handbücher ersetzen. Es dient nur dazu eine Einführung zu bekommen. Daher als erstes der Verweis auf weiterführende Quellen:

Schulungen: Die uib GmbH bietet opsi-Schulungen in Mainz und Inhouse Schulungen an:
http://uib.de/de/support-schulung/schulung/

Handbücher: http://uib.de/de/opsi-dokumentation/dokumentationen/
Besonders wichtig:
opsi-winst-Reference-Card und opsi-winst-Handbuch

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

Support Forum: siehe http://forum.opsi.org

7.1.2 Methoden der nicht interaktiven Softwareinstallation

Prinzipiell gibt es drei Verfahren der Einbindung eines Softwarepakets in die automatische Softwareverteilung für Windows-Betriebssysteme, zuzüglich einer Variante, die sich auf die Pakete für den Microsoft Installer Service bezieht.

1. **Unattended / Silent Setup**:
   Das Original-Setupprogramm wird verwendet und über Kommandozeilenargumente in einen nicht-interaktiven Modus versetzt. Der wichtigste Spezialfall davon ist der
   „stille“ Aufruf eines MSI-Pakets:
   Ein Paket für den Microsoft Installer Service ist vorhanden und wird mit einer „quiet“-Option aufgerufen.
2. **Interaktives Setup mit automatisierten Antworten**:
   Zur Vorbereitung wird bei einem Lauf des Original-Setupprogramms festgestellt, welche Fenstertitel das Programm zeigt und welche Fragen und Antworten beim Setup anfallen. Dies wird in einem Skript niedergeschrieben. Im Prozess der Softwareverteilung läuft das Setupprogramm dann unter Kontrolle eines Automatisierungs-Programms wie z.B. AutoIt oder Autohotkey, welches das Setupprogramm gemäß dem Skript steuert.

3. **Analysieren und Neu-Paketieren**:
   Es wird (teil-automatisiert) untersucht, welche Komponenten auf einem Test-PC, auf dem nur das Betriebssystem bzw. allgemeine Basissoftware verfügbar ist, installiert werden müssen, damit die Software wie gewünscht läuft. Diese Analyse dient als Basis, um ein neues Verteilungspaket zu bauen. Das Paket kann dabei direkt mit opsi-winst-Mitteln erstellt werden. Es kann aber auch als MSI-Paket ausgeführt werden, das dann in einen beliebigen Verteilungsmechanismus eingebunden werden kann.

Anmerkung
Opsi unterstützt alle drei Varianten. In der Praxis werden sie häufig ergänzend verwendet.

7.1.3 Struktur eines opsi-script / opsi-winst-Skripts

Zunächst ein Beispiel für ein einfaches opsi-winst-Skript:

```
[Actions]
WinBatch_tightvnc_silent_install

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

Ein opsi-winst-Skript besteht aus **primären** und **sekundären** Sektionen. Sektionen werden, wie von ini-Dateien bekannt, mit einem Sektions-Namen in eckigen Klammern eingeleitet.
Die eigentlichen Arbeiten zur Software-Installation finden in den sekundären Sektionen statt, die von den primären Sektionen aufgerufen werden.

Die sekundären Sektionen sind „Themen-spezifisch“ und verfügen jeweils über eine spezielle Syntax.
Der Sektionsname einer sekundären Sektion beginnt mit deren Typ, gefolgt von einem frei definierbaren Namen.

Im Beispiel ruft die primären Sektion `[Actions]` eine sekundäre Sektion `[WinBatch_tightvnc_silent_install]` auf. Die sekundäre Sektion ist vom Typ `WinBatch`. Der Inhalt einer `WinBatch`-Sektion wird über die Windows-API ausgeführt.
In diesem Fall wird also das Setup-Programm `tightvnc-1.3.9-setup.exe` mit dem Parameter `/silent` gestartet.

7.1.4 Primäre Sektionen:

Actions/Aktionen
Die `[Actions]` Sektion ist das eigentliche Hauptprogramm. Hier beginnt die Skript-Verarbeitung.

Sub-Sektionen
Programmabschnitte, die wiederholt benötigt werden, können in Sub-Sektionen (Unterprogramme) ausgelagert werden. Es besteht die Möglichkeit Sub-Sektionen in externe Dateien auszulagern.

Die primären Sektionen sind das Hauptprogramm in dem der Ablauf des Skripts gesteuert wird. Hierzu gibt es:

- Variablen: Strings und Stringlisten
- `if else endif` Anweisungen
- `for` Schleifen über Stringlisten
- Funktionen

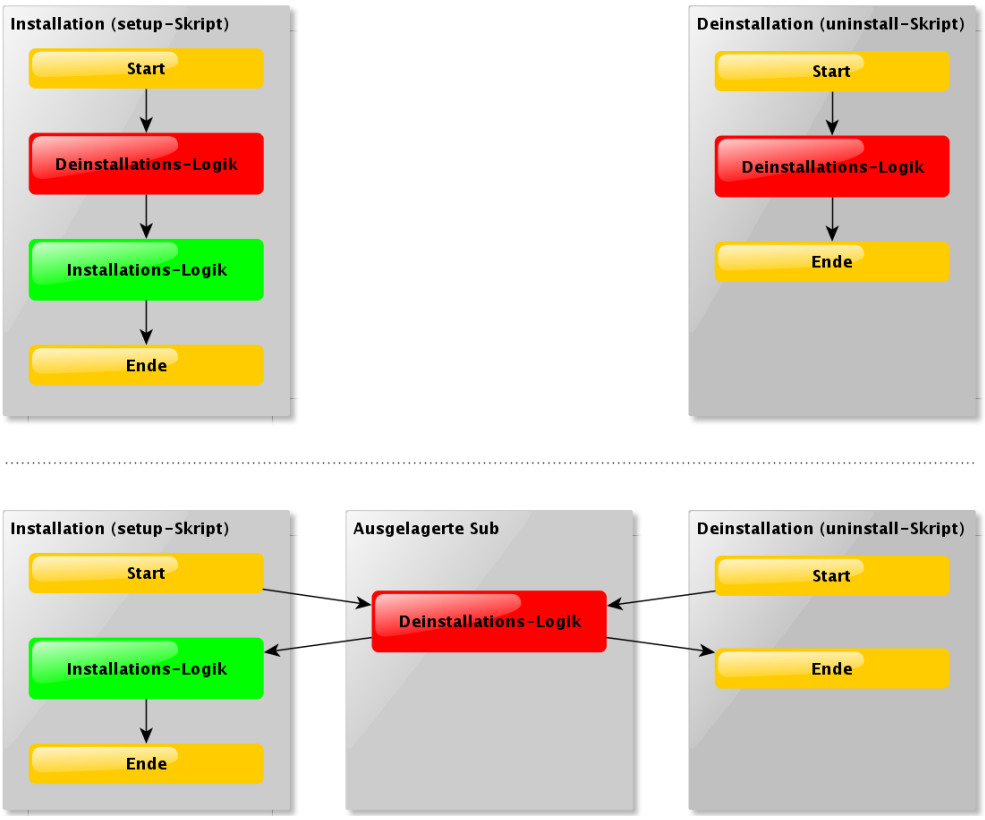

Abbildung 7.1: Vermeidung doppelten Codes über ausgegliederte Sub

7.1.5 Wichtige sekundäre Sektionen:

Files
Datei-Operationen, wie:

- kopieren (mit Versionskontrolle, rekursiv ...)
- löschen
- Verzeichnisse anlegen
- ...

WinBatch
Dient zum Aufrufen von Programmen über die Windows-API. Beispielsweise werden Aufrufe von Setup-Programmen im silent mode in diesen Sektionen durchgeführt.

DosBatch/DosInAnIcon
Der Inhalt dieser Sektionen wird der *cmd.exe* zur Ausführung übergeben. Hier können also normale Batch-Skripte abgelegt werden.
Eine Variante von *DosBatch* ist *DosInAnIcon*, wobei die *cmd.exe* mit minimiertem Fenster aufgerufen wird.

ExecWith
Der Inhalt dieser Sektionen wird einem externen Programm (Interpreter) zur Ausführung übergeben. Beispielsweise können über *ExecWith* AutoIt-Skripte http://www.autoitscript.com direkt in das opsi-winst-Skript integriert werden.

Registry
Die *Registry-Sektionen* dienen dem Bearbeiten der Registry.

LinkFolder
LinkFolder-Sektionen dienen dem Erstellen und Entfernen von Verknüpfungen. Es können beispielsweise Verknüpfungen auf dem Desktop oder im Startmenü erstellt werden.

7.1.6 Globale Konstanten

Globale Konstanten sind Text-Platzhalter, die in primären und sekundären Sektionen eingesetzt werden können und zur Laufzeit textuell durch ihre Werte ersetzt werden.
Über die Verwendung von Platzhaltern kann sichergestellt werden, dass Pfade in unterschiedlichen Umgebungen (z.B. auf System mit unterschiedlichen Sprachen oder Betriebssystem-Versionen) richtig gesetzt sind.

Beispiele:

%ProgramFiles32Dir%
c:\programme

%Systemroot%
c:\windows

%System%
c:\winnt\system32

%Systemdrive%
c:\

%Scriptpath%
<Pfad zu laufenden Script>

7.1.7 Zweites Beispiel: tightvnc

Zur Erläuterung nun ein einfaches Script zur Installation von *tightvnc*. Eigentlich würde dieses Script mit dem Aufruf der Silent-Installation in der Winbatch-Sektion auskommen. Bei einer wiederholten Installation erscheint hier (wegen des Neustarts eines laufenden Services) jedoch ein interaktiver Dialog. Dieses Dialog-Fenster wird (so es auftaucht) mit Hilfe von *AutoIt* geschlossen.

```
[Actions]
Message "Installiere 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 Elementare Befehle für primäre Sektionen

7.1.8.1 String-Variable

Variablen-Deklaration
DefVar <variable name>

Variablen-Zuweisung
Set <variable name> = <value>

Beispiel:

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

Wichtig
Stringvariablen werden in primären und sekundären Sektionen unterschiedlich behandelt. In primären Sektionen sind Stringvariablen eigenständige Objekte. Nur hier können sie deklariert und ihnen Werte zugewiesen werden. Entsprechend ist die Verbindung von Variablen und Strings zu einem Stringausdruck mit einem Operator "+" durchzuführen.
Beispiel: `"Installing "+ $ProductId$ +" ..."`
In sekundären Sektionen werden Stringvariablen vor der Ausführung der Sektion durch den Inhalt der Variable ersetzt.
Beispiel: `"Installing $ProductId$ ..."`
Dies ist zu beachten, wenn entsprechende Stringausdrücke per Cut&Paste im Skript kopiert werden.
Der Vorteil dieser Konstruktion ist, dass in Sektionen die außerhalb des *opsi-winst* ausgeführt werden (DosBatch / Execwith) problemlos mit opsi-winst-Variablen gearbeitet werden kann.

