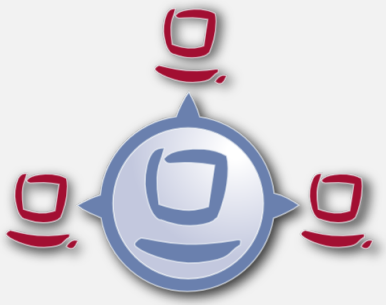

opsi Version 4.0.3 Release Notes

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

Stand: 26.02.2013

Inhaltsverzeichnis

1 Copyright **1**

2 Übersicht der Neuerungen **2**

3 Abkündigung **4**

3.1 Abkündigung der Python 2.4 Unterstützung 4

3.2 Abkündigung: Distributionsversionen 4

4 Hinweise zur Installation **5**

5 opsi Installation beim Shutdown **6**

5.1 Einführung 6

5.2 Vorbedingungen für die Installation beim Shutdown 6

5.3 Inbetriebnahme der Installation beim Shutdown 7

5.4 Technisches Konzept 7

5.4.1 Überblick 7

5.4.2 Durchführung per Shutdown-Skript 8

5.4.3 Registry-Einträge für die Ausführung des Shutdown-Skripts 9

5.4.4 Erforderliche Konfiguration des opsiclientd 10

5.4.5 Spezielle Konfiguration der Installation bei Shutdown 11

5.4.6 Lokale Logdatei für den Fehlerfall 11

6 opsi Feature *SilentInstall* **13**

6.1 Vorbedingungen für die Silent Installation 13

6.2 Überblick über das SilentInstall-Feature 13

6.3 Auslösen der Silent Installation 14

6.4 Konfigurationen des opsi-Feature: *SilentInstall* 14

7 Schutz Ihrer Änderungen vor Updates: Das custom Verzeichnis **17**

8 Anpassungen bei der Treiberintegrationsmethode byAudit **18**

9 HostControlSafe-Backend **19**

10 opsi-winst 4.11.3.5 **20**
10.1 Skinnable opsi-winst . . . 20
10.1.1 Fenster Modus / Skin . . . 20
10.2 Kommandos zur Steuerung des Logging . . . 21
10.3 Umgang mit Zahlen . . . 21
10.4 PatchTextFile-Sektionen . . . 24
10.4.1 Aufrufparameter . . . 24
10.5 LinkFolder-Sektionen . . . 24
10.6 ExecWith Sektionen . . . 25
10.6.1 Aufrufparameter (Modifier) . . . 25
10.7 WinBatch-Sektionen . . . 25
10.7.1 Aufrufparameter (Modifier) . . . 25
10.8 LDAPsearch Sektion . . . 26
10.8.1 LDAPsearch Sektion Syntax . . . 26
10.8.2 Beispiele . . . 26

11 opsi-configed 4.0.3.1 **28**
11.1 Host-Parameter in der Client- und der Serverkonfiguration . . . 28
11.2 Logdateien . . . 28

12 Sonstiges **30**
12.1 Verweise auf das opsi-manual . . . 30
12.1.1 Freischaltung kostenpflichtiger Module . . . 30
12.1.2 Konfiguration über den Webservice (Host-Parameter) . . . 30
12.2 Changelogs: . . . 30
12.2.1 Changelog opsi-winst . . . 30
12.2.2 Changelog opsi-winst-test . . . 31
12.2.3 Changelog jedit . . . 32
12.2.4 Changelog opsi-adminutils . . . 32
12.2.5 Changelog opsi-template . . . 33
12.2.6 Changelog windows netboot products . . . 33
12.2.7 Changelog opsi-client-agent . . . 33
12.2.8 Changelog python-opsi . . . 35
12.2.9 Changelog opsipxeconfd . . . 35
12.2.10 Changelog opsiconfd . . . 36
12.2.11 Changelog opsi-utils . . . 36
12.2.12 Changelog opsi-linux-bootimage . . . 36
12.2.13 Changelog opsi-depotserver . . . 37
12.2.14 Changelog opsi4ucs . . . 37
12.2.15 Changelog opsi-configed . . . 38

Kapitel 1

Copyright

Das Copyright an diesem Handbuch liegt bei der uib gmbh in Mainz.

Dieses Handuch ist veröffentlicht unter der creative commons Lizenz
Namensnennung - Weitergabe unter gleichen Bedingungen (by-sa).

Eine Beschreibung der Lizenz finden Sie hier:
http://creativecommons.org/licenses/by-sa/3.0/de/

Der rechtsverbindliche Text der Lizenz ist hier:
http://creativecommons.org/licenses/by-sa/3.0/de/legalcode

Die Software von opsi ist in weiten Teilen Open Source.
Nicht Open Source sind die Teile des Quellcodes, welche neue Erweiterungen enthalten die noch unter Kofinanzierung stehen, also noch nicht bezahlt sind.
siehe auch: http://uib.de/www/kofinanziert/index.html

Der restliche Quellcode ist veröffentlicht unter der GPLv3 und wird im Rahmen von opsi 4.0.3 auf AGLv3 umgestellt:

Der rechtsverbindliche Text der GPLv3 Lizenz ist hier:
http://www.gnu.org/licenses/gpl.html

Deutsche Übersetzung:
http://www.gnu.de/documents/gpl.de.html

Der rechtsverbindliche Text der AGPLv3 Lizenz ist hier:
http://www.gnu.org/licenses/agpl-3.0-standalone.html

Deutsche Infos zur AGPL: http://www.gnu.org/licenses/agpl-3.0.de.html

Für Lizenzen zur Nutzung von opsi im Zusammenhang mit Closed Source Software kontaktieren Sie bitte die uib gmbh.

Die Namen *opsi*, *opsi.org*, *open pc server integration* und das opsi-logo sind eingetragene Marken der uib gmbh.

Kapitel 2

Übersicht der Neuerungen

Das opsi Service Releases 4.0.3 weist eine Fülle von Neuerungen und Detailverbesserungen auf. Hier eine Übersicht:

- opsi Erweiterung: Install-on-shutdown (Kofinanzierungsprojekt)
- neues opsi Feature: Silent-Install
- Lokalisierung für Italienisch und Dänisch
- Freigabe von opsi 4.0.3 für folgende Distributionen:
  - UCS 3.1
  - Ubuntu 12.10

Die entsprechenden Installationsanleitungen finden Sie wie immer im *opsi-getting-started* Handbuch

- Abkündigung zum 31.03.2013 der Unterstützung für:
  - CentOs 5
  - RedHat 5
  - OpenSuse 11.3 / 11.4
- opsi-client-agent
  - Umfangreich überarbeitete Version
  - Freigabe der opsi-Erweiterung: Dynamische Depotauswahl
  - opsi Erweiterung: Install-on-shutdown
  - opsi Feature: Silent-Install
  - Verbessertes Customizing
  - Anpassungen für Windows 8 Support
  - Überarbeitetes Sessionhandling: Bessere Identifikation der Benutzerkonten.
  - Verbesserungen im WAN-Support
  - Mehrere kleinere Fixes
- opsi Serverkomponenten
  - Verbesserungen im DHCP-Backend
  - Neues HostControlSafe-Backend
  - Neuerstellung des Webservice-SSL-Zertifkats über opsi-setup

- Windows Netboot Produkte
  - Windows 8 / Server 2012 Unterstützung
  - Verbesserungen an byAudit-Verfahren
  - WinPE-Partitionsgröße konfigurierbar
  - Neues Property data_partition_create : Soll eine Datenpartition erzeugt werden, wenn noch Platz verfügbar ist (true/false ; default: true).
  - UEFI Support
  - Unterstützung für Softwareraid 1: neues Property *useRaid1* behandelt 2 vorhandene Platten identisch.
- opsi-winst:
  Eine Fülle von Erweiterungen, welche weiter unten beschrieben sind.
- Ankündigung Lizenzwechsel GPLv3 zu AGPLv3:
  Im Laufe des opsi 4.0.3 Releases werden wir zur AGPLv3 wechseln.
  Dieser Lizenzwechsel betrifft nur Firmen, welche opsi als Bestandteil einer Closed Source Lösung vertreiben. Wer eine Lizenz für den Vertrieb von opsi als Bestandteil einer Closed Source Lösung benötigt, möge uns kontaktieren.

Für Details lesen Sie bitte das opsi-v403-releasenotes Hanbuch bzw. die überarbeiteten opsi Handbücher.

Kapitel 3

Abkündigung

In diesem Kapitel werden die Abkündigung aufgelistet. Diese Distributionsversionen werden aus verschiedenen Gründen nicht mehr weiter von opsi unterstützt.

3.1 Abkündigung der Python 2.4 Unterstützung

Mit diesem Featurepack-Release wird offiziell die Unterstützung für Python 2.4 eingestellt. Dies betrifft Python 2.4 Distributionen: Redhat 5 und Centos 5. Die betroffenen Repositories werden am 31.03.2013 geschlossen. Die Paketstände zu diesem Termin, werden auf download.uib.de im Archiv veröffentlicht. Für Betroffene opsi-Systeme empfehlen wir den Upgrade auf die Neuen Distributionsversionen, die noch Unterstützt werden.

3.2 Abkündigung: Distributionsversionen

Folgende Systeme werden mangels Maintainance vom Hersteller abgekündigt:

- openSuse 11.3
- Ubuntu 10.10
- Ubuntu 11.04
- Ubuntu 11.10

Auch die oben genannten Repositories werden am 31.03.2013 geschlossen. Für betroffene Installationen empfehlen wir dringend ein OS-Upgrade durch zu führen. Allerdings sollten Sie vorher prüfen, ob die Version zu der Sie aktualisieren wollen auch von opsi unterstützt wird.

Kapitel 4

Hinweise zur Installation

Die Produkte welche im Rahmen dieses Releases veröffentlicht werden, sind in etlichen Teilen voneinander abhängig. Sie sollten daher nicht versuchen nur Teile zu installieren.

Sie sollten dabei zuerst den Server updaten und danach die opsi-Produkte.

Die Installation erfordert keine besonderen Arbeiten. Sie erfolgt im Rahmen der normalen Updates ihres Servers und der opsi-Produkte.

Sie sollten dabei zuerst den Server updaten und danach die opsi-Produkte. Dies erledigt der opsi-product-updater:

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

Sollten Sie eine Multidepot Umgebung haben, so machen Sie zunächst das Upgrade auf Ihrem config-server, bevor Sie die Depots upgraden.

