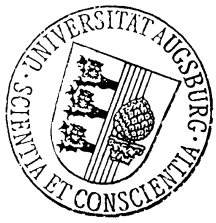


Universität Augsburg

Rechenzentrum

Fachbereich Informatik
(Lehrstuhl für Softwaretechnik und Programmiersprachen)

S E M I N A R A R B E I T

Desktop Management

Systeminstalltion und Softwareverteilung

vorgelegt von:	Florian Mücke	Adalbert Ochotta
Ort:	Königsbrunn	Königsbrunn
geboren am:	28. 12. 1980	27. 04. 1980
Matr.-Nr.:	739 469	736 526

Wintersemester 2005/2006

Abstract

Stationäre Computer und mobile Endgeräte wie Personal Digital Assistants oder Laptops sind aus dem heutigen Geschäftsalltag nicht mehr wegzudenken. Umgestaltungen der Softwarelandschaft, sowie die sicherheitstechnische Pflege der Geräte bedingen Softwarelösungen, welche die IT-Abteilung eines Unternehmens bei der Programmverwaltung unterstützen. Die hohe Anzahl an Endgeräten bedingt eine automatisierbare Möglichkeit zur Systemvorbereitung, sowie der Installation von Betriebssystem und weiteren Anwendungen. Dabei stellt gerade die Heterogenität der Hardwareumgebung und die Erhaltung benutzerspezifischer Einstellungen eine nicht unerhebliche Herausforderung dar. Die vorliegende Arbeit soll aus dem Uneins der vorhandenen Informationsquellen und den verschiedenen verfügbaren Produkten und Lösungen der Softwareverteilung, ihren Prozess umfassend darstellen und Gemeinsamkeiten aufzeigen. Es werden gängige Verfahren erläutert und eine Übersicht über einige, am Markt befindlichen Produkte geben.

Inhaltsverzeichnis

1 Einleitung

Stationäre Computer und mobile Endgeräte wie Personal Digital Assistants oder Laptops sind aus dem heutigen Geschäftsalltag nicht mehr wegzudenken. Umgestaltungen der Softwarelandschaft, sowie die sicherheitstechnische Pflege der Geräte bedingen Softwarelösungen, welche die IT-Abteilung eines Unternehmens bei der Programmverwaltung unterstützen. Die hohe Anzahl an Endgeräten bedingt eine automatisierbare Möglichkeit zur Systemvorbereitung, sowie der Installation von Betriebssystem und weiteren Anwendungen. Dabei stellt gerade die Heterogenität der Hardwareumgebung und die Erhaltung benutzerspezifischer Einstellungen eine nicht unerhebliche Herausforderung dar.

Die automatisierte Verteilung und Wartung von Software gewinnt deshalb stetig an Bedeutung. Da sich, wie in [1] angegeben, die Innovationszyklen für System- und Anwendungssoftware verkürzen, vergrößert sich das Volumen an Software-Komponenten. Das Resultat sind immer häufigere Installationen und Programmupdates. Die Marktstudie *Client Management 2002* [2] belegt, dass allein 2002 bei den befragten Unternehmen, pro System durchschnittlich neun Installationen in diesem Jahr durchgeführt wurden. Von Hand ist dieser Aufwand nur noch teuer zu bewältigen. Durch automatisierte Softwarelösungen kann hier bares Geld gespart werden.

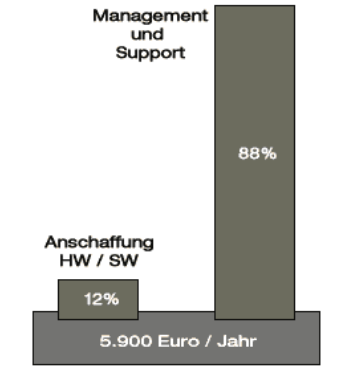


Abb. 1: *Gesamtkosten pro PC*

Softwaredeploymentsysteme bieten dazu verschiedene Lösungen, um die Erfordernisse eines Unternehmens abzudecken. Schwerpunkte sind die Inventarisierung, die Verteilung und Installation von Betriebssystemen, Anwendungen und Systemeinstellungen.

Durch den Einsatz eines Softwareverteilungstools sollen vornehmlich folgende Ziele erreicht werden:

- verringerter Administrationsaufwand
- einheitliche Installationen
- schnelle Updates der Systeme
- schnelles System-Recovery
- Installationsmöglichkeit außerhalb der Arbeitszeit
- Möglichkeit für umfassende Rollouts
- zentrale Betreuung dezentraler Infrastrukturen
- Kosteneinsparung

Die vorliegende Arbeit soll aus dem Uneins der vorhandenen Informationsquellen und den verschiedenen verfügbaren Produkten und Lösungen zur Softwareverteilung, den Prozess der Softwareverteilung umfassend darstellen und Gemeinsamkeiten aufzeigen. Auch werden gängige Verfahren erläutert und eine Übersicht über einige, am Markt befindlichen Produkte gegeben und auf das IT-Umfeld der Universität Augsburg eingegangen