7.1.8.2 Message / ShowBitmap

Zur Textausgabe während der Installation:
`Message <string>`

Beispiel:

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

Zur Ausgabe einer Grafik während der Installation:
`ShowBitmap <filename> <subtitle>`

Beispiel:

```
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 Funktionen

HasMinimumSpace
Prüft auf freien Platz auf der Festplatte.

FileExists
Prüft auf Existenz einer Datei oder eines Verzeichnisses.

7.1.8.5 Fehler, Logging und Kommentare

Kommentarzeichen *;*
Zeilen, die mit einem Semikolon (;) beginnen, werden nicht interpretiert.

Comment
Schreibt eine Kommentar-Meldung in die Log-Datei.

LogError
Schreibt eine Fehlermeldung in die Log-Datei.

IsFatalError
Bricht die Ausführung des laufenden Skriptes ab und meldet die Installation als gescheitert zurück.

7.1.8.6 Bedingung zur Ausführung

requiredWinstVersion
gibt die (mindestens) benötigte opsi-winst Version an.

7.1.9 Drittes Beispiel: Standard-Template *opsi-template*

Verwenden Sie dieses Template (bzw. eine aktualisierte Versionen von http://download.uib.de) als Basis für Ihre eigenen Skripte. Das Template-Paket können Sie auf Ihrem Server mittels `opsi-package-manager` installieren (`-i`) oder entpacken (`-x`), um an die enthaltenen Skripte zu gelangen.

setup32.opsiscript: Installationsscript

```
; 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 seperator
Set $ProductId$       = "opsi-template"
Set $MinimumSpace$    = "1 MB"
; the path were we find the product 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: Ausgelagerte Deinstallations-Sub-Sektion

```
; 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]
```

```
;(.... siehe oben .....)
```