Achtung
Seit opsi 4.0.3 haben die opsi Server Pakete keine Abhängigkeit mehr zu den Paketen *dhcp Server* und *java Laufzeitumgebung*.
Diese Änderung ist sinnvoll, da die meisten opsi Anwender einen anderen DHCP-Server betreiben und den opsi-configed nicht diekt auf dem Server ausführen.
Sollten Sie eines diser Pakete auf Ihrem Server benötigen, weil der opsi Server auch DHCP Server sein soll oder der opsi-configed auch auf dem opsi-server laufen soll, so müssen Sie diese Pakete eventuell nach dem upgrade auf opsi 4.0.3 erneut installieren, da sie aufgrund der weggefallenen Abhängigkeit automatisch deinstalliert wurden.

Kapitel 5

opsi Installation beim Shutdown

5.1 Einführung

Standardmäßig wird die Installation von opsi Software-Paketen beim Hochfahren des Clients gestartet. Der Anwender muss dann auf die Beendigung der Software-Installationen warten, bevor er sich am Rechner anmelden kann. Von daher kann es wünschenswert sein, die Software-Installationen vorwiegend beim Runterfahren des Clients durchzuführen.

Das opsi Modul zur Installation beim Shutdown stellt genau diese Funktionalität zur Verfügung. Individuell für bestimmte Clients kann die Installation beim Shutdown aktiviert werden.

5.2 Vorbedingungen für die Installation beim Shutdown

Das Modul Installation beim Shutdown kann auf Clients ab **Windows XP** eingesetzt werden. Die erforderliche Grundfunktionalität ist Bestandteil des Paketes *opsi-client-agent*. Da das Modul sich derzeit noch in der Kofinanzierungsphase befindet, ist zur Freischaltung eine entsprechende modules-Datei erforderlich. Weitere Details hierzu finden Sie in Abschnitt 12.1.1 sowie unter dem Link kofinanzierte opsi Erweiterungen.

Das Paket *opsi-client-agent* muss mindestens die Version 4.0.2.3-2 haben mit einem `opsiclientd` mindestens der Version 4.0.75.

Einschränkungen und Besonderheiten bestehen in folgendem Zusammenhang:

- WAN-Erweiterung: das On-Shutdown-Modul ist derzeit nur auf Clients anwendbar, die nicht mit der WAN-Erweiterung betrieben werden. Bei der WAN-Erweiterung ist teilweise die lokale und nicht die serverseitige Konfiguration relevant, dies kollidiert mit der Zustandssteuerung für die Installation beim Shutdown.
- Group Policies: da ein Teil des Mechanismusses für die Installation beim Shutdown auf Shutdown-Skripten per Local Group Policy beruht, werden diese Einstellungen eventuell durch übergeordnete Group Policies überschrieben. In diesem Fall sollten die notwendigen Einstellungen in die übergeordneten Group-Policies aufgenommen werden. Siehe hierzu Abschnitt 5.4.2.
- Windows Home-Edition: Windows Home enthält nicht die erforderlichen Group Policy Shutdown-Skript-Mechanismen. Von daher kann die On-Shutdown-Installation auf Windows-Home-Edition nicht eingesetzt werden.
- Windows 2000: Auf Windows 2000 beträgt die maximale Wartezeit für Shutdown-Skripte 10 Minuten, danach wird die Installation durch Windows automatisch abgebrochen. Aus diesem Grund ist das Modul nicht auf Windows 2000 Clients einsetzbar.

5.3 Inbetriebnahme der Installation beim Shutdown

Da die Grundfunktionalität für die Installation beim Shutdown bereits im aktuellen `opsi-client-agent`-Paket enthalten ist, muss zur Freischaltung und Inbetriebnahme lediglich eine gültige `modules`-Datei erworben und auf den Server gelegt werden.

Dann kann für geeignete Clients (siehe Abschnitt 5.2) der Produktschalter `on_shutdown_install` des Paketes `opsi-client-agent` auf `on` und der `opsi-client-agent` für diese Clients auf setup gesetzt werden.

Weitere Einstellungen sind im Normalfall nicht notwendig.

Die Installation beim Shutdown wird zusätzlich zur Installation beim Hochfahren ausgeführt. Dies ist im Normalfall sinnvoll, da somit auch Rechner, die längere Zeit ausgeschaltet waren (z.B. nach Urlaub des Anwenders), vor der Benutzung die neuesten Security-Updates bekommen. Falls gewünscht, kann die Installation beim Hochfahren abgeschaltet werden, siehe Abschnitt 5.4.5. Angefangene Installationen werden aber auf jeden Fall beim Hochfahren des Rechners fortgesetzt.

- Windows unterscheidet beim Herunterfahren systemtechnisch nicht zwischen einem Shutdown und einem Reboot. Die Installation beim Shutdown wird also sowohl beim Shutdown, als auch beim Reboot ausgeführt und es ist nicht möglich, diese beiden Fälle bei der Ausführung zu unterscheiden. Wurde der Rechner nicht als Reboot, sondern als Shutdown heruntergefahren, lässt Windows keine Umwandlung des Shutdown in einen Reboot zu. Falls einzelne Software-Pakete eine mehrphasige Installation mit Reboot benötigen, wird die Installation erst beim nächsten Start des Clients fortgesetzt.

5.4 Technisches Konzept

Die folgenden Erläuterungen dienen dem besseren Verständnis der technischen Zusammenhänge für spezielle Konfigurationsvarianten sowie der Untersuchung im Fehlerfall. Im Normalfall werden alle erforderlichen Einstellungen vom Paket `opsi-client-agent` durchgeführt.

5.4.1 Überblick

Die Installation beim Shutdown basiert auf dem Zusammenspiel verschiedener System-Komponenten. Ein wesentlicher Bestandteil ist die Nutzung des Windows Shutdown-Skript-Mechanismus per Local Group Policy. Shutdown-Skripte ermöglichen die Durchführung von Tasks genau zu dem Zeitpunkt des Shutdown-Vorgangs, an dem der Benutzer bereits abgemeldet ist und alle Benutzer-Tasks beendet sind, aber noch alle Systemdienste laufen.

Per Shutdown-Skript wird ein opsi-Task ausgeführt, der über den opsi Systemdienst *opsiclientd* eine Installation anstößt und auf deren Beendigung wartet. Erst dann wird das System ganz runter gefahren. Systemtechnisch wird hier nicht zwischen einem Shutdown und einem Reboot unterschieden, so dass die Installation auch bei einem Reboot ausgelöst wird.

Der opsi-Client-Systemdienst *opsiclientd* ist für die Aktionsart `on_shutdown` konfiguriert, die die Installation handhabt. Falls für die Installation Reboots benötigt werden, ist die precondition `installation_pending` für die korrekte Steuerung des Ablaufs zuständig. Falls während der Installation im Shutdown ein Reboot benötigt wird, führt die Precondition `installation_pending` (unabhängig davon, ob `gui_startup` aktiviert ist oder nicht) zu einer direkten Fortführung der Installation beim nächsten Hochfahren des Systems, gegebenenfalls auch mit weiteren Reboots. Im Zustand `installation_pending` wird bei eventuell erforderlichen weiteren Reboots keine Installation beim Shutdown ausgeführt, da ansonsten zwischen der Installation beim Hochfahren und der Installation beim Runterfahren kein Reboot liegen würde. D.h. es wird bis zum Abschluss der aktuellen Installation beim Hochfahren des Systems weiter installiert, aber nicht beim Shutdown, da sonst kein Reboot zwischen den einzelnen Installationsphasen liegen würde.

Im Folgenden werden die beiden Komponenten im Detail beschrieben.

5.4.2 Durchführung per Shutdown-Skript

Über entsprechende Registry-Einträge wird per Local Group Policy beim Herunterfahren des Systems ein Shutdown-Skript ausgeführt, das die Installation anstößt. Die Registry-Einträge entsprechen den Einstellungen, wie sie auch mit dem Group Policy-Editor `gpedit.msc` erzeugt werden können.

Dies erzeugt per Group Policy-Editor den Eintrag eines Shutdown-Skriptes:

- Richtlinien für Lokaler Computer
- Computerkonfiguration
- Windows-Einstellungen
- Skripts (Start/Herunterfahren)
- Herunterfahren
- Skripts - Hinzufügen - Durchsuchen
- C:\Programme\opsi.org\opsi-client-agent\on_shutdown\doinstall32.cmd (bzw. doinstall64.cmd für 64Bit-Systeme)

Damit das System mit dem Shutdown wartet, bis die Installation vollständig abgeschlossen ist, wird die Wartezeit für Shutdown-Skripte auf unendlich gesetzt (0 Sekunden):

- Richtlinien für Lokaler Computer
- Computerkonfiguration
- Administrative Vorlagen
- System - Skripts
- Maximale Wartezeit für Gruppenrichtlinienskripts
- Einstellung - Aktiviert - Sekunden: 0

Das eingetragene Shutdown-Skript `doinstall32.cmd` bzw. `doinstall64.cmd` wechselt das Arbeitsverzeichnis und löst das *on_shutdown*-Event aus:

```
echo Start opsi product installation ...
cd "%ProgramFiles%\opsi.org\opsi-client-agent"
opsiclientd_shutdown_starter.exe on_shutdown
```

bzw. für 64Bit-Systeme:

```
echo Start opsi product installation ...
cd "%ProgramFiles(x86)%\opsi.org\opsi-client-agent"
opsiclientd_shutdown_starter.exe on_shutdown
```

Der `opsiclientd_shutdown_starter` wartet auf die Beendigung der Installation, so dass der System Shutdown so lange aufgehalten wird.