uninstall32.opsiscript: Deinstallations-Skript

```
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 Interaktives Erstellen und Testen eines opsi-winst Skriptes

Sie können ein Skript interaktiv anpassen und testen.

Erstellen Sie sich dazu ein Verzeichnis (z.B. `c:\test`) und kopieren Sie die Scripte des opsi-template (`setup.ins`, `delsub.ins` und `uninstall.ins`) in dieses Verzeichnis.

Starten Sie opsi-winst (`winst32.exe`) per Doppelklick. (Beim Starten des opsi-winst auf einem Windows 7 Client muss "ausführen als Administrator" über die rechte Maustaste verwendet werden.) Wenn der `opsi-client-agent` bereits auf Ihrem Rechner installiert ist, finden Sie opsi-winst unter `C:\Programme\opsi.org\opsi-client-agent\opsi-winst`. Wenn nicht, kopieren Sie sich das Verzeichnis `opsi-winst` vom share `\\<opsiserver\opsi_depot_rw`, aus dem Verzeichnis `install\opsi-winst\files`. Sie sehen dann folgendes Fenster:

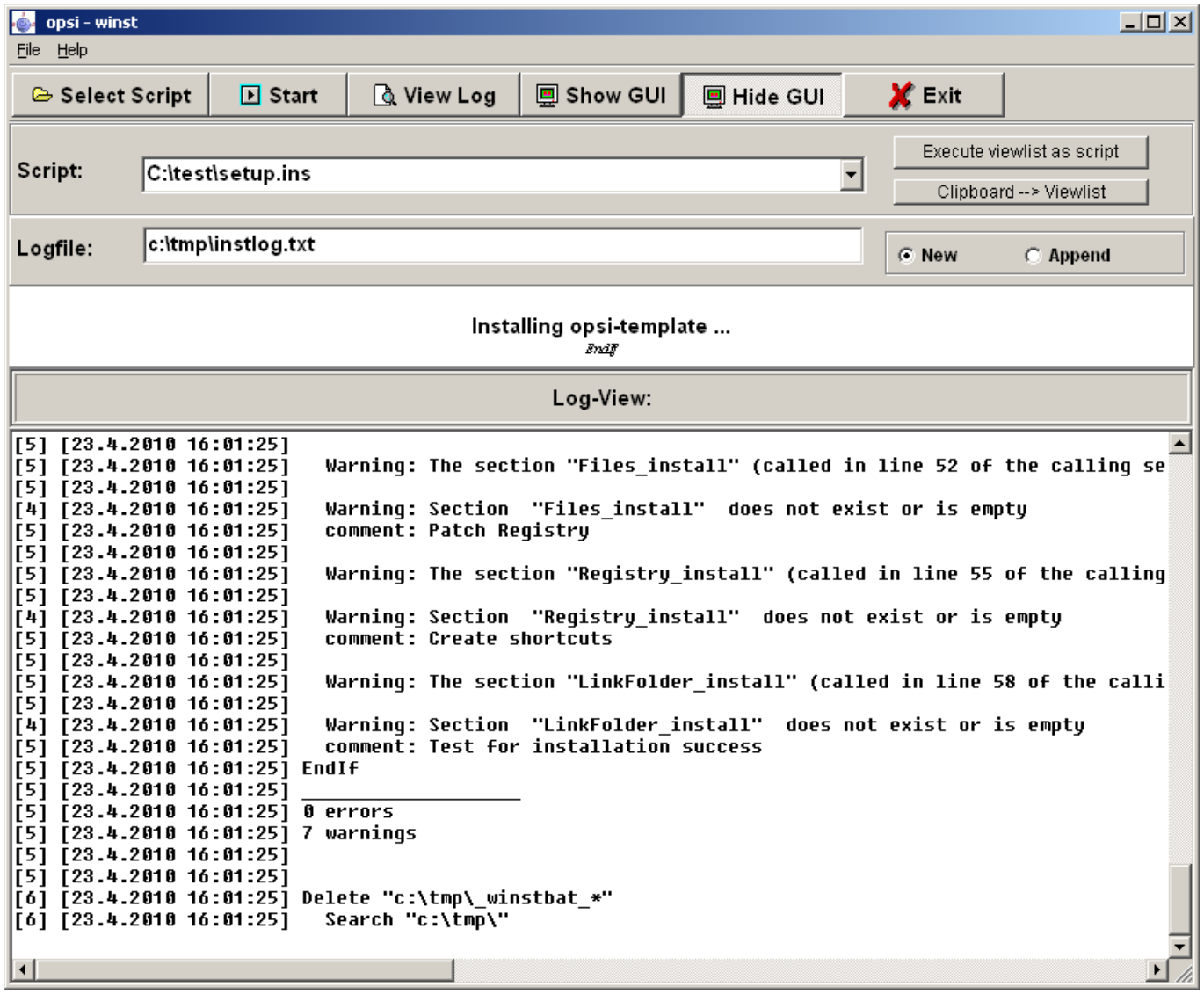

Abbildung 7.2: opsi-winst im interaktiven Modus

Über *Select Script* können Sie das Skript auswählen, dass Sie ausführen möchten. Mit *Start* können Sie das Script starten. Dabei wird das Script auf diesem Rechner ausgeführt. Über *View Log* können Sie sich die Log-Datei des Skript-Laufes anschauen.

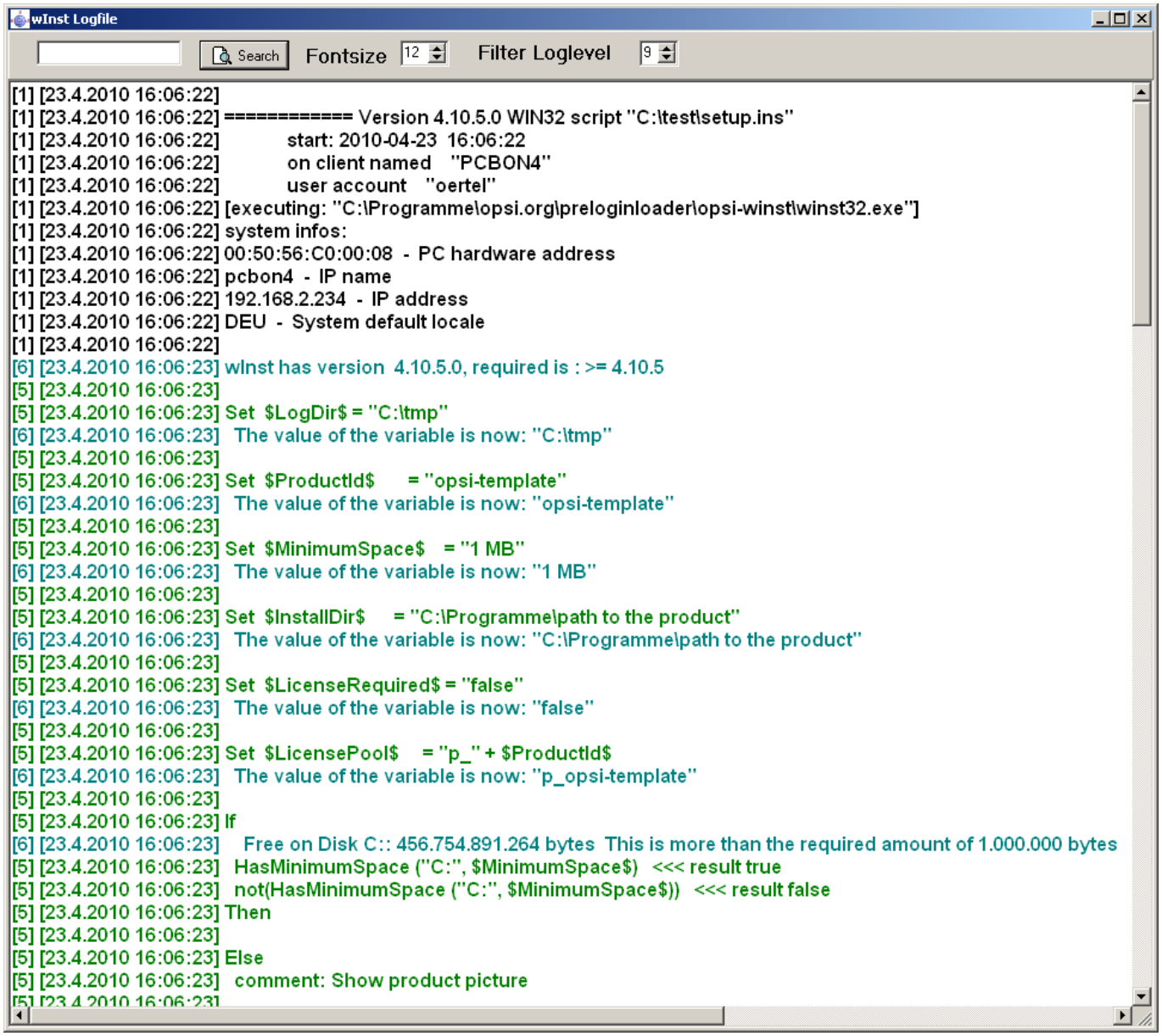

Abbildung 7.3: opsi-winst Log View Fenster

- Schauen Sie sich anhand der Log-Datei an, wie der opsi-winst das Skript interpretiert.
- Kopieren Sie die `Setup.exe`, welche Sie installieren wollen, in das Verzeichnis, in dem die Skripte liegen (z.B. `c:\test`).
- Öffenen Sie das Script `setup.ins` in einem Editor. Im Prinzip können Sie jeden beliebigen Editor verwenden. Wir empfehlen den Editor *jEdit* mit opsi-winst Syntax-Highlighting, wie Sie ihn in der Grundausstattung der opsi-Produkte finden.

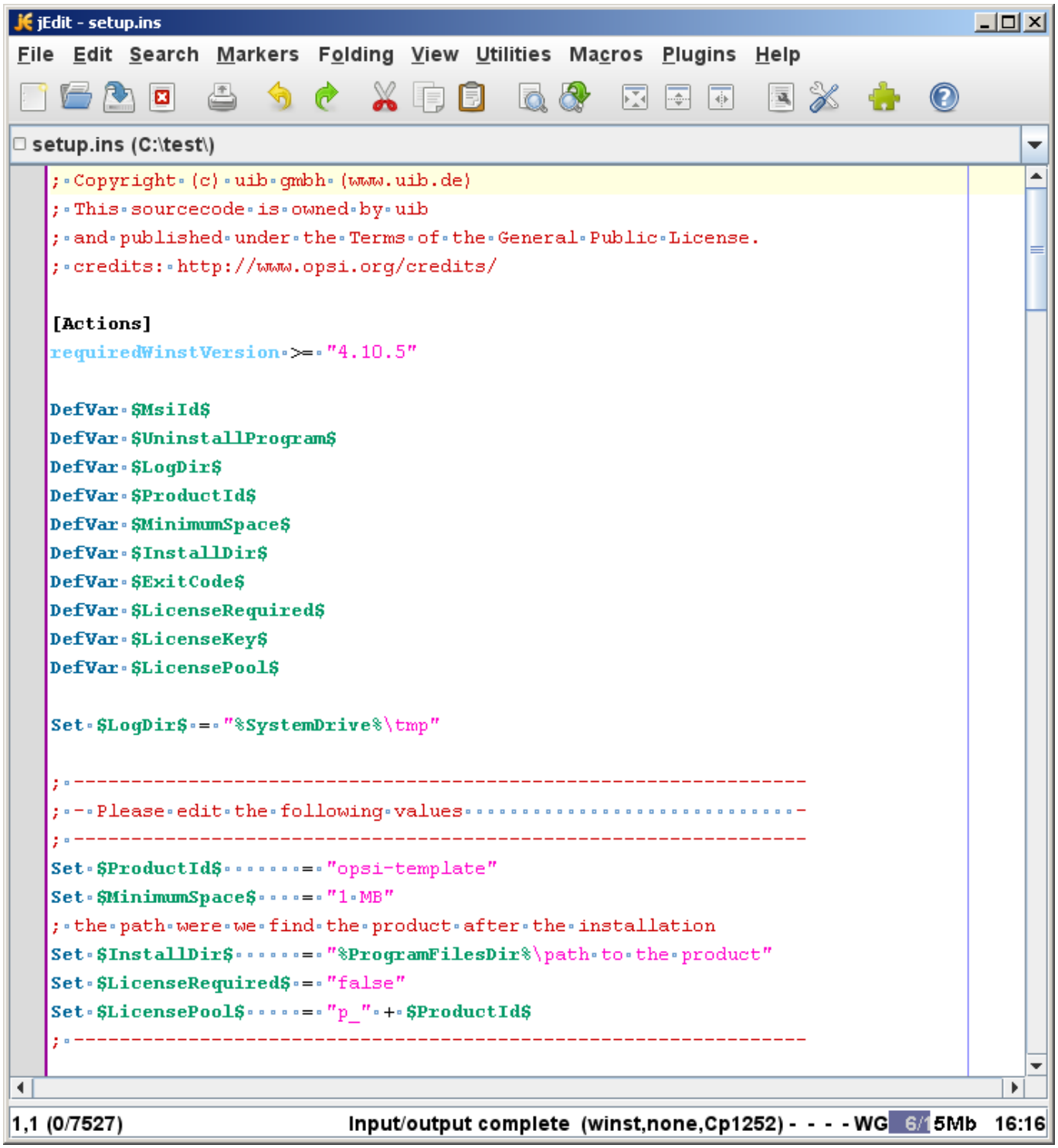

Abbildung 7.4: jEdit mit einem opsi script

- Sie können nun das Skript im Editor anpassen und speichern (Sie können den Editor geöffnet lassen). Wechseln Sie zum opsi-winst-Fenster und starten Sie das Skript erneut über den Knopf *Start* (das Skript muss nicht neu ausgewählt werden). Schauen Sie sich das auf Basis Ihrer Änderungen im Skript veränderte Log über `View Log` an.
- Auf diese Art und Weise, also über die Wiederholung der Punkte:

- Anpassung des Skriptes und speichern
- Skript ausführen
- Log überprüfen
  können Sie nach und nach Ihre Skripte so anpassen, dass sie das tun, was Sie wünschen.

Hinweise zur Lösung von Detail-Problemen finden Sie im nächsten Kapitel. Im übernächsten Kapitel wird erklärt, wie Sie aus den so erstellten Skripten ein opsi-Produkt erstellen, das Sie auf dem opsi-server installieren können.

7.1.11 Hinweise zu den Teilaufgaben im opsi-template

7.1.11.1 Silent oder Unattended Schalter finden

Beim „unattended“ oder „silent setup“ wird das Original-Setup-Programm über Kommandozeilen-Argumente in einen nicht interaktiven Modus gestellt.

Das Problem dieser Installationsmethode ist es die geeigneten Kommandozeilenargumente zu finden.

Suche in Schaltersammlungen im Internet: Bevor man sich in Forschungen stürzt, ist dringend zu empfehlen bei opsi.org zu schauen, ob jemand das Problem bereits gelöst hat:

Fertige opsi-winst-Skripte aus der Community gibt es im opsi community Wiki.

Ermitteln des Herstellers des Setup-Programms: Die meisten Setupprogramme sind auf Basis von Frameworks wie *Inno*, *NSIS*, *Installshield* oder *Wise* gebaut. Jedes dieser Frameworks hat eigene typische Setupschalter. Um das Framework zu ermitteln kann unter anderem folgende Methode verwendet werden: Mit dem Kommandozeilen Programm `strings` werden die Strings aus der `setup.exe` extrahiert und danach mit `grep` bzw. `findstr` nach den Namen der Frameworks gesucht.

Unter Linux sieht der dazu nötige Befehl wie folgt aus (setzen Sie für <mysetup.exe> den Namen Ihrer setup.exe ein):

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

Unter Windows muß der Befehl `strings.exe` erst installiert werden. Einen entsprechenden Download findet man hier: http://technet.microsoft.com/en-us/sysinternals/bb897439

Unter Verwendung dieses Programms, sieht der Befehl unter Windows dann wie folgt aus (setzen Sie für <mysetup.exe> den Namen Ihrer setup.exe ein):

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

Die selbe Methode verwendet der `opsi-setup-detector`.

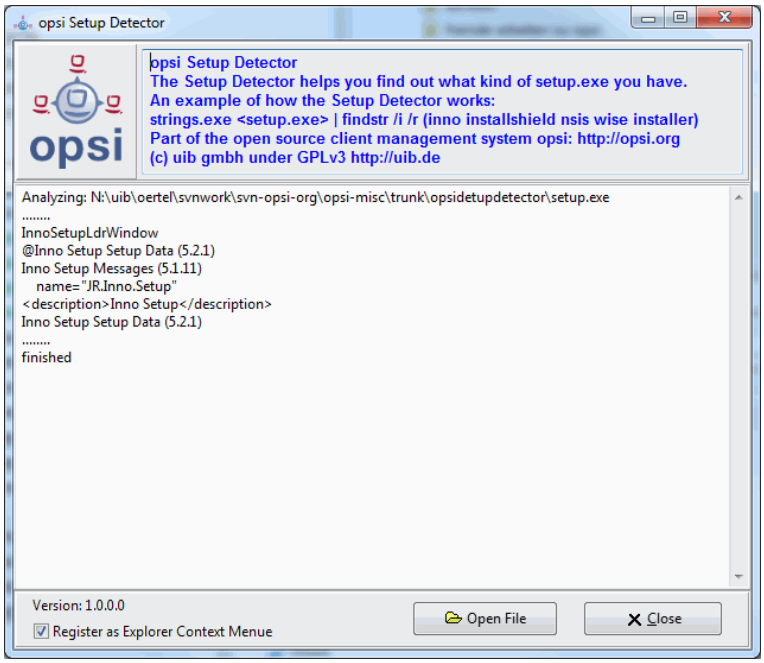

Abbildung 7.5: opsi setup detector

Dieses grafisch-interaktive Programm lässt sich zudem in das Kontextmenü des Explorers Einbinden.

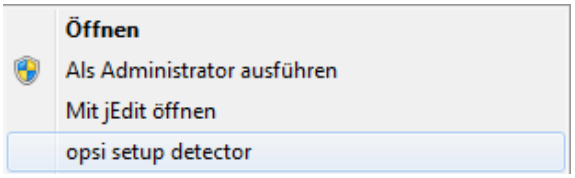

Abbildung 7.6: opsi setup detector im Kontextmenü des Explorers

Der *opsi setup detector* ist Bestandteil der opsi-Repositories für Windows und kann über diese bezogen werden.

Suche beim Hersteller des Setup-Programms: Setup-Programme werden in der Regel von den Herstellern der diversen Softwareprodukte nicht selbst geschrieben. In den meisten Fällen bedienen sich vielmehr die Produkthersteller selbst spezieller Softwareprodukte, mit denen Setup-Programme relativ einfach erstellt werden können. Die verwendbaren Kommandozeilenargumente sind daher zumeist typisch für das verwendete Produkt zur Erstellung von Setup-Programmen.

Im opsi Wiki: Software integration web-links finden weitere Weblinks zu Seiten, die erläutern, wie man Setupprogramme erkennt und wie Ihre typischen Schalter sind.

Auf der Homepage des Herstellers des Setupprogramms wird man zumeist mit der Suche nach Stichworten wie "silent", "silent Install" oder "unattended" fündig.

Suche beim Hersteller des Programms: Da die Anforderung einer automatischen Installation inzwischen vielen Herstellern bewusst ist, finden sich häufig in der Produkt-Dokumentation oder auf der Homepage des Herstellers Anleitungen oder Hinweise hierzu.

7.1.11.2 Weitere wichtige opsi-winst Funktionen

Einen Überblick über die opsi-winst Funktionen gibt die Referencecard:
http://download.uib.de/opsi4.0/doc/opsi-winst-reference-card-en.pdf

Eine detaillierte Dokumentation ist im opsi-winst Handbuch zu finden:
http://download.uib.de/opsi4.0/doc/opsi-winst-manual-de.pdf

Hier noch einige Hinweise auf besonders wichtige Elemente:

Stringlisten: Stringlisten sind sehr mächtig, insbesondere zur Auswertung von Ausgaben externer Programme. Lesen Sie dazu die opsi-winst-Dokus.

ExitWindows: Neustart/Herunterfahren des Systems und Beendung des opsi-winst.

- `ExitWindows /Reboot`
  Rechner-Neustart nach Abschluss des laufenden Skriptes.
- `ExitWindows /ImmediateReboot`
  Sofortiger Neustart.
- `ExitWindows /ImmediateLogout`
  Sofortige Beendigung der Skript-Bearbeitung und Beendung des opsi-winst.

Product-Properties: Für manche Produkte ist es erforderlich, Optionen zur Verfügung zu stellen. Diese werden zur Laufzeit Client-spezifisch ausgewertet. Wie solche Properties erstellt werden, ist im Kapitel Erstellen eines opsi-Produkt-Pakets beschrieben.

Der Zugriff auf die Werte der Properties geschieht über die Funktion `GetProductProperty`:

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

7.1.11.3 Installation mit angemeldetem Benutzer

Vereinzelt taucht das Problem auf, dass sich Installationen nur bei angemeldetem Benutzer durchführen lassen. Ein Hinweis auf diese Problematik ist es, wenn ein opsi-winst-Skript das eine unattended Installation enthält, beim manuellen Aufruf durch einen administrativen Benutzer funktioniert, im Rahmen der automatischen Installation über opsi jedoch scheitert.

Eine mögliche Ursache ist dann, dass dieses Setupprogramm einen angemeldetem Benutzer bzw. den Zugriff auf ein Benutzer-Profil benötigt. Handelt es sich um eine MSI-Installation, hilft eventuell die Option `ALLUSERS=1`.

Beispiel:

```
[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
```

Eine weitere Möglichkeit ist, dass sich das Installationsprogramm zu früh beendet, z.B. weil ein ein Sub-Prozess gestartet wird. In diesem Fall können die Parameter `/WaitSeconds <Anzahl Sekunden>` oder `/WaitForProcessEnding "program.exe" /TimeOutSeconds "<Anzahl Sekunden>"` für den WinBatch-Aufruf helfen.

Eine andere, wenn auch aufwendigere, Möglichkeit dieses Problem zu lösen ist, einen administrativen Benutzer temporär anzulegen und diesen zur Installation der Software zu verwenden. Dies können Sie auf Basis des Templates *opsi-template-with-admin* durchführen.

7.1.11.4 Arbeiten mit MSI-Paketen

Microsoft hat mit Windows 2000 ein eigenes Installationskonzept vorgestellt, das auf dem Microsoft Installer Service, kurz „MSI“ beruht. Inzwischen sind viele Setup-Programme MSI-konform.

MSI-Konformität bedeutet, dass die eigentliche Installation darin besteht, dass an den MSI ein Paket von Installations-Anweisungen übergeben wird (im Prinzip eine Datei mit einen Namen der Form „produkt.msi“) und der MSI dieses Paket dann ausführt.

In der Praxis sieht dies meist so aus, dass die zu einem Produkt gehörige „setup.exe“ eine Datei „produkt.msi“ und ein zusätzliches Steuerprogramm für die Installation enthält. Das Steuerprogramm packt „produkt.msi“ aus und fragt, ob eine Installation starten soll. Wird dies bestätigt, prüft das Steuerprogramm, ob der MSI schon eingerichtet ist und übergibt bei positivem Ergebnis der Prüfung diesem die „produkt.msi“. Ist der MSI nicht eingerichtet und wird insbesondere das Programm „msiexec.exe“ nicht gefunden, so startet das Steuerprogramm zuerst eine Installationsprogramm für den MSI.

Klickt man bei der Frage, ob die Installation starten soll, nicht auf „weiter“, sondern ruft den Explorer auf, so findet sich das ausgepackte MSI-Paket meist in einem temporären Verzeichnis.

Dieses Paket kann nun dazu verwendet werden, eine Installation „unattended“ - also „unbewacht“, d.h. ohne dass ein Benutzereingriff erforderlich ist - auszuführen. Dazu ist bei vorhandener `msiexec.exe` aufzurufen:

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

7.1.11.5 Customizing nach einer silent/unattended Installation

Häufig will man nach einer erfolgreichen Silent-Installation Anpassungen an der Installation vornehmen. Hierzu bietet der opsi-winst ein mächtiges Werkzeug. Doch bevor dies eingesetzt werden kann muss oft ermittelt werden, welche in der graphischen Oberfläche getätigten Änderungen zu welchen Veränderungen in Dateien und der Registry führen.

Hierzu können die unter Abschnitt 7.1.11.7 vorgestellten Werkzeuge eingesetzt werden. Häufig führen aber auch kleinere Werkzeuge schneller zum Erfolg.

Einige beliebte Werkzeuge sind:

- sysinternals
- regshort

7.1.11.6 Einbindung mittels interaktiven Setup-Programms und automatisierten Antworten

Eine weitere schnelle Möglichkeit zur Einbindung in die automatische Softwareverteilung ist das *Setup mit automatisierten Antworten*. Hierzu wird eine Steuerungs-Software verwendet, die über ein Skript die Interaktion eines Anwenders mit den erscheinenden Dialog-Fenstern automatisiert.
Wir empfehlen hierfür den Einsatz der Software AutoHotkey oder AutoIt.

AutoIt bietet eine ganze Reihe zusätzlicher Möglichkeiten, den Setup-Prozess zu steuern. Auch eventuelle Fehlerzustände können (so vorher bekannt) mit dem Einsatz von `[ADLIB]`-Sektionen im Skript abgefangen werden.

Ein prinzipielles Problem bleibt bestehen: Nicht vorhergesehene (und im Skript berücksichtigte) Fenster können das Skript zum Stoppen bringen.

Außerdem kann der Anwender mittels Maus und Tastatur (wenn diese nicht gesperrt sind) in den automatisierten Prozess eingreifen und den Ablauf damit verändern. Ein Unattended- oder Silent-Setup ist daher immer die bessere Lösung.

Sehr gut kann auch eine Kombination aus beiden Ansätzen funktionieren: Das Silent-Setup übernimmt die eigentliche Installation und ein AutoIt-Skript fängt bekannte Sonderbedingungen ab.

Wenn das Ausführen von Installationen in der opsi-client-agent Konfiguration auf einen anderen Desktop verlegt wird oder der Desktop gesperrt wird, haben verschiedene autoit Funktionen Probleme.

Daher sollten in *opsi-winst* Skripten die folgenden Funktionen gemieden werden:

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

Das sind leider genau die 3 am meisten verwendeten.

winwait()
kann ersetzt werden durch die Funktion
`opsiwinwait($title, $text, $maxseconds, $logname)`
welche in der folgenden Weise definiert wird:

```
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
```

Dabei ist:

- `$title` der Titel des Fensters
- `$text` ein Teil des sichtbaren Textes im Fenster
- `$maxseconds` der timeout in Sekunden
- `$logname` der Name der Logdatei

Send()
kann ersetzt werden durch die Funktion
`opsiControlClick($title, $text, $id, $maxseconds, $logname)`
bzw. durch
`opsiControlSetText($title, $text, $id,$sendtext, $maxseconds, $logname)`
welche in der folgenden Weise definiert werden:

```
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
```

Dabei ist:

- `$title` der Titel des Fensters
- `$text` ein Teil des sichtbaren Textes im Fenster
- `$id` die numerische ControlId des Buttons oder Editierfeldes
- `$sendtext` der Text welcher eingfügt werden soll
- `$maxseconds` der timeout in Sekunden
- `$logname` der Name der Logdatei

Dabei muss mit der `Au3info.exe` die *ControlId* ermittelt werden. Bitte die numerische *ControlId* verwenden, andere Varianten scheinen Probleme zu machen:

Hier ein Auszug aus einem script.
In diesem wird dann noch eine Logdatei angelegt, die mit folgenden Befehlen in die Logdatei des *opsi-winst* integriert wird:

```
includelog "c:\tmp\au3.log" "500"
```

Das Beispiel:

```
[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 , $text,$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