5.4.3 Registry-Einträge für die Ausführung des Shutdown-Skripts

Diese Registry-Einträge werden über das Setup des `opsi-client-agent` Paketes gesetzt und führen zur Ausführung des angegebenen Shutdown-Skriptes auf WinXP / 32Bit. Für 64Bit-Systeme ist der Skriptname `doinstall64.cmd` (statt `doinstall32.cmd`).

```
[HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\Group Policy\State\Machine\Scripts\Shutdown\0]
"GPO-ID"="LocalGPO"
"SOM-ID"="Local"
"FileSysPath"="C:\\WINDOWS\\System32\\GroupPolicy\\Machine"
"DisplayName"="opsi shutdown install policy"
"GPOName"="opsi shutdown install policy"

[HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\Group Policy\State\Machine\Scripts\Shutdown\0\0]
"Script"="C:\\Programme\\opsi.org\\opsi-client-agent\\on_shutdown\\doinstall32.cmd"
"Parameters"=""
"ExecTime"=hex(b):00,00,00,00,00,00,00,00,00,00,00,00,00,00,00,00

[HKEY_LOCAL_MACHINE\SOFTWARE\Policies\Microsoft\Windows\System\Scripts\Shutdown\0]
"GPO-ID"="LocalGPO"
"SOM-ID"="Local"
"FileSysPath"="C:\\WINDOWS\\System32\\GroupPolicy\\Machine"
"DisplayName"="opsi shutdown install policy"
"GPOName"="opsi shutdown install policy"

[HKEY_LOCAL_MACHINE\SOFTWARE\Policies\Microsoft\Windows\System\Scripts\Shutdown\0\0]
"Script"="C:\\Programme\\opsi.org\\opsi-client-agent\\on_shutdown\\doinstall32.cmd"
"Parameters"=""
"ExecTime"=hex:00,00,00,00,00,00,00,00,00,00,00,00,00,00,00,00

[HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\policies\system]
"MaxGPOScriptWait"=dword:00000000
```

Dies sind die entsprechenden Registry-Einträge für Win6 64Bit (Vista / Win7 / Win8):

```
[HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\Group Policy\State\Machine\Scripts\Shutdown\0]
"GPO-ID"="LocalGPO"
"SOM-ID"="Local"
"FileSysPath"="C:\\WINDOWS\\System32\\GroupPolicy\\Machine"
"DisplayName"="opsi shutdown install policy"
"GPOName"="opsi shutdown install policy"

[HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\Group Policy\State\Machine\Scripts\Shutdown\0\0]
"Script"="C:\\Programme\\opsi.org\\opsi-client-agent\\on_shutdown\\doinstall32.cmd"
"Parameters"=""
"ExecTime"=hex(b):00,00,00,00,00,00,00,00,00,00,00,00,00,00,00,00

[HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\Group Policy\Scripts\Shutdown\0]
"GPO-ID"="LocalGPO"
"SOM-ID"="Local"
"FileSysPath"="C:\\Windows\\System32\\GroupPolicy\\Machine"
"DisplayName"="opsi shutdown install policy"
"GPOName"="opsi shutdown install policy"
"PSScriptOrder"=dword:00000001

[HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\Group Policy\Scripts\Shutdown\0\0]
"Script"="C:\\Program Files (x86)\\opsi.org\\opsi-client-agent\\on_shutdown\\doinstall64.cmd"
"Parameters"=""
"IsPowershell"=dword:00000000
"ExecTime"=hex:00,00,00,00,00,00,00,00,00,00,00,00,00,00,00,00

[HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\policies\system]
"MaxGPOScriptWait"=dword:00000000
```

5.4.4 Erforderliche Konfiguration des opsiclientd

Der opsi-Client-Systemdienst *opsiclientd* hat für das neue Event *on_shutdown* zusätzliche Standard-Einträge in der Konfigurationsdatei *opsiclientd.conf* bekommen. Hier alle relevanten Einträge:

```
[event_gui_startup]
active = True

[event_gui_startup{installation_pending}]
active = True

[event_on_shutdown]
active = False

[event_on_shutdown{installation_pending}]
active = False

[precondition_installation_pending]
installation_pending = true
```

Die Precondition `installation_pending` zeigt an, ob noch eine Installation im Gange ist. Falls bei Beendigung des Skriptes immer noch der Zustand `installation_pending` auf true gesetzt ist, bedeutet dies, dass die aktuelle Installation einen Reboot benötigt und noch nicht abgeschlossen ist. Im Normalbetrieb ohne Installation bei Shutdown sind die Sektionen für die neue Aktion `event_on_shutdown` deaktiviert.

Für einen Client mit aktivierter Installation bei Shutdown ist dies die erforderliche Konfiguration:

```
[event_gui_startup]
active = True

[event_gui_startup{installation_pending}]
active = True

[event_on_shutdown]
active = True

[event_on_shutdown{installation_pending}]
active = False

[precondition_installation_pending]
installation_pending = true
```

Der einzige Unterschied ist hier also

```
[event_on_shutdown]
active = True.
```

Diese Einstellung wird über den Produktschalter `on_shutdown_install` des Paketes `opsi-client-agent` gesteuert.

Die Precondition `precondition_installation_pending = true` besagt, dass eine angefangene Installation noch nicht beendet ist. Dieser Zustand bleibt über einen oder mehrere Reboots hinweg so lange bestehen, bis die Installation abgeschlossen ist. Wenn die unvollständige Installation einen Reboot benötigt, wird beim nächsten Hochfahren durch die Konfiguration `[event_gui_startup{installation_pending}] active = True` die begonnene Installation beim Hochfahren des Systems fortgesetzt. Diese Einstellung darf nicht verändert werden, da eine angefangene Installation auf jeden Fall beendet werden muss, bevor der Benutzer sich anmelden darf.

Der Eintrag `[event_on_shutdown{installation_pending}] active = False` muss auch in jedem Fall auf False bleiben, da bei einer angefangenen Installation ansonsten kein Reboot wäre zwischen den Installationsphasen beim Startup und beim Shutdown.

Sobald die laufende Installation abgeschlossen ist, wird die Precondition `installation_pending = false` und somit auch wieder die Installation im Shutdown aktiv.

5.4.5 Spezielle Konfiguration der Installation bei Shutdown

Im Normalfall ist zur Inbetriebnahme der Installation bei Shutdown nur, wie in Abschnitt 5.3 beschrieben, eine für dieses Modul gültige modules-Datei notwendig. Dann kann für geeignete Clients mit dem `opsi-client-agent`-Produktschalter `on_shutdown_install` die Installation beim Shutdown aktiviert werden.

Die Installation beim Hochfahren des Systems bleibt standardmässig auch aktiv. Somit ist gewährleistet, dass auch ein längere Zeit ausgeschalteter Client (z.B. nach dem Urlaub des Benutzers) auf jeden Fall die neuesten Versionsstände bekommt, bevor der Benutzer sich anmelden kann.

Sollte dies nicht erwünscht sein, kann die Installation beim Hochfahren deaktiviert werden. Da die Konfiguration des *opsi-client-agents* auch zentral über den Webservice erfolgen kann (siehe: Abschnitt 12.1.2), ist zu empfehlen, dass folgender *Host-Parameter* angelegt wird:

- `opsiclientd.event_gui_startup.active` (boolean, default: `true`)

Über diesen *Host-Parameter* kann dann das gui_startup-Event Client-spezifisch aktiviert bzw. deaktiviert werden. Die *Host-Parameter* können über den *opsi-configed* oder *opsi-admin* angelegt werden.

Zum Anlegen des *Host-Parameter* über *opsi-admin* ist der folgende Befehl auf dem *opsi-configserver* auszuführen:

```
opsi-admin -d method config_createBool opsiclientd.event_gui_startup.active "gui_startup active" true
```

Der Standard-Wert true entspricht hierbei dem Wert in der mitgelieferten `opsiclientd.conf`.

Wenn für einen Install_on_shutdown-Client die Installation im Startup deaktiviert werden soll, wird der entsprechende *Host-Parameter* wie folgt konfiguriert:

- `opsiclientd.event_gui_startup.active`: `false`

Dies sollte aber nur in begründeten Ausnahmefällen geschehen. Die Einstellungen mit der Zusatzbedingung *installation_pending* sollten unter keinen Umständen geändert werden, es müssen hier immer die default-Werte verwendet werden, um eine korrekte Steuerung des Ablaufs zu gewährleisten.

Zum Setzen des *Host-Parameter* über *opsi-admin* ist der folgende Befehl auf dem *opsi-configserver* auszuführen (im Beispiel für einen Client mit der opsi-host-Id `myclient.domain.de`):

```
opsi-admin -d method configState_create opsiclientd.event_gui_startup.active myclient.domain.de false
```

Diese Konfiguration hat zur Folge, dass beim Start des Rechners kein Verbindungsaufbau zum *opsi-configserver* und somit keine Installation stattfindet. Ausser bei einer angefangenen Installation, die durch die Zusatzbedingung `installation_pending` angezeigt wird. In diesem Fall wird durch die Einstellung `event_gui_startup{installation_pending}` beim Systemstart die angefangene Installation weiter fortgesetzt. Wenn ein weiterer Reboot erforderlich ist, wird durch die Einstellung `event_on_shutdown{installation_pending}` (die ebenfalls nicht verändert werden darf) verhindert, dass zusätzlich auch beim Shutdown die Installation weitergeführt wird. Ansonsten wäre kein System-Neustart zwischen den einzelnen Installationsphasen.

5.4.6 Lokale Logdatei für den Fehlerfall

Beim Install-On-Shutdown wird eine lokale Logdatei angelegt:

- `C:\opsi.org\tmp\doinstall.log`

mit normalerweise folgendem Inhalt:

```
doinstall32.cmd started
Aktuelles Datum: 29.01.2013
Geben Sie das neue Datum ein: (TT-MM-JJ) Backend connected.
modules: passed first checkpoint.
Modules file signature verified (customer: my company)
Check completed.
Event fired
Task completed.
```

Diese Logdatei wird jedes Mal neu erzeugt und kann im Fehlerfall überprüft werden.

Kapitel 6

opsi Feature *SilentInstall*

Das SilentInstall-Feature bietet opsi-Administratoren die Möglichkeit, Software bei Anwendern zu installieren, ohne dass diese bei Ihrer Arbeit gestört werden. Die folgende Dokumentation beschreibt die Besonderheiten dieser Erweiterung und bietet einen Leitfaden zur Konfiguration der neuen Installationsmethode.

6.1 Vorbedingungen für die Silent Installation

Um dieses Feature benutzten zu können, braucht man opsi mindestens in der Version 4.0.3. Hauptsächlich wird der *opsi-client-agent* ab Version 4.0.3.1 benötigt.

6.2 Überblick über das SilentInstall-Feature

Die SilentInstall Methode bietet die Möglichkeit eine festgelegte Liste von Produkten zu installieren ohne dass der Anwender seine Arbeit unterbrechen muss. Der große Unterschied zur Installation per onDemand-Event (push Installation) besteht darin, dass bei dieser Installationsmethode nichts angezeigt wird.

Alle Ausgaben werden unterdrückt und auf keinem Desktop angezeigt. Diese Art der Installation birgt natürlich auch Gefahren. Bei einem Problem wie z.B. ein Syntaxfehler im *opsi-winst* Skript, hat man keine Möglichkeit Einfluss auf den Zustand zu nehmen. Der Grund dafür liegt daran, dass eventuell auftauchende Dialogfenster nicht angezeigt werden. Die Folge in diesem Fall wäre, dass der *opsi-winst* nicht beendet wird und damit das Event nicht zum Abschluss kommen kann und eventuell auftretende Events blockiert. Um diesen "Worstcase" zu vermeiden, wird die maximale Installationszeit in Form einer Timeout-Einstellung festgelegt. Dieser Wert muss eventuell angepasst werden, wenn im Vorfeld klar ist, dass die Installation definitiv länger dauert. Näheres dazu wird im Konfigurationsabschnitt dieser Dokumentation beschrieben.