; 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
```

siehe auch:
* AutoIT FAQ: Why doesn't my script work on a locked workstation? * AutoIT Dokumentation * AutoIT: Dokumentation: Controls * AutoIT: Dokumentation: Functions

7.1.11.7 Analyse und Neu-Paketieren

Wenn der Entwickler einer Anwendung ein Paket zur Auslieferung der Anwendung schnürt, kennt er die benötigten Komponenten. Im Nachhinein, wenn schon ein Paket existiert, das mittels eines Setup-Programm zu installieren ist, kann die Kenntnis, welche Komponenten installiert werden müssen, damit eine Anwendung wie gewünscht auf einem Arbeitsplatzrechner lauffähig ist, aus der Studie der Effekte bei der Ausführung des vorhandenen Setup-Programms gewonnen werden.

Eine Reihe von Werkzeuge zum Analysieren von Setup-Programmen kommen hierbei in Frage. So z.B.:

- InstallWatch Pro
- appdeploy-repackager

7.1.11.8 Verfahren zur Deinstallation von Produkten

Um eine installierte Software von einem Rechner zu entfernen, kann ein Deinstallations-Skript erstellt werden. Grundsätzlich besteht bei einer Deinstallation die Schwierigkeit, dass nicht immer klar ist, wie das Produkt auf dem Rechner vorliegt und was alles entfernt werden muss. Auch nach der Installation können neue Dateien oder Registry-Einträge, die die Software betreffen, hinzugekommen sein. Weiterhin muss darauf geachtet werden nicht zu viel zu entfernen, um die Systemstabilität nicht zu gefährden. Meist weiß nur der Hersteller genau, wie mit seinem Produkt bei der Deinstallation umzugehen ist. Ähnlich wie bei der Installation, existieren zu diesem Zweck Deinstallations-Routinen, die dem Produkt beiliegen. Wenn es die Möglichkeit gibt, diese ohne Benutzer-Interaktion auszuführen, kann dies schon ein entscheidender Schritt sein. Ist eine solche Routine nicht vorhanden oder muss diese erweitert werden, so existieren viele opsi-winst-Befehle, die zur Deinstallation nützlich sein können. Im Folgenden soll nun ein Überblick über Möglichkeiten zur Deinstallation gegeben werden, die durch Beispiele verdeutlicht werden.

Verwenden einer Deinstallations-Routine Liefert der Hersteller des Produkts ein Programm (oder ein MSI-Paket) zur Deinstallation, so muss zunächst geprüft werden, ob dies auch ohne Benutzer-Interaktion ausgeführt werden kann

(silent-mode). Sollte dies nicht von Hause aus möglich sein, kann der Einsatz eines AutoIt-Skriptes in Verbindung mit der Deinstallations-Routine hilfreich sein. Der Aufruf der ausführbaren Datei kann im opsi-winst-Skript in einer Winbatch-Sektion geschehen, z.B.:

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

Trotz dieser Unterstützung des Herstellers sollte man sich jedoch nicht auf die korrekte Beseitigung des Produkts verlassen und auf einem Testsystem zusätzlich prüfen, ob das System nach der Deinstallation weiter stabil läuft und ob Dateien oder Registry-Einträge zurückgeblieben sind.

Falls das Produkt als MSI-Paket bereitgestellt und mittels *msiexec* installiert wurde, ist es in der Regel auch eine Deinstallation mittels *msiexec* möglich. Dazu ruft man den `msiexec.exe` mit dem Parameter `/x` auf und übergibt zusätzlich den MSI-Paket oder dessen *GUID* an. Um die Benutzer-Interaktion zu deaktivieren, kann zusätzlich der Parameter `/qb-!` übergeben werden.
Dies ergibt nun den folgenden Aufruf:

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

Statt das MSI-Paket zu übergeben, gibt es auch die Möglichkeit die *GUID* an `msiexec.exe` zu übergeben. Diese ID ist eindeutig, Produkt-spezifisch und auf allen Systemen, auf denen das MSI-Paket installiert ist, gleich. Sie findet sich zum Beispiel im Zweig `HKLM\Software\Microsoft\Windows\CurrentVersion\Uninstall` der Registry.
Ein Beispiel einer Deinstallation mittels dieser GUID sieht folgendermaßen aus:

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

Sollten diese Methoden nicht oder nicht vollständig funktionieren, so muss mit einem opsi-winst-Skript nachgeholfen werden, wie es im nächsten Abschnitt beschrieben ist.

Nützliche opsi-winst-Befehle zur Deinstallation Wurde ein Produkt mit den opsi-winst-Funktionen installiert oder gibt es keine Deinstallation vom Hersteller, so muss ein eigenes opsi-winst-Skript zu Deinstallation geschrieben werden. Um den Programmierer bei dieser Arbeit zu unterstützen stellt der *opsi-winst* einige Funktionen bereit, die speziell bei der Deinstallation notwendig oder hilfreich sind. Es soll an dieser Stelle nur ein kurzer Überblick gegeben werden, eine genaue Beschreibung der Befehle und deren Parameter findet sich im opsi-winst-Handbuch.

Der einfachste Fall ist das Löschen einer oder mehrerer Dateien vom System. Dies geschieht in einer Files-Sektion mittels des Befehls

```
delete -f Dateiname
```

oder für ein Verzeichnis mit Unterverzeichnissen

```
delete -sf Verzeichnisname\
```

Der Parameter `f` steht dabei für force, um die Datei wirklich zu löschen, auch wenn diese schreibgeschützt ist, der Parameter `s` für subdirectories (mit Unterverzeichnissen/rekursiv). Soll eine Datei oder ein Verzeichnis aus allen Benutzer-Profilen gelöscht werden, so kann diese Files-Sektion mit dem Parameter `/AllNTUserProfiles` aufgerufen werden. (siehe opsi-winst-Handbuch)

Möchte man einen Verzeichnisbaum löschen, in dem sich auch Dateien mit dem Attribut „versteckt“ oder „systemdatei“ befinden, muss momentan ein Umweg über den Befehl `rmdir` gegangen werden, der über eine *DosInAnIcon*-Sektion aufgerufen werden kann.

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

Muss vor dem Löschen evtl. ein laufender Prozess beendet werden, so kann dies mit dem Namen des Prozesses (zu sehen im Task-Manager) und dem opsi-winst-Befehl `KillTask` geschehen:

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

Sollte das Produkt – oder Teile davon – als Service laufen, so muss dieser vor der Deinstallation beendet werden. Man kann dazu den Service in der Registry auf "inaktiv" schalten und den Rechner neu starten oder aber man benutzt den System-Befehl `net` mit dem Parameter `stop`, um den Service sofort zu stoppen und anschließend – ohne Neustart – die zugehörigen Dateien zu löschen.

```
net stop <servicename>
```

Besondere Vorsicht ist beim Löschen von .dll-Dateien geboten, die noch von anderen Produkten verwendet werden könnten. Sie müssen individuell behandelt werden, weshalb hier leider kein allgemein gültiges Rezept gegeben werden kann.

Um einzelne Einträge aus der Registry mittels opsi-winst zu löschen, kommt der Befehl `DeleteVar` zum Einsatz, der innerhalb einer *Registry*-Sektion eines opsi-winst-Skripts verwendet werden kann. Er löscht Einträge aus dem momentan geöffneten Key:

```
DeleteVar <VarName>
```

Möchte man einen Registry-Key samt seiner Unterschlüssel und Registry-Variablen löschen, so geschieht dies mittels des opsi-winst-Befehls `DeleteKey`, z.B.:

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

7.1.11.9 Bekannte Besonderheiten der 64 Bit-Unterstützung

Der opsi-winst ist ein 32-Bit Programm. Skripte zur Installation von 32-Bit Programmen funktionieren in der Regel auch auf 64-Bit Systemen korrekt. Einige Konstanten, wie `%ProgramFilesDir%`, liefern Werte, die für die Verwendung mit 64-Bit-Programmen nicht richtig sind. Neuere opsi-winst-Versionen kennen spezielle Befehle und Konstanten für 64-Bit-Systeme. Beachten Sie beim Arbeiten auf 64-Bit-Systemen das entsprechende Kapitel im opsi-winst-Handbuch.

7.2 Erstellen eines opsi-Produkt-Pakets

In opsi werden die Installationsdateien, das opsi-winst-Skript zur Installation auf den Client und die Metadaten zu einem Paket zusammengefasst, welches zur Installation dieses Softwareproduktes auf einem opsi-server dient.

Die wesentlichen Vorteile dieses Paketformates sind:

- Einfache menügeführte Erstellung mit dem Programm `opsi-newprod`.
- Ablage aller relevanten Metadaten in einer einfach zu editierenden Datei.
- Optional menügeführtes Auspacken des Paketes mit der Möglichkeit Vorgaben zu ändern.
- Informationen über die im Paket enthaltene Produktversion, Paketversion und eventueller kundenspezifischer Erweiterungen werden abgespeichert und sind am Paketnamen erkennbar. Diese werden im Installationsverzeichnis abgelegt und im opsi-Configeditor angezeigt. Auf diese Weise wird der Überblick über unterschiedliche Versionen erleichtert (Productlifecycle Management).
- Zur Erstellung und zum Auspacken von Produkten sind keine root-Rechte erforderlich. Es langen hierzu die Rechte der Gruppe `pcpatch`.

Das Paket selber besteht aus einem per Gzip komprimierten cpio Archiv. In diesem Archiv befinden sich zwei Verzeichnisse:

CLIENT_DATA
Hier liegen die Dateien, die im opsi-Depot (`/var/lib/opsi/depot/<productid>`) für die Clients verfügbar sein sollen.

OPSI
In der Datei `control` finden sich die Metadaten des Produkts wie der Name, die Version oder auch Produkt-Abhängigkeiten. Weiterhin finden sich im Verzeichnis `OPSI` die Skripte `preinst` und `postinst`, die vor bzw. nach der Installation des Produkt-Pakets auf einem opsi-depotserver ausgeführt werden. Hier können Sie, wenn benötigt, entsprechende Erweiterungen unterbringen.

7.2.1 Erstellen, Packen und Auspacken eines neuen Produktes

Zum Erstellen eines Produktes müssen Sie sich auf dem Server einloggen (von Windows aus z.B. per `putty.exe` http://www.chiark.greenend.org.uk/~sgtatham/putty/download.html).

Die wesentlichen Befehle zum Erstellen und Installieren eines opsi-Produktes sind:

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

Zum Erstellen eines neuen Produktes benötigt man mindestens die Rechte der Gruppe `pcpatch`.

Opsi unterstützt die parallele Kompression mittels `pigz`, sofern dieses mindestens in Version 2.2.3 vorliegt. Ist diese oder eine höhere Version installiert, so wird opsi diese automatisch zur (De-)Kompression von Produkten verwenden. Bitte beachten Sie dabei, dass sowohl `gzip` als auch `pigz` Archive erstellen, welche von der bandbreitenschonenden Synchronisierung mittels `rsync` profitieren können, allerdings sind die Dateien nicht bit-kompatibel. Das bedeutet, dass falls bereits Produkte mittels `gzip` komprimiert und an ein anderes Depot übertragen wurden, nun aber ein mittels `pigz` komprimiertes Paket übertragen wird, mehr als nur die eigentlichen Deltas übertragen werden. Das ist nur bei der ersten Umstellung des verwendeten Komprimierungsprogramms der Fall, bei alle nachfolgenden Übertragungen werden wiederum nur die Unterschiede übertragen. Um die Verwendung von `pigz` auf einem Server explizit zu deaktivieren, setzen Sie bitte den Wert `use_pigz` in der Sektion `packages` in der Datei `/etc/opsi/opsi.conf` wie nachfolgend gezeigt auf `False`:

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

Sie sollten die Produkte in dem Verzeichnis `/home/opsiproducts` erstellen, welches der Gruppe `pcpatch` gehört und die Rechte 2770 hat (Setgroupid Bit für Gruppe `pcpatch` gesetzt), sowie als Share `opsi_workbench` freigegeben ist.

Achtung
Unter Distributionen der SUSE-Familie ist dieses Verzeichnis unter `/var/lib/opsi/workbench`.

7.2.1.1 Erstellen mit opsi-newprod

Zum Erstellen wechselt man in dieses Verzeichnis und ruft `opsi-newprod` auf. Das Programm fragt daraufhin nach dem Typ des zu erstellenden Paketes. Dies ist üblicherweise der Typ `localboot` für Produkte, die über den *opsi-client-agent*/*opsi-winst* installiert werden. Der Typ `netboot` steht für Produkte, die über das opsi-Linux-Bootimage ausgeführt werden (wie z.B. die Betriebssystem-Installationen).

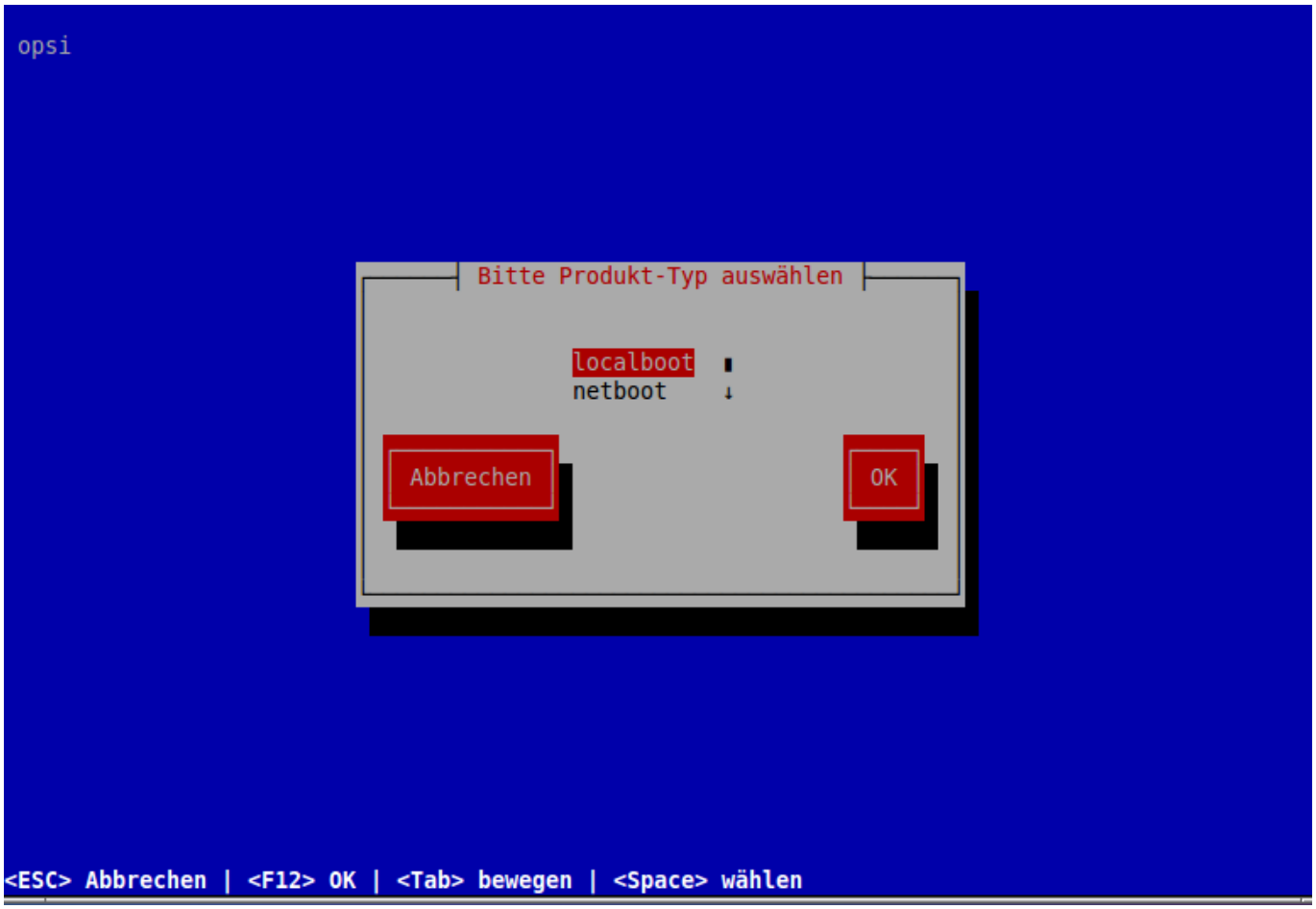

Abbildung 7.7: Auswahl des Produkttyps: localboot

Wählen Sie nun mit Tab OK (oder bestätigen mit F12). Nun müssen Sie die wesentlichen Produktdaten eingeben. Am oberen Rand ist hierzu eine Hilfe, die erläutert was die Felder bedeuten.

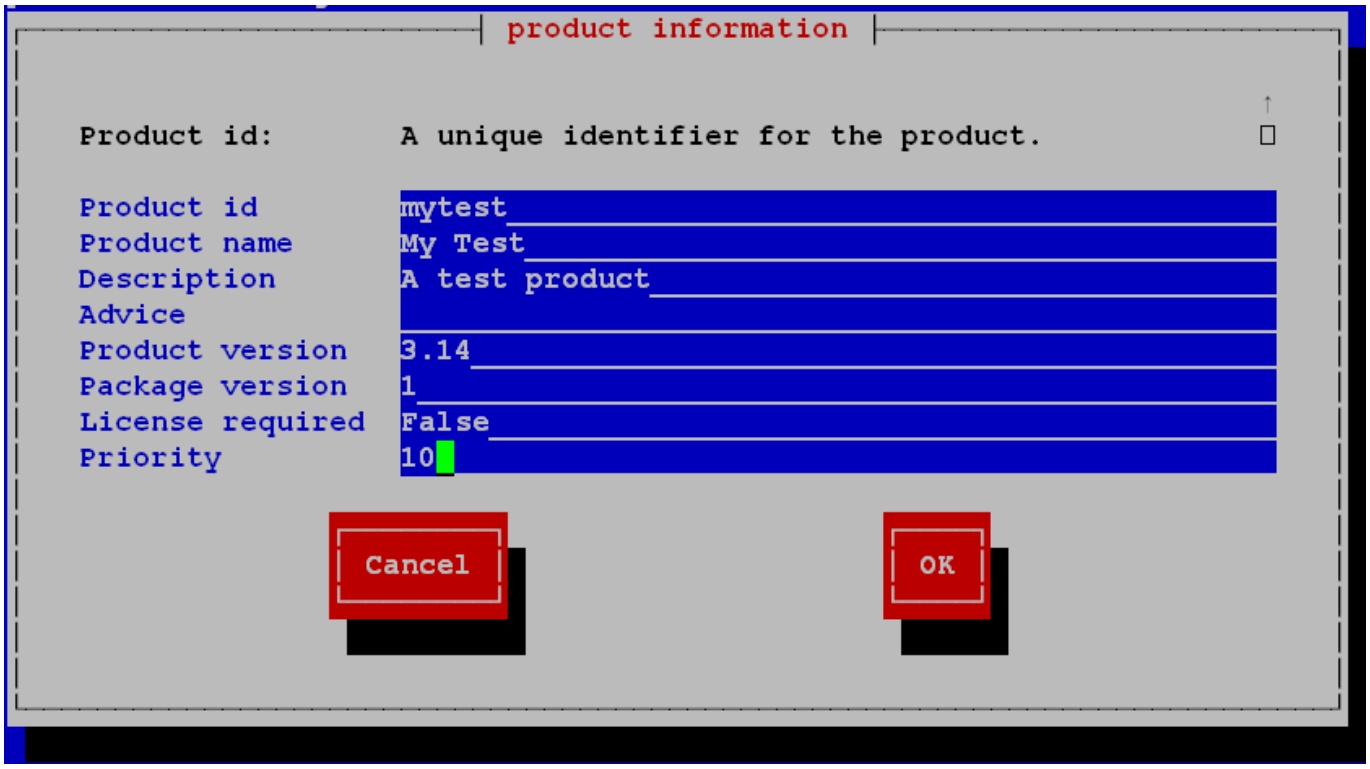

Abbildung 7.8: Eingabe der Produktinformationen

Product Id
ist ein eindeutiger Bezeichner für das Produkt in der Regel unabhängig von der Version
Bitte nur Kleinbuchstaben verwenden, keine Umlaute, keine Leerzeichen, keine Sonderzeichen - - ist als Trenner erlaubt.

Product name
ist der Klartextname des Produkts (wir empfehlen die Vermeidung von Umlauten, - ist erlaubt, keine Leerzeichen).

Description
ist eine ergänzende Beschreibung zum Produkt, die z.B. im opsi-Configeditor unter `Beschreibung` angezeigt wird.

Advice
ist eine ergänzende Beschreibung, in der Regel zum Umgang mit dem Produkt, die zu beachten ist und im opsi-Configeditor unter `Notiz` angezeigt wird.

Product version
ist die Version der eingepackten Software (max. 32 Zeichen).

Package Version
ist die Version des Paketes für die Produktversion. Sie dient dazu, Pakete mit gleicher Produktversion, aber z.B. korrigiertem opsi-winst-Skript zu unterscheiden.

License required
hat bei localboot Produkten keinen Einfluss. Bei netboot Produkten entscheidet diese Option, ob ein Lizenzkey aus dem Lizenzmanagement geholt wird.

Priority
beeinflusst die Installationsreihenfolge. Mögliche Werte liegen zwischen 100 (ganz am Anfang) und -100 (ganz am Ende). Existieren auch Produktabhängigkeiten, so beeinflussen diese zusätzlich die Installationsreihenfolge.

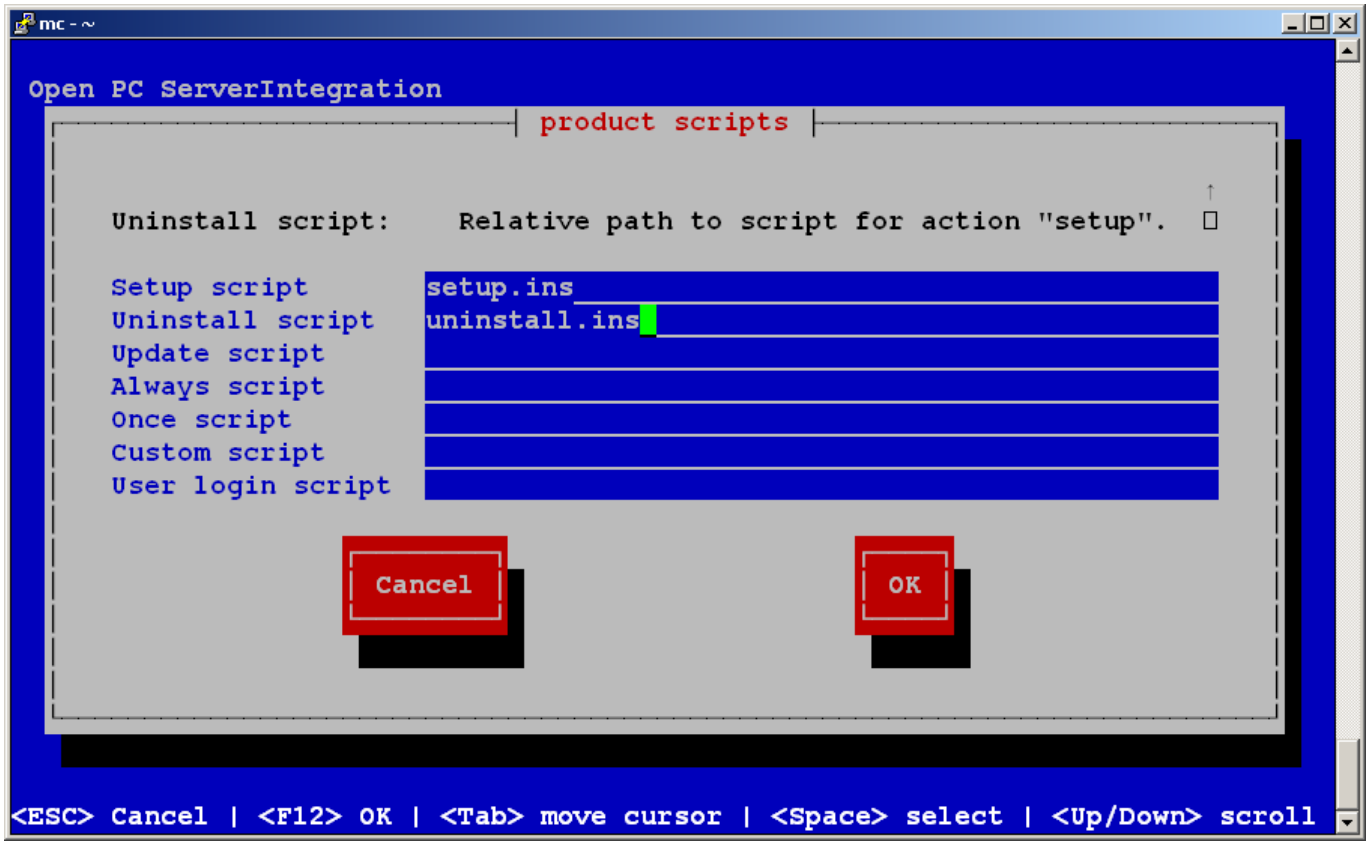