Eine weitere und sehr wichtige Besonderheit dieses Feature ist die fest vorgegebene Liste der Produkte. Es wird zwar Kontakt zum Service hergestellt, aber anders als üblich werden die ActionRequests vom Service ignoriert. Die zu installierende Software wird über eine Konfiguration im *opsi-client-agent* fest vorgegeben. Diese Produkte werden abgearbeitet und müssen dafür nicht extra auf setup gestellt werden. Es wird immer "setup" ausgeführt. Wie in diesem Fall üblich wird nach dem setup-Skript das update-Skript ausgeführt, sofern eines hinterlegt wurde. `Abhängigkeiten werden nicht aufgelöst.` Entweder man sollte komplett auf Pakete mit Abhängigkeiten in diesem Modus verzichten oder man muss dafür sorgen, dass die Produkte inklusive der Abhängigkeiten mit in die Bearbeitungsliste konfiguriert werden sollten. Abschliessend wird der Installationstatus und die Installationslogs zum Service übertragen.

Zusammenfassend wird empfohlen für diesen Modus nur Produkte auszuwählen, die folgende Kriterien erfüllen:

- kleine Pakete oder Installationen
- wenig Last produzieren: Einige Softwareinstallation, unter anderem auch viele MSI Installationen belegen die kompletten Ressourcen des Clients. Somit kann es passieren, dass der Client des Anwenders nicht mehr oder zeitweise sehr träge reagiert.

- in einer festgelegten Zeit installierbar sein: Das Event dieser Erweiterung ist in der Standardkonfiguration mit einem Timeout von 300 Sekunden angelegt. Sollte die Installation in dieser Zeit nicht abgeschlossen sein, wird der *opsi-winst*-Prozess beendet, damit das Event abgeschlossen werden kann.
- Software die einen Reboot bei der Installation benötigt sollte vermieden werden. In der Standardkonfiguration ist dieses Event so konfiguriert, dass angeforderte Reboots nicht ausgeführt werden. Ohne diese Absicherung würde der *opsi-client-agent* ohne Vorwarnung den Client Rebooten. Sollte ein Anwender angemeldet sein, droht Datenverlust. Die Folge daraus könnte sein, dass man eventuell Software im Hintergrund installiert, die in diesem Zustand nicht funktioniert.

In der Standardkonfiguration wird mit diesem Modus swaudit und hwaudit über diese Methode installiert. Die Inventarisierungsprodukte von opsi erfüllen alle oben genannte Kriterien und eignen sich deshalb für diesen Modus. Mit der Standardkonfiguration wird die opsi Hard- und Software-Inventarisierung auf Wunsch ausgeführt, ohne dass die Pakete erst im *opsi-configed* auf setup gestellt werden müssen. Mit dieser Methode kann man bei Bedarf die Informationen in Echtzeit beschaffen. Denkbar wären auch Konfigurationsprodukte, die zum Beispiel Reparaturarbeiten erledigen oder dafür sorgen, dass ein festgelegter Sollzustand der Clienteinstellungen regelmäßig wiederherstellt wird.

6.3 Auslösen der Silent Installation

Dieses Event wird nicht von alleine ausgelöst wie andere Events. Deshalb gibt es zwei Möglichkeiten dieses Event zu aktivieren.

Die erste Möglichkeit wäre das Event über den opsi-Webservice auszulösen, wie z.B.:

```
opsi-admin -d method hostControl_fireEvent silent_install client.domain.local
```

Dies ermöglicht es auch, diese Schritte in eigenen Skripten unterzubringen. Mit diesen Skripten kann man dieses Feature steuern und z.B.: mit einem at-Job kombinieren, um die Auslösung des Events zu planen.

Als Alternative dazu kann das Event über ein Timer-Event nach einem festgelegten Intervall gestartet werden. In der Standardkonfiguration ist diesem Event ein Intervall von 6 Stunden vorgegeben. Dieser Wert geht davon aus, dass ein Arbeitsplatz-Client im Durchschnitt 8 Stunden eingeschaltet ist. Mit dieser Annahme würde dieses Event jeden Arbeitstag aber nur einmal am Tag ausgeführt werden. Näheres zu der Konfiguration und Aktivierung dieses Events wird im Konfigurationsabschnitt erläutert.

6.4 Konfigurationen des opsi-Feature: *SilentInstall*

Im folgenden werden die Standardkonfiguration dieses Features erläutert. Als erstes gibt es folgendes Event in der standard opsiclientd.conf. Das Listing zeigt nur die wichtigen Optionen:

Standard Event SilentInstall:

```
[event_silent_install]
process_shutdown_requests = false
action_processor_productIds = swaudit,hwaudit
action_processor_command = %action_processor.command% /productlist %action_processor_productIds% /silent
action_processor_desktop = winlogon
action_processor_timeout = 300
```

- action_processor_productIds
  - Diese Option ist eine der wichtigen Neuerungen für die Eventsteuerung. Mit dieser Option kann man allen Events, die Produktaktionen ausführen eine Liste von Produkten übergeben. Die Konfiguration für diese Option muss als kommaseparierte Liste angegeben werden.
- process_shutdown_request = false

  - Sorgt dafür, dass ein Reboot, welches vom *opsi-winst* angefordert wurde, nicht ausgeführt wird.

- action_processor_command
  - Hier wird der Aufruf vom *opsi-winst* vorbereitet.
- action_processor_desktop
  - In dieser Option wird beeinflusst auf welchen Desktop die Gui vom *opsi-winst* angezeigt werden soll.
- action_processor_timeout
  - Hier wird der Zeitraum festgelegt, ab wann der *opsi-winst*-Prozess beendet wird

Das zweite Event ist das Timer Event, welches vorgesehen ist, um das Event nach einem festgelegten Intervall auszulösen:

Standard Timer Event für SilentInstall

```
[event_timer_silentinstall]
super = silent_install
type = timer
active = false
interval = 21600
```

- super
  - Diese Option beschreibt von welchem Event dieses Timer-Event erbt. Standardmäßig wird vom Event silent_install geerbt.
- tpye
  - Diese Option legt fest, dass diese Event-Konfiguration ein Timer-Event ist.
- active
  - Standardmäßig ist dieses Event deaktiviert. Um es zu aktivieren muss diese Option auf *true* gestellt werden.
- interval
  - Mit dieser Option wird der Intervall festgelegt. Nach Ablauf dieser Zeit wird das Event ausgelöst. Dieser Wert ist Standardmäßig auf 6 Stunden voreingestellt. Diese Zeit sollte wie alle Timerintervalle nicht zu gering gewählt werden, da ansonsten das Event ständig aktiv wird und ggf. andere Aktionen damit blockiert werden. Man sollte das Intervall aber auch nicht zu hoch setzen, da der *opsi-client-agent* über das gesamte Intervall durchlaufen muss, um das Event aus zu lösen. Wenn innerhalb des Intervalls der Client selbst oder der *opsi-client-agent* neu gestartet wird, wird dieses Event nie ausgelöst.

Es ist auch denkbar, das Event SilentInstall beim Eintreten eines Systemereignisses auszulösen. Dafür muss zu dem vorhandenen Event noch die Option *wql* konfiguriert werden. Wie man das genau umsetzt kann man beim event_net_connection nachschauen. Sollte die *wql* Option eingesetzt werden, wird empfohlen das Event mit *active = false* standardmäßig auszuschalten, damit dieses Event bei Bedarf aktiviert werden kann.

Um das Event wie vorgesehen per Timer zu starten, genügt es eigentlich einen Hostparameter zu setzen. Dazu muss wie gewohnt erst eine Default-Konfiguration angelegt werden. In diesem Fall reicht es aus, das entsprechende Timer-Event zu aktiveren.

Zunächst wird die Standardoption angelegt, im folgenden werden die nötigen *Host-Parameter* über *opsi-admin* angelegt. Man kann diese Konfiguration auch über den *opsi-configed* erledigen:

```
opsi-admin -d method config_createBool opsiclientd.event_timer_silentinstall.active "event_timer_silentinstall active" \
    false
```

Damit ist für alle Clients dieses Event zunächst deaktiviert. Nun ist es möglich, dieses Event für einzelne Clients aktiv zu setzen:

```
opsi-admin -d method configState_create opsiclientd.event_timer_silentinstall.active silentclient.domain.de true
```

Um die Produkte, die abgearbeitet werden sollen, zu beeinflussen, muss man folgende Einstellungen vornehmen. Wenn man zum Beispiel statt *swaudit* und *hwaudit* das Produkt firefox installieren will, sollte man wieder vorgehen wie oben beschrieben. Zunächst muss man einen Defaultwert eintragen:

```
opsi-admin -d method config_createUnicode opsiclientd.event_silent_install.action_processor_productIds "\
    event_silent_install productIds" "swaudit,hwaudit"
```

Mit dieser Option wird für alle Clients Standardmäßig, die Produktliste für das Silent Install Event auf swaudit und hwaudit gesetzt. Um nun diese Einstellung für einen Client zu ändern kann man folgenden Befehl ausführen:

```
opsi-admin -d method configState_create opsiclientd.event_silent_install.action_processor_productIds client.domain.de "\
    firefox"
```

Damit ist es auch möglich einem Client eine andere Liste von Produkten zur Verarbeitung zu übergeben.

Kapitel 7

Schutz Ihrer Änderungen vor Updates: Das custom Verzeichnis

(Seit opsi-client-agent Version 4.0.3.1)

Möchten Sie Änderungen, welche Sie an den oben genannten Dateien durchgeführt haben, davor schützen, dass selbige beim Einspielen einer neuen Version des opsi-client-agenten verloren gehen, so können Sie hierfür das `custom` Verzeichnis (`/opt/pcbin/install/opsi-client-agent/files/opsi/custom`) verwenden. Das komplette `custom` Verzeichnis wird bei der Installation einer neuen Version des opsi-client-agenten gesichert und wieder hergestellt, so dass hier gemachte Änderungen bei einem Update nicht verloren gehen.

- `custom/cfg/config.ini`
  Diese custom Config-Datei wird als letzte eingelesen, somit gelten die in dieser Datei enthaltenen Werte statt den entsprechenden Einstellungen aus der Standard-Datei `cfg/config.ini`. Ausnahme: die Werte für `pckey` und `bootmode` werden nie aus dieser Datei geholt. Schreiben Sie in diese Datei **nur** die Werte, die Sie tatsächlich gegenüber dem Default geändert haben wollen.

- `custom/winstskin/*.*`
  Alle Dateien aus diesem Verzeichnis werden bei der Installation des opsi-client-agent auf dem Client nach `C:\Program Files (x86)\opsi.org\opsi-client-agent\custom\winstskin` kopiert. Falls vorhanden, wird (ab opsi-winst Version 4.11.3.4) dieses winstskin Verzeichnis bevorzugt verwendet. Es muss alle benötigten Dateien und Einträge vollständig enthalten, da die Inhalte des Standard-winstskin-Verzeichnisses bei Verwendung des `custom/winstskin/` ignoriert werden.

- `custom/notifier/*.*`
  Alle Dateien aus diesem Verzeichnis werden bei der Installation des opsi-client-agent auf dem Client nach `C:\Program Files (x86)\opsi.org\opsi-client-agent\notifier` kopiert und überschreiben dabei die entsprechenden aus dem serverseitigen Standard-Verzeichnis `files/opsi/dist/notifier/` stammenden Dateien.

- `custom/opsiclientd.conf`
  wird, falls vorhanden, bei der Installation des opsi-client-agent auf dem Client nach `C:\Program Files (x86)\opsi.org\opsi-client-agent\opsiclientd` kopiert und überschreibt dabei die aus dem serverseitigen `files/opsi/dist/opsiclientd/` Verzeichnis stammende `opsiclientd.conf`. Es werden also nicht einzelne Werte überschrieben, sondern die komplette Datei, die deshalb alle benötigten Einstellungen vollständig enthalten muss.
  Achtung:
  Die Anpassung der `opsiclientd.conf` über diese Methode wird nicht empfohlen. Verwenden Sie zur Konfiguration Ihrer Clients Hostparameter/Configs wie im Kapitel zum opsi-client-agent beschrieben. Die Verwendung einer `custom/opsiclientd.conf` ist nur bei extrem komplexen Anpassungen der opsiclientd.conf sinnvoll. Wenn Sie diese Methode anwenden, müssen Sie bei jedem Einspielen einer neuen opsi-client-agent Version auf dem Server überprüfen, ob in der Default Datei `files/opsi/dist/opsiclientd/opsiclientd.conf` Änderungen gemacht oder neue Einträge hinzugefügt wurden, welche Sie in Ihrer Version nachpflegen müssen. Also:
  Finger weg, es sei denn Sie wissen wirklich was Sie tun !

Kapitel 8

Anpassungen bei der Treiberintegrationsmethode byAudit

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.

Kapitel 9

HostControlSafe-Backend

Eine Besonderheit beim Standardverhalten von opsi4.0 Methoden ist, dass bei einer Abfrage ohne Angaben von Parametern, alle Objekte abgerufen werden. Beispielsweise gibt der Befehl "host_getObjects" ohne Parameter aufgerufen, alle Host-Objekte zurück. Dieses Verhalten ist im HostControl-Backend etwas problematisch. Besonders bei den beiden Befehlen: "hostControl_shutdown" und "hostControl_reboot". In diesen Fällen würde ein Aufruf dieser Methoden ohne Parameter alle Clients herunterfahren bzw. neustarten.

Deshalb gibt es mit Service Release opsi 4.0.3 an dieser Stelle zwei Änderungen:

- Die Methoden: "hostControl_shutdown" und "hostControl_reboot" brechen seit dieser Release mit dem opsi 4.0 Standardverhalten. Diese beiden Methoden geben nun eine Fehlermeldung zurück, wenn kein Parameter übergeben wurde.
- Es wurde ein neues Backend: HostControlSafe Backend eingeführt, welches standardmäßig bei allen Methoden eine Fehlermeldung ausgegeben wird, wenn keine korrekte Angaben zu den Clients übergeben wird. Um mit einer Methode vom HostControlSafe-Backend alle Clients an zu sprechen, kann man das *-Zeichen verwenden:

```
opsi-admin -d method hostControlSafe_shutdown *
```

Aus den oben genannten Gründen, empfehlen wir hostControlSafe-Methoden zu verwenden, wenn man etwas unsicher auf der Konsole ist oder neu anfängt sich mit den Servicemethoden zu beschäftigen.

Kapitel 10

opsi-winst 4.11.3.5

Die folgenden Beschreibungen sind Auszüge aus dem *opsi-winst* Handbuch.

"(...)" kennzeichnet die ausgelassenen Teile.

10.1 Skinnable opsi-winst

Ab Version 3.6 verfügt *opsi-winst* über eine veränderbare Oberfläche. Seine Elemente liegen im Unterverzeichnis winstskin des Verzeichnisses, in dem der ausgeführte *opsi-winst* liegt. Die editierbare Definitionsdatei ist skin.ini.

Seit Version 4.11.3.5 sucht der *opsi-winst* nach dem zu verwendenden Skin Verzeichnis in folgender Reihenfolge, wobei das erste Verzeichnis, welches eine skin.ini enthält verwendet wird:

1. `%WinstDir%\..\custom\winstskin`
2. `%WinstDir%\winstskin`

Mit dem Befehl `SetSkinDirectory` kann ein SkinDirectory auch im Script angegeben werden. Wird bei diesem Befehl ein leerer oder ungültiger Pafd angegeben, so wird der Defaultpfad verwendet.

Beispiel:

```
SetSkinDirectory "%ScriptPath%\testskin"
sleepseconds 1
SetSkinDirectory ""
```

10.1.1 Fenster Modus / Skin

- `SetSkinDirectory` <skindir> setzt das zu verwendende SkinDirectory und lädt den Skin Wird bei diesem Befehl ein leerer oder ungültiger Pafd angegeben, so wird der Defaultpfad verwendet.

Beispiel:

```
SetSkinDirectory "%ScriptPath%\testskin"
sleepseconds 1
SetSkinDirectory ""
```

10.2 Kommandos zur Steuerung des Logging

(. . .)

- `SetConfidential` <secret string>
  Dient dazu vertrauliche Informationen (z.B. Passwörter) aus den Logdateien fernzuhalten. Diese werden dort durch ***(confidential)*** ersetzt.
  Wir der Loglevel auf *9* gesetzt, so werden die 'confidential's im Klartext gelogt.
  Seit Version 4.11.3.5

Beispiel:

```
message "SetConfidential"
SetConfidential "forbidden"
comment "This is a forbidden string"
comment "should be in the log file: This is a ***(confidential)*** string"
```

Log:

```
message SetConfidential
comment: This is a ***(secret)*** string
comment: should be in the log file: This is a ***(confidential)*** string
```

10.3 Umgang mit Zahlen

Es gibt im *opsi-winst* keine speziellen Variablen für Zahlen. Es gibt allerdings einige Funktionen welche beim Umgang mit Zahlen helfen.

- `calculate(`<str>`)`
  Stringfunktion welche den arithmetischen Ausdruck im String <str> berechnet und als gerundeten integer String zurückgibt.
  Intern werden die Berechnungen mit reellen Zahlen durchgeführt. Die Funktion kennt derzeit die Operatoren `+`, `-`, `*`, `/` sowie Klammern `(,)`.
  Im Fehlerfall wird ein leerer String zurückgegeben und der Errorcounter um eins erhöht. Enthält der übergebene String Zeichen, die keine Zahlen oder gültige Operatoren sind, so ist dies ein Fehler.
  Fehlt der zweite Operand, so wird hierfür der erste verwendet: $5+ = 10$; $5* = 25$. Daher sollte beim Zusammensetzen des übergebenen strings verwendete Variablen z.B. mit der Funktion `isNumber` auf Gültigkeit geprüft werden.
  Seit 4.11.3.5

Beispiele:

```
set $ConstTest$ = "0"
set $CompValue$ = calculate("-1+1")
if ($ConstTest$ = $CompValue$)
        comment "passed"
else
        set $TestResult$ = "not o.k."
        LogWarning "failed"
endif
set $ConstTest$ = "1"
set $CompValue$ = calculate("0+1")
if ($ConstTest$ = $CompValue$)
        comment "passed"
else
```

```
        set $TestResult$ = "not o.k."
        LogWarning "failed"
endif
set $ConstTest$ = "-1"
set $CompValue$ = calculate("0-1")
if ($ConstTest$ = $CompValue$)
        comment "passed"
else
        set $TestResult$ = "not o.k."
        LogWarning "failed"
endif
set $string1$ = "5"
set $string2$ = "5"
set $ConstTest$ = "25"
set $CompValue$ = calculate($string1$+"*"+$string2$)
if ($ConstTest$ = $CompValue$)
        comment "passed"
else
        set $TestResult$ = "not o.k."
        LogWarning "failed"
endif
set $string1$ = "5"
set $string2$ = "5"
set $ConstTest$ = "1"
set $CompValue$ = calculate($string1$+"/"+$string2$)
if ($ConstTest$ = $CompValue$)
        comment "passed"
else
        set $TestResult$ = "not o.k."
        LogWarning "failed"
endif
set $string1$ = "5"
set $string2$ = "0"
set $ConstTest$ = ""
comment " expecting devision by zero error and empty string result"
set $CompValue$ = calculate($string1$+"/"+$string2$)
if ($ConstTest$ = $CompValue$)
        comment "passed"
else
        set $TestResult$ = "not o.k."
        LogWarning "failed"
endif
set $string1$ = "9"
set $string2$ = "10"
set $ConstTest$ = "1"
comment "result 0.9 is rounded to 1 "
set $CompValue$ = calculate($string1$+"/"+$string2$)
if ($ConstTest$ = $CompValue$)
        comment "passed"
else
        set $TestResult$ = "not o.k."
        LogWarning "failed"
endif
set $string1$ = "10"
set $string2$ = "9"
set $ConstTest$ = "1"
comment "result 1.1111 is rounded to 1 "
```

```
set $CompValue$ = calculate($string1$+"/"+$string2$)
if ($ConstTest$ = $CompValue$)
        comment "passed"
else
        set $TestResult$ = "not o.k."
        LogWarning "failed"
endif
set $string1$ = "5"
set $string2$ = "5"
set $ConstTest$ = "55"
comment " rule * before +"
set $CompValue$ = calculate($string1$+"+"+$string2$+"*10")
if ($ConstTest$ = $CompValue$)
        comment "passed"
else
        set $TestResult$ = "not o.k."
        LogWarning "failed"
endif
set $string1$ = "5"
set $string2$ = "5"
set $ConstTest$ = "100"
comment "brackets before  rule * before + "
set $CompValue$ = calculate("("+$string1$+"+"+$string2$+")*10")
if ($ConstTest$ = $CompValue$)
        comment "passed"
else
        set $TestResult$ = "not o.k."
        LogWarning "failed"
endif
set $string1$ = "5"
set $string2$ = "ten"
set $ConstTest$ = ""
comment "invalid char error"
set $CompValue$ = calculate($string1$+"*"+$string2$)
if ($ConstTest$ = $CompValue$)
        comment "passed"
else
        set $TestResult$ = "not o.k."
        LogWarning "failed"
endif
set $string1$ = "5"
set $string2$ = ""
set $ConstTest$ = "25"
comment "5* is interpreted as 5*5"
set $CompValue$ = calculate($string1$+"*")
if ($ConstTest$ = $CompValue$)
        comment "passed"
else
        set $TestResult$ = "not o.k."
        LogWarning "failed"
endif
set $string1$ = "5"
set $string2$ = ""
set $ConstTest$ = "10"
comment "5+ is interpreted as 5+5"
set $CompValue$ = calculate($string1$+"+")
if ($ConstTest$ = $CompValue$)
```

```
        comment "passed"
else
        set $TestResult$ = "not o.k."
        LogWarning "failed"
endif
set $string1$ = "nothing"
set $string2$ = "foo"
set $ConstTest$ = ""
comment "invalid char error"
set $CompValue$ = calculate($string1$+"*"+$string2$)
if ($ConstTest$ = $CompValue$)
        comment "passed"
else
        set $TestResult$ = "not o.k."
        LogWarning "failed"
endif
set $string1$ = "5"
set $string2$ = "foo"
set $ConstTest$ = ""
comment "invalid char error"
set $CompValue$ = calculate($string1$+"/"+$string2$)
if ($ConstTest$ = $CompValue$)
        comment "passed"
else
        set $TestResult$ = "not o.k."
        LogWarning "failed"
endif
```