Abbildung 7.9: Eingabe der opsi-winst-Skript Namen für unterschiedliche Aktionen

Nach Eingabe der Produktinformationen werden Sie aufgefordert, die Skripte anzugeben, die Sie für die unterschiedlichen möglichen Aktionen bereit stellen werden.

Üblicherweise heißt das **Setup script** gleich `setup.ins`.

Üblicherweise heißt das **Uninstall script** gleich `uninstall.ins`.

Ein **Update-Script** dient zur geringfügigen Veränderung einer existierenden großen Installation. Wird das Produkt auf setup gestellt, so wird nach dem Abarbeiten des Setup-Skriptes automatisch auch das Update-Skript ausgeführt.

Ein **Always-Script** wird bei jedem aktiv werden des opsi-Clientagenten ausgeführt (z.B. bei jedem Boot).

Ein **Once-Script** hat den Folgestatus `not_installed`. Es handelt sich hierbei um einen sehr selten verwendeten Schalter, den Sie ignorieren sollten, wenn Sie nicht genau wissen, was Sie damit tun wollen.

Ein **Custom-Script** verändert weder Folgeaktion noch Folgestatus. Es handelt sich hierbei um einen sehr selten verwendeten Schalter, den Sie ignorieren sollten, wenn Sie nicht genau wissen, was Sie damit tun wollen.

Ein **userLoginScript** dient dazu nach dem Login des users Modifikationen am Profil des eingeloggten users vorzunehmen. Dies Funktioniert nur im Zusammenhang mit der opsi Erweiterung *User Profile Management* und ist im entsprechenden Kapitel des opsi-Handbuchs beschrieben.

Typ	Folgestatus	Folgeaktion
setup	installed	none
uninstall	not_installed	none
update	installed	none
always	installed	always
once	not_installed	none
custom	*unverändert*	*unverändert*
User login	*unverändert*	*unverändert*

Nachdem nun das Produkt selber beschrieben ist, können Sie eine oder mehrere Produktabhängigkeiten definieren. Wollen Sie keine Produktabhängigkeit definieren so geben Sie `No` ein.

Abbildung 7.10: Eine (weitere) Produktabhängigkeit definieren: Ja / Nein

Zur Erstellung einer Produktabhängigkeit geben Sie die folgenden Daten an. Beachten Sie auch die Hilfe im oberen Teil des Fensters:

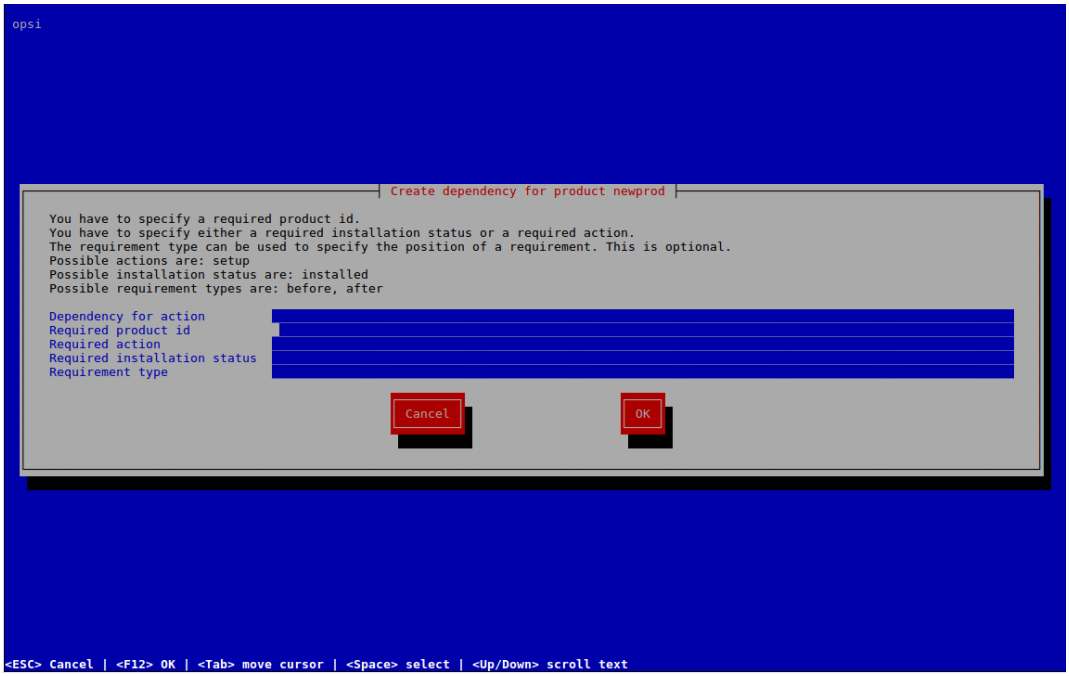

Abbildung 7.11: Eingabe der Daten zur Erstellung einer Produktabhängigkeit

Dependency for Action

Für welche Aktion des Produktes, welches Sie gerade erstellen, soll die Abhängigkeit gelten (setup, deinstall . . .).

Required product id
Productid (Bezeichner) des Produkts zu dem eine Abhängigkeit besteht.

Required action
Sie können entweder eine Aktion anfordern oder (siehe unten) einen Status. Aktionen können z.B. sein : setup, uninstall, update . . .

Required installation status
Status den das Produkt, zu dem eine Abhängigkeit besteht, haben soll (typischerweise `installed`). Liegt ein anderer Status vor, so wird das Produkt auf `setup` gestellt.

Requirement type
Installationsreihenfolge. Wenn das Produkt, zu dem eine Abhängigkeit besteht, installiert sein muss bevor mit der Installation des aktuellen Produkts begonnen werden kann, dann ist dies `before`. Muss es nach dem aktuellen Produkt installiert werden so ist dies `after`. Ist die Reihenfolge egal so muss hier nichts eingetragen werden.

Hinweis:

Leider gibt es derzeit keinen generischen Mechanismus für Deinstallations-Produktabhängigkeiten. Zuverlässig ist der ProductDependency-Mechanismus nur für action: setup und die hierbei zu triggernden (before- oder after-) setup Aktionen und installed Status. Ein requiredAction: uninstall führt leider definitiv zu Fehlern.

Nachdem eine Produktabhängigkeit definiert ist, werden Sie wieder gefragt, ob Sie eine (weitere) Produktabhängigkeit definieren wollen. Wenn ja, wiederholt sich der Vorgang; wenn nein, so werden Sie gefragt, ob Sie eine Produkteigenschaft (Zusatzschalter) definieren wollen mit dem Sie die Installation des Produktes modifizieren können.

Noch ein Hinweis:

Die tatsächliche Installationsreihenfolge ermittelt sich aus einer Kombination von Produktabhängigkeiten und Produktpriorisierung. Details hierzu finden Sie im opsi-Handbuch im Kapitel *Beeinflussung der Installationsreihenfolge durch Prioritäten und Produktabhängigkeiten*

Abbildung 7.12: Eine (weitere) Produkteigenschaft definieren

Antworten Sie ja, so müssen Sie die Produkteigenschaft beschreiben:

Die Produkteigenschaft wird clientspezifisch gespeichert und besteht aus einem Namen (key) der verschiedene Werte (Values) zugeordnet bekommen kann und die dann vom opsi-winst-Skript abgefragt werden können.

Zunächst müssen Sie angeben, ob es sich um ein Textwert (unicode) oder um einen logische Wert also wahr/falsch (boolean) handelt. Wenn Sie unsicher sind, wählen Sie `unicode`.

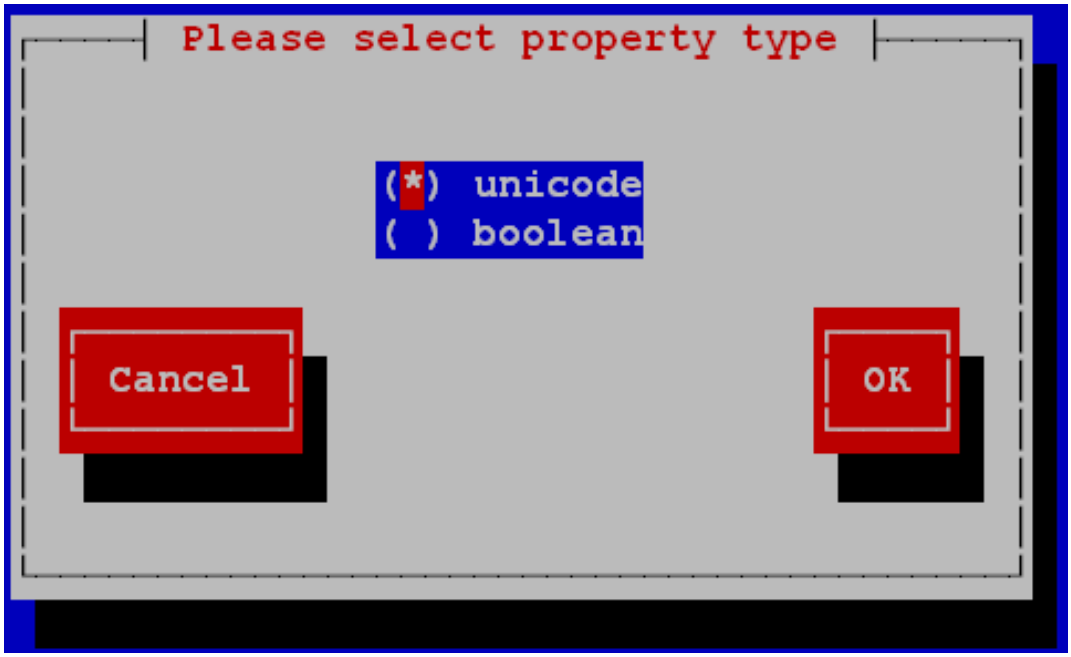

Abbildung 7.13: Datentyp der Produkteigenschaft wählen

Weiterhin wird eine Beschreibung benötigt, die im opsi-configed als Hilfe angezeigt wird. Weiterhin müssen Sie, durch Kommas getrennt, alle Werte angeben, die der Key annehmen darf. Wird hier nichts angegeben, so kann später im opsi-Configeditor ein beliebiger Wert eingegeben werden. Über `Editable` (true/false) können Sie entscheiden, ob neben der vorgegebenen Liste auch andere Werte eingegeben werden dürfen.

Anmerkung
Enthält ein Wert einen Backslash \, so muss dieser doppelt angegeben werden.
Eine Pfadangabe kann beispielsweise wie folgt aussehen: `C:\\temp`

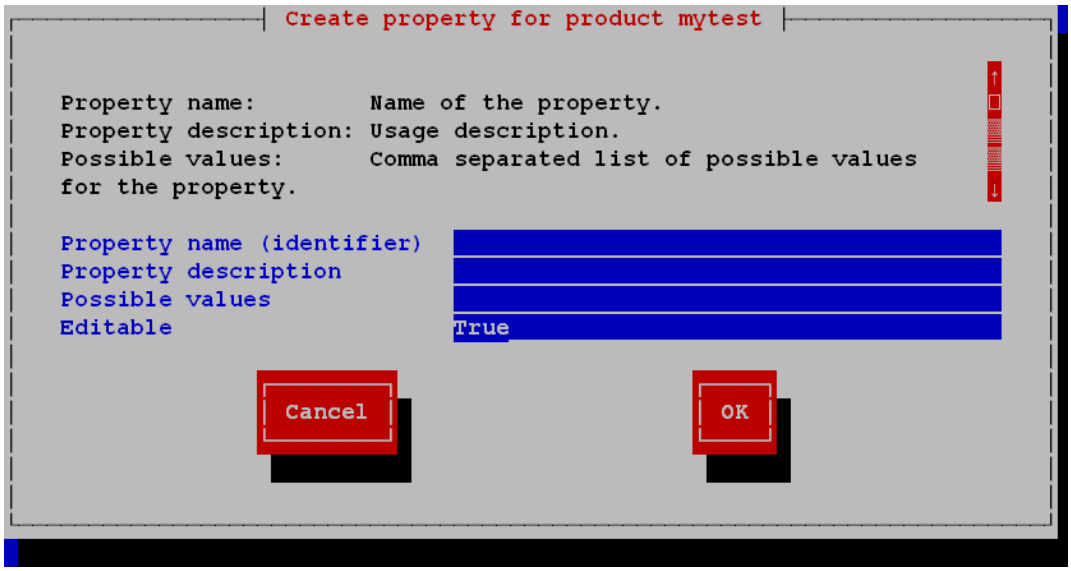

Abbildung 7.14: Beschreibung der Produkteigenschaft

Im Folgefenster müssen Sie festlegen, was der Defaultwert dieser Produkteigenschaft ist.

Abbildung 7.15: Festlegung des Defaultwerts der Produkteigenschaft

Wenn Sie als Typ *boolean* wählen, so reduziert sich die Beschreibung auf *Property name* und *Property description*.

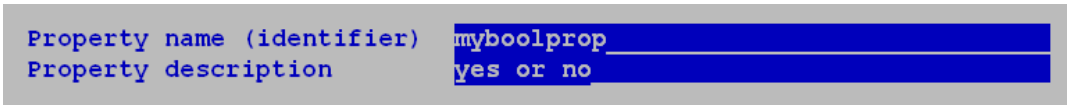

Abbildung 7.16: Beschreibung eines boolschen Properties

Nachdem eine Produkteigenschaft definiert ist, werden Sie wieder gefragt, ob Sie eine (weitere) Produkteigenschaft definieren wollen. Wenn ja, wiederholt sich der Vorgang; wenn nein, so werden Sie als nächstes nach Name und Mail-Adresse gefragt. Diese werden im Changelog des Paketes verwendet und müssen angegeben werden.

Abbildung 7.17: Eingabe der Maintainer Daten

Danach ist das Grundgerüst des Produktes fertig gestellt.

Mit Hilfe des `ls` Befehls finden Sie die oben beschriebene Verzeichnis Struktur. Wechseln Sie in den OPSI-Ordner und setzen Sie erneut den `ls` Befehl ab. Hier befindet sich unter anderem die *control*-Datei, welche die eben eingegebenen Daten enthält und Ihnen auch die Möglichkeit bietet, diese im Editor zu kontrollieren oder zu modifizieren.

Beispiel einer control Datei