Für weitere Beispiele beachten Sie das Produkt *opsi-winst-test* und dort den Bereich *$Flag_calculate$ = "on"*

10.4 PatchTextFile-Sektionen

10.4.1 Aufrufparameter

Der Name der zu patchenden Datei wird als Parameter übergeben.

Als optionalen Modifier gibt es:

- `/AllNTUserProfiles`
  Wird eine *PatchTextFile* Sektion mit diesem Modifier aufgerufen und der Pfad zur zu patchenden Datei enthält die Konstante `%UserProfileDir%`, so wird diese Patchsektion für alle Profile ausgeführt.
  Eine *PatchTextFile* Sektion welche in einer `[ProfileActions]` Sektion aufgerufen wird hat im *Machine* Modus den Modifier `/AllNTUserProfiles` implizit. Im Loginscript Modus wird dann `%UserProfileDir%` als `%CurrentProfileDir%` interpretiert.
  (Seit Version 4.11.3.5)

10.5 LinkFolder-Sektionen

(. . .)

In einer LinkFolder-Sektion muss zuerst bestimmt werden, in welchem virtuellen Systemfolder die nachfolgenden Anweisungen arbeiten sollen. Dafür existiert die Anweisung
`set_basefolder` *<virtueller Systemfolder>*

Virtuelle Systemfolder, die angesprochen werden können, sind:

desktop, sendto, startmenu, startup, programs, desktopdirectory, common_startmenu, common_programs, common_startup, common_desktopdirectory

Die Folder sind virtuell, weil erst durch das Betriebssystem(-Version) bestimmt wird, an welchem physikalischen Ort des Dateisystems sie real existieren.

Im Rahmen einer normalen *Maschinen* Installation sind nur die `common*` Systemfolder relevant.

Die Systemfolder *desktop, sendto, startmenu, startup, programs, desktopdirectory* können nur im Kontext eines eingloggten users bzw. in einem *userLoginScript* im Rahmen der opsi-Erweiterung *user Profile Management* verwendet werden.

10.6 ExecWith Sektionen

10.6.1 Aufrufparameter (Modifier)

(. . .)

Es sind folgende *opsi-winst*-Optionen verfügbar:

- `/32Bit`
  Das ist der Default. Der angegebene Interpreterpfad wird als 32 Bit Pfad interpretiert.
  Beispiel: `c:\windows\system32\WindowsPowerShell\v1.0\powershell.exe` ruft (auch auf einem 64bit System) die 32 Bit *powershell.exe* auf.
- `/64Bit`
  Der angegebene Interpreterpfad wird als 64 Bit Pfad interpretiert.
  Beispiel: `c:\windows\system32\WindowsPowerShell\v1.0\powershell.exe` ruft (auf einem 64bit System) die 64 Bit *powershell.exe* auf.
- `/SysNative`
  Der angegebene Interpreterpfad wird gemäß der OS Architektur interpretiert.
  Beispiel: `c:\windows\system32\WindowsPowerShell\v1.0\powershell.exe` ruft auf einem 64bit System die 64 Bit *powershell.exe* und auf einem 32bit System die 32 Bit 'powershell.exe'auf.

Seit Version 4.11.3.5 wird, wenn im angegebene Interpreterpfad *powershell.exe* vorkommt, die temporäre Datei als `.ps1` gespeichert.

10.7 WinBatch-Sektionen

10.7.1 Aufrufparameter (Modifier)

- `/32Bit`
  Das ist der Default. Die in der Sektion angegebene Pfade werden als 32 Bit Pfade interpretiert.
  Beispiel: `c:\windows\system32\regedit.exe` ruft (auch auf einem 64bit System) die 32 Bit *regedit.exe* auf.
- `/64Bit`
  Die in der Sektion angegebene Pfade werden als 64 Bit Pfade interpretiert.
  Beispiel: `c:\windows\system32\regedit.exe` ruft (auf einem 64bit System) die 64 Bit *regedit.exe* auf.
- `/SysNative`
  Die in der Sektion angegebene Pfade werden gemäß der OS Architektur interpretiert.
  Beispiel: `c:\windows\system32\regedit.exe` ruft auf einem 64bit System die 64 Bit *regedit.exe* und auf einem 32bit System die 32 Bit 'regedit.exe'auf.

Beispiel:

```
Winbatch_add_reg winst /64Bit
[Winbatch_add_reg]
"c:\windows\system32\regedit.exe" /s "%scriptpath%\my64.reg"
```

- `/RunAsLoggedonUser` //seit 4.11.3.5
  Nur im Kontext eines *userLoginScripts* verfügbar. Startet das Programm als der user der sich gerade einlogt. Dieser Modifier hat folgende Einschränkungen:

  – Unter NT6 wenig getestet und evtl. nur eingeschränkt wirksam.

10.8 LDAPsearch Sektion

10.8.1 LDAPsearch Sektion Syntax

Eine LDAPsearch Sektion beinhaltet die Spezifikationen:

(. . .)

- `user:`
  Zu verwendender user Name. Seit 4.11.3.5
- `password:`
  Zu verwendendes user Passwort. Seit 4.11.3.5

(. . .)

10.8.2 Beispiele

(. . .)

Beispiel mit user / password

```
comment ""
comment "-------------------------------------"
comment "Testing: "
comment "user / password"
Set $LdapHost$ = "vmix7.uib.local"
Set $LdapPort$ = "389"
Set $LdapUser$ = "cn=Administrator,cn=Users,dc=uib,dc=local"
Set $LdapPassword$ = "Linux123"
Set $LdapResultType$ = "objects"
Set $LdapSearchDn$ = "cn=Users,dc=uib,dc=local"
Set $LdapSearchAttributes$ = "name,objectClass"
Set $LdapFilter$ = "(&(objectclass=*))"

markErrorNumber
set $list1$ = getReturnListFromSection("ldapsearch_users /" + $LdapResultType$)
if errorsOccuredSinceMark > 0
        comment "failed while ldapsearch"
        set $TestResult$ = "not o.k."
else
        comment "passed"
endif
```

```
[ldapsearch_users]
targethost: $LdapHost$
targetport: $LdapPort$
user: $LdapUser$
password: $LdapPassword$
dn: $LdapSearchDn$
attributes: $LdapSearchAttributes$
filter: $LdapFilter$
```

Kapitel 11

opsi-configed 4.0.3.1

11.1 Host-Parameter in der Client- und der Serverkonfiguration

Zur besseren Übersicht werden die Host-Parameter jetzt gegliedert nach Funktionsgruppen aufgeführt. Die Gruppen sind auf der linken Seite in einer baumartigen Struktur dargestellt. Jeweils die zugehörigen Parameter und ihre Werte werden, wenn die Gruppe links aktiviert ist, auf der rechten Seite präsentiert.

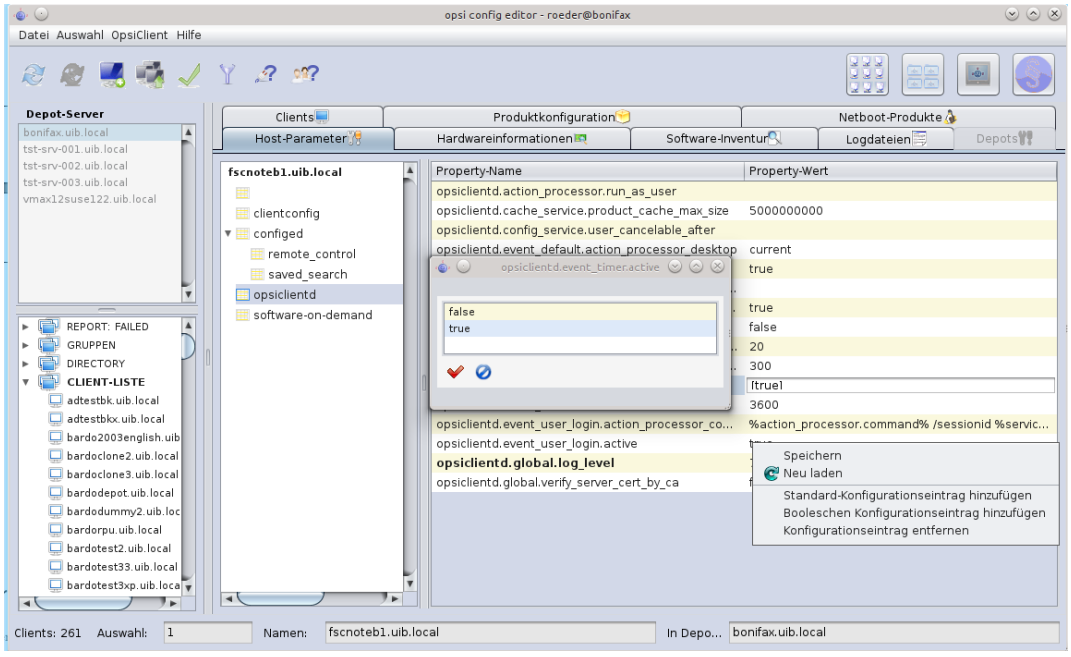

Abbildung 11.1: *opsi-configed*: Tab Hostparameter (Server- und Client-Konfiguration)

11.2 Logdateien

Der Level, bis zu dem die Logeinträge sichtbar sind, kann jetzt mit einem Schieberegler variiert werden, so dass Fehler leichter gefunden werden können. Der Regler ist auch mausrad-bedienbar.