```
[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
```

Als nächstes müssen Sie das, für das Produkt erstellte, opsi-winst-Skript und die entsprechenden Dateien nach *CLIENT_DATA* kopieren.

Wenn Sie z.B. das erstellte Skript unter *c:\test* liegen haben, so mounten Sie *\\<opsiserver\opsi_workbench* z.B. nach *w:* und kopieren den Inhalt von *c:\test* in das Verzeichnis *CLIENT_DATA*.

7.2.1.2 Packen mit opsi-makeproductfile

Danach können Sie das Produkt packen. Gehen Sie dazu in das Stammverzeichnis des Produkts und rufen Sie *opsi-makeproductfile* auf. Es wird nun das Produkt gepackt.

opsi-makeproductfile kennt einige Optionen, die sein Verhalten modifizieren:

```
#  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
```

Es ist zu empfehlen die Pakete gleich mit einer zugehörigen md5-Prüfsummendatei zu erstellen. Diese Datei wird unter anderem vom opsi-product-updater genutzt, um nach der Paketübertragung die Paketintegrität sicher zu stellen. Eine solche Datei wird mit dem Schalter `-m` erstellt.

Bei der Übertragung von Paketen auf opsi-depotserver kann auf zsync zurück gegriffen werden, um nur Unterschiede zwischen verschiedenen Paketen zu übertragen. Damit dieses Verfahren verwendet werde kann, wird eine Datei besondere `.zsync`-Datei benötigt. Eine solche Datei wird mit dem Schalter `-z` erstellt.

Wenn es beim Erstellen großer Pakete zu Platzproblemen im temporären Verzeichnis `/tmp` kommt, ist es möglich mittels `-t` ein abweichendes temporäres Verzeichnis anzugeben.

Wenn schon ein Paket dieser Version existiert, so zeigt opsi-makeproductfile eine Rückfrage:

```
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:
```

Mit `o` wählen Sie überschreiben, mit `c` brechen Sie den Vorgang ab und mit `n` können Sie wählen, dass Sie nach einer neuen Product- bzw. Package-Version gefragt werden.

Das gepackte Paket können Sie mit `opsi-package-manager -i <paketname>` auf dem Server installieren.

Weitere Informationen zum opsi-packet-manager siehe opsi-Handbuch.

Kapitel 8

Weitere Informationen

Das opsi-Handbuch (http://download.uib.de/opsi_stable/doc/opsi-handbuch-stable-de.pdf) enthält eine Fülle von weiteren Informationen, die für den produktiven Betrieb wichtig sind.

Wenn Sie dort nicht fündig werden oder Hilfe benötigen, wenden Sie sich an
https://forum.opsi.org

Für produktive Installationen empfehlen wir professionelle Unterstützung durch uib im Rahmen eines Pflege- und Supportvertrages:
http://uib.de/de/support-schulung/support/