Event-bezogene Logs können selektiert werden.

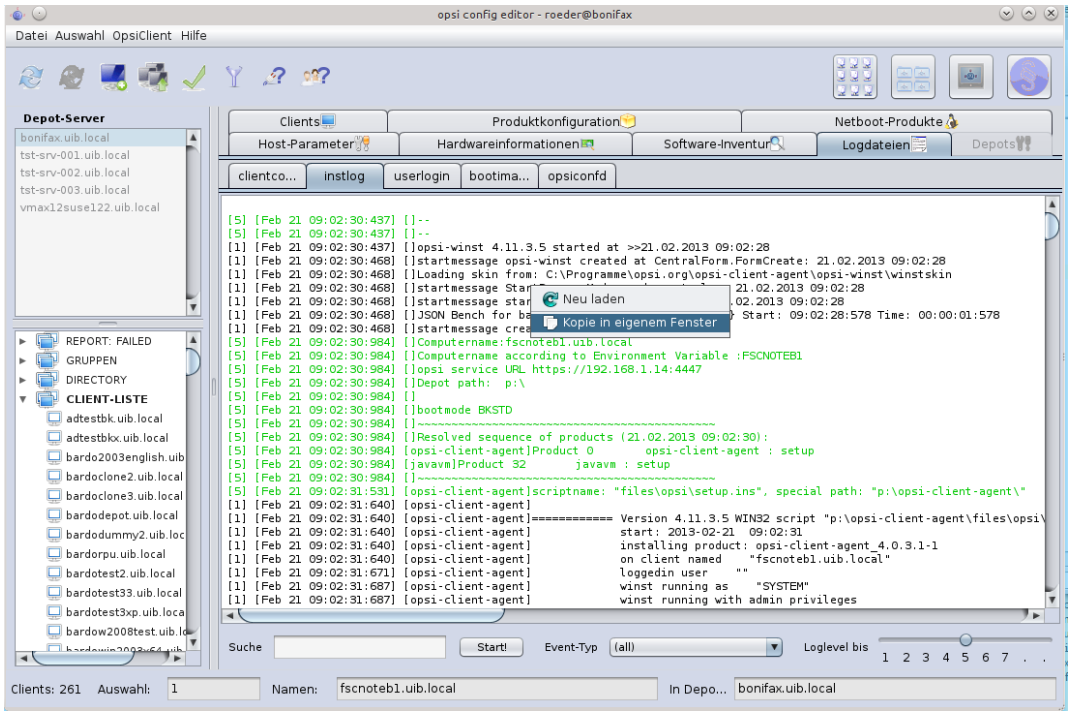

Abbildung 11.2: *opsi-configed*: Tab Logdateien

Kapitel 12

Sonstiges

12.1 Verweise auf das opsi-manual

Einige Kapitel dieser Release Notes sind Bestandteil des opsi Handbuchs *opsi-manual* und verweisen auf weitere Kapitel des opsi Handbuchs.

12.1.1 Freischaltung kostenpflichtiger Module

Weitere Informationen zur Freischaltung kostenpflichtiger Module finden Sie im *opsi-manual* im gleichnamigen Kapitel.

12.1.2 Konfiguration über den Webservice (Host-Parameter)

Weitere Informationen zur Konfiguration per Webservice finden Sie im *opsi-manual* im gleichnamigen Kapitel.

12.2 Changelogs:

12.2.1 Changelog opsi-winst

opsi-winst (4.11.3.6) stable; urgency=low

- fix: no [productId] while listing of products

— Detlef Oertel <d.oertel@uib.de> Thu, 11 Mar 2013:15:00:00 +0200

opsi-winst (4.11.3.5) stable; urgency=low

- del / delete will not delete if the given Filename starts with \ (eg. *Dummy*)
- danish localization added
- italien localization added
- extended LDAPSearch (username/password)
- fix: do not store isfatalerror in registry if nor running as admin
- execwith: winst (/32bit|/64bit|/sysnative) no allowed
- execwith: if *powershell.exe* is part of the interpreter string the script extension is *.ps1*

- winbatch: /32bit|/64bit|/sysnative no allowed
- wilog: replace ' -→' by ' =⇒' because the opsi web service get in trouble because it is interpreted as xml comment, fixes #350
- switch to Lazarus 1.0 / fpc 2.6.0
- wisynt: doAktionen, fileActionsMain: do ApplyTextConstants for filesections in fileActionsMain. fixes wrong resolution of %CurrentProfileDir% in Files /AllNTuserProfiles
- wisynt: doLinkFolderActions: fix: now comments allowed between set_link and end_link
- wisynt: doLinkFolderActions: linkfolder section now usable in userLoginScripts: the predefined basfolders: *desktop*, *sendto*, *startmenu*, *programs*, *startup*, *desktopdirectory* are pointing to the loggedin user directories if called from userLoginScript
- wimain: ProcessNonZeroScript: fix: do not run update after setup if setup is failed, fixes #432
- fix: widatamodul: opsidata4.initproduct now boolean function: false = error
- new: new winst cli parameter '/productlist <productid>[,<productid>]* as sub parameter to /opsiservice try to run the setup-scripts of the given products, change installation states, send log files
- new: string function calculate(<string expr>) ; calculates the mathematical expression and returns the result as string. on error the result is a empty string and the errorcounter is increased
- new command *setConfidential <confidential string>* ; adds the given string to a list of strings which are replaced with ***(confidential)*** while logging
- logging: added [<productid>] to log line to use the new configed filter possibilities

— Detlef Oertel <d.oertel@uib.de> Thu, 05 Dec 2012:15:00:00 +0200

opsi-winst (4.11.3.4) stable; urgency=low

- new winbatch parameter /RunAsLoggedOnUser
- PatchTextFile sections may now called with %userprofiledir% as part of the file name and /AllNTuserProfiles so it will run on all profiles. If running in Machine mode, the */AllNTuserProfiles* is implizit in the [ProfileActions]. If running in LoginScript mode, the *%userprofiledir%* is the %CurrentProfileDir%.
- will now look for skins (more exactly: skin.ini) in the following sequence: 1) by parameter 2) new custom directory of opsi-client-agent ../custom/winstskin 3) own winstskin: winstskin

— Detlef Oertel <d.oertel@uib.de> Thu, 26 Oct 2012:15:00:00 +0200

12.2.2 Changelog opsi-winst-test

opsi-winst-test (4.11.3.5-1) stable; urgency=low

- Test of Linkfolder in login.opsiscript
- Test of use of temporary loop constant in subsub call
- Test of calculate
- Test of *no update script if setup is failed*: new property: setfailed
- Test of SetConfidential
- Test of SetSkinDirectory
- Test of LDAPSearch (username/password)

- Test of execwith: winst (/32bit|/64bit|/sysnative)
- Test of winbatch: /32bit|/64bit|/sysnative
- Test of PatchTextFile /AllNTuserProfiles

— detlef oertel <d.oertel@uib.de> Wed, 16 Jan 2013 17:00:13 +0200

opsi-winst-test (4.11.3.4-1) stable; urgency=low

- Test of Winbatch /RunAsLoggeedOnUser in login.ins

— detlef oertel <d.oertel@uib.de> Thu, 26 Oct 2012 17:00:13 +0200

12.2.3 Changelog jedit

jedit_5.0.0-3 stable; urgency=low

- opsi-winst.xml for version 4.11.3.5
- code cleanup

— detlef oertel <d.oertel@uib.de> Fri, 15 Feb 2013 16:01:53 +0200

jedit_5.0.0-2 stable; urgency=low

- jedit version with localization support

— rupert röder <r.roeder@uib.de> Tue, 27 Nov 2012 16:01:53 +0200

12.2.4 Changelog opsi-adminutils

opsi-adminutils (4.0.3.-1) stable; urgency=low

- change to 4.0.3

— detlef oertel <d.oertel@uib.de> 15 Feb 2013

opsi-adminutils (4.0.2.1-5) stable; urgency=low

- correct parameters for configed.jnlp URL

— rupert roeder <r.roeder@uib.de> 10 Jan 2013

opsi-adminutils (4.0.2.1-4) stable; urgency=low

- removed parameter from configed.jnlp URL
- removed opsi-configed version from control file

— detlef oertel <d.oertel@uib.de> 04 Jan 2013

opsi-adminutils (4.0.2.1-3) stable; urgency=low

- regshot 1.8.3 (include 64-bit)
- added kitty

— detlef oertel <d.oertel@uib.de> 11 Oct 2012

opsi-adminutils (4.0.2.1-2) stable; urgency=low * swingx.jar * configed.jar 4.0.2.5.4 — bardo wolf <b.wolf@uib.de> 04 Sep 2012

opsi-adminutils (4.0.2.1-1) stable; urgency=low

12.2.5 Changelog opsi-template

opsi-template (4.0.3-1) stable; urgency=low

- default is now without property and 32 Bit

— detlef oertel <d.oertel@uib.de> Thu, 27 Sep 2012 16:01:53 +0200

12.2.6 Changelog windows netboot products

windows (4.0.3-2) stable; urgency=low

- fix in logging in real uefi mode

```
-- uib GmbH <info@uib.de>  Thu, 11 Mar 2013 15:19:15 +0000
```

windows (4.0.3-1) stable; urgency=low

- do not copy winpe uefi boot files if not in uefi mode
- code cleanup
- new property winpe_partition_size
- showdrivers.py:
  - Option -V for version output added.
  - byAudit: Translating model and vendor from hwinvent: characters <>?":|\/* will be translated to _

— uib GmbH <info@uib.de> Thu, 14 Feb 2013 15:19:15 +0000

windows (4.0.2-2) experimental; urgency=low

- new property data_partition_create
- detect if we are on uefi
- if we are on uefi use gpt partitions
- win8 (NT 6.2) support
- new property use_raid1

— uib GmbH <info@uib.de> Mon, 27 Aug 2012 15:19:15 +0000

12.2.7 Changelog opsi-client-agent

opsi-client-agent (4.0.3.1-2) stable; urgency=low

- opsi-winst 4.11.3.6

— Detlef Oertel <d.oertel@uib.de> Tue, 12 Mar 2013 15:00:00 +0100

opsi-client-agent (4.0.2.3-1) testing; urgency=low

- opsiclientd.conf:
  - added new silent_install event

  - added new time event for silent_install
- fix search algorithm for finding usersessions
- fixed setting userrights for local opsi-winst
- added new service-method for cleaning cache (WAN-Support)

```
-- Erol Ueluekmen <e.ueluekmen@uib.de>  Wed, 14 Nov 2012 09:00:00 +0100
```

opsi-client-agent (4.0.2.3-2) testing; urgency=low

- opsiclientd.conf:
  -
- opsiclientd:
  - v4.0.1.75
  - new state installation_pending for future use (event_on_shutdown)
- new product property: on_shutdown_active (true/false)
- new updatescript, which installs shutdown files if on_shutdown_active=true

```
-- Miriam Michaelis <m.michaelis@uib.de>  Wed, 05 Dec 2012 13:00:00 +0100
```

opsi-client-agent (4.0.2.3-1) testing; urgency=low

- opsiclientd:
  - fix for user_cancelable = 0
  - fix for getSessioninformation
  - updated sessionhelper in utilities
  - added /usercontext parameter for user-login events. This option gives now opsi-winst the username.
  - fix for User-Login-Events on Windows 8: Do not fire event, if the Window Management Service is logged on.
- new custom directory
  - is saved in preinst and restored in postinst
  - cfg/config.ini values will be overwritten by values from custom/config.ini (execption: pckey , bootmode) fixes #333
  - it is a good idea to delete all not used key from a custom/config.ini
  - files at the depot in custom\winstskin*.* will be copied to custom\winstskin\ at the client and will be used by opsi-winst since version
  - files at the depot in custom\notifier*.* will be overwrite the files at the clients notifier directory
  - a existing file at the depot: custom\opsiclientd.conf will overwrite the opsiconfd.conf files at the clients opsiclientd directory, which comes from the *dist* directory

```
-- Erol Ueluekmen <e.ueluekmen@uib.de>  Wed, 14 Nov 2012 09:00:00 +0100
```

12.2.8 Changelog python-opsi

python-opsi (4.0.3.1-1) testing; urgency=low

- dhcp-backend: ddns-rev-domainname added to list where the values are written in double quotas
- System: opsi-setup --init-current-config gives an warning instead of error, when vendor not found for network device
- Posix:
  - saveImage returns the result from partclone if run was successfull.
  - readPartitionTable: Try to find out the right filesystem with blkid tool.
  - createPartition: allows linux as filesystem-type and produces partition with id 83
- WindowsDriver: byAudit: Translating model and vendor from hwinvent: characters <>?":|\/* will be translated to _
- python-opsi locale: danish added
- compareVersion: fixed handling with versions from custom packages.
- global.conf: fixed hostname entries
- fixed resource directory listing for custom packages /repository
- fix for ubuntu 12.10

```
-- Erol Ueluekmen <e.ueluekmen@uib.de>  Tue, 05 Feb 2013 17:40:23 +0100
```

python-opsi (4.0.2.6-1) testing; urgency=low

- Posix: getBlockDeviceControllerInfo():
  - if no devices attached on a AHCI-Controller (maybe a lshw or a kernel bug) try to find AHCI-Controller, if found try return the first found AHCI Controller for textmode-driverintegration (only for nt5)
- Posix: modifications for newer ms-sys version
- rpm-spec-file: noreplace option for dispatch.conf.default in files-section

```
-- Erol Ueluekmen <e.ueluekmen@uib.de>  Mon, 07 Nov 2012 17:34:13 +0100
```

python-opsi (4.0.2.5-1) testing; urgency=low

- fix in hwinvent procedure, don't crash if lshw don't work properly
- fix for resizeNTFSPartition if blockAlignmnet is used (ntfs-restore-image)

```
-- Erol Ueluekmen <e.ueluekmen@uib.de>  Fri, 02 Nov 2012 15:00:34 +0200
```

12.2.9 Changelog opsipxeconfd

opsipxeconfd (4.0.3.1-1) stable; urgency=low

- featurepack-release 4.0.3
- this version is stable, because no changes were made, only version modified for new featurepack-release

```
-- Erol Ueluekmen <e.ueluekmen@uib.de>  Thu, 14 Feb 2013 17:57:02 +0100
```

12.2.10 Changelog opsiconfd

opsiconfd (4.0.3.1-1) testing; urgency=low

- ressource: webstart jnlp build fixed for arguments.
- added monitoring debug switch in opsiconfd.conf
- fixed init-Script for using global.conf

```
-- Erol Ueluekmen <e.ueluekmen@uib.de>  Tue, 05 Jan 2013 13:15:56 +0100
```

12.2.11 Changelog opsi-utils

opsi-utils (4.0.3.1-1) testing; urgency=low

- opsi-product-updater: Added Blank line between header and message-body to fix empty body mails on exchange environment.
- opsi-product-updater: fix for handling with custom products.
- opsi-admin: added new HostControlSafeBackend in BackendManager initialization.

```
-- Erol Ueluekmen <e.ueluekmen@uib.de>  Tue, 12 Feb 2013 12:42:44 +0200
```

12.2.12 Changelog opsi-linux-bootimage

opsi-linux-bootimage (20130207-1) testing; urgency=low

- added bc

```
-- Erol Ueluekmen <e.ueluekmen@uib.de>  Fri, 01 Feb 2013 12:34:59 +0100
```

opsi-linux-bootimage (20130117-1) experimental; urgency=low

- downgrade: ms-sys: 2.1.2 kernel 3.6.11

```
-- Erol Ueluekmen <e.ueluekmen@uib.de>  Wed, 17 Jan 2013 16:25:11 +0100
```

opsi-linux-bootimage (20130111-1) experimental; urgency=low

- python-opsi 4.0.2.7-1
- locale: danish and italian localisation added
- kernel 3.7.1

```
-- Erol Ueluekmen <e.ueluekmen@uib.de>  Fri, 11 Jan 2013 11:48:23 +0100
```

opsi-linux-bootimage (20121107-1) experimental; urgency=low

- new ms-sys-version: 2.2.1

```
-- Erol Ueluekmen <e.ueluekmen@uib.de>  Wed, 07 Nov 2012 17:51:31 +0100
```

opsi-linux-bootimage (20121105-1) experimental; urgency=low

- python-opsi 4.0.2.6-1

```
-- Erol Ueluekmen <e.ueluekmen@uib.de>  Mon, 05 Nov 2012 17:36:27 +0100
```

opsi-linux-bootimage (20121102-1) experimental; urgency=low

- kernel 3.6.5
- python-opsi 4.0.2.5-1

```
-- Erol Ueluekmen <e.ueluekmen@uib.de>  Fri, 02 Nov 2012 15:11:43 +0100
```

opsi-linux-bootimage (20121024-1) unstable; urgency=low

- kernel 3.6.3
- python-opsi 4.0.2.5-1
- added packages: cloop, cloop-utils, dmraid, rsync, partclone-utils, nbd-client, genisoimage

```
-- Erol Ueluekmen <e.ueluekmen@uib.de>  Wed, 24 Oct 2012 12:57:00 +0200
```

opsi-linux-bootimage (20121011-1) experimental; urgency=low

- kernel 3.6.1
- python-opsi 4.0.2.4-1
- Missing firmware from Broadcom-Chipsets added

```
-- Erol Ueluekmen <e.ueluekmen@uib.de>  Thu, 11 Oct 2012 00:45:26 +0200
```

12.2.13 Changelog opsi-depotserver

opsi-depotserver (4.0.3.1-1) testing; urgency=low

- opsi-setup: added renew-opsiconfd-cert task
- opsi-setup: added special files in set rights to add execute bit for these files in depot

```
-- Erol Ueluekmen <e.ueluekmen@uib.de>  Mon, 28 Jan 2013 17:34:42 +0100
```

12.2.14 Changelog opsi4ucs

opsi4ucs (4.0.3.1-1) testing; urgency=low

- join script: execute /usr/lib/univention-pam/ldap-group-to-file.py to resync the groups (for ucs 3.1 support)
- opsi-setup: set executebit for special opsi files in depot

```
-- Erol Ueluekmen <e.ueluekmen@uib.de>  Tue, 05 Feb 2013 11:07:47 +0100
```

12.2.15 Changelog opsi-configed

opsi-configed (4.0.3.2-1) testing; urgency=low

- checking names for remote controls
- improvement of restart after language change
- improvement of swaudit page

```
-- Rupert Roeder <r.roeder@uib.de>  Fri, 13 Mar 2013 12:00:00 +0200
```

opsi-configed (4.0.3.1-1) testing; urgency=low

- host config editing grouped by a tree
- last change column for products now possible
- the configed log files are now placed in the user home directory in a folder .configed
- log panes now allow selection of displayed levels
- log panes allow selection of event type if a log file has lines with event type specification
- improved search handling in log panes
- log panes, software and hardware audit panes can be copied into separated frames

```
-- Rupert Roeder <r.roeder@uib.de>  Fri, 25 Jan 2013 12:00:00 +0200
```

- search function for windows software panel
- reset client selection when returning from server or depot editing
- bugfix in mac address editing
- configed checks if read only version is requested and, if wanted, produces it

```
-- Rupert Roeder <r.roeder@uib.de>  Mon, 03 Dec 2012 12:00:00 +0200
```

opsi-configed (4.0.2.10-1) testing; urgency=low

- new logging format, included time check for service calls

```
-- Rupert Roeder <r.roeder@uib.de>  Fri, 02 Nov 2012 12:00:00 +0200

opsi-configed (4.0.2.8-1) testing;  urgency=low
```

- check of transgression of activated client numbers
- read only option

```
-- Rupert Roeder <r.roeder@uib.de>  Mon, 22 Oct 2012 12:00:00 +0200
```

opsi-configed (4.0.2.7-1) testing; urgency=low

- reopening a saved search

```
-- Rupert Roeder <r.roeder@uib.de>  Mon, 15 Oct 2012 12:00:00 +0200
```

opsi-configed (4.0.2.7-1) testing; urgency=low

- more caching and less reloading actions

```
-- Rupert Roeder <r.roeder@uib.de>  Mon, 15 Oct 2012 12:00:00 +0200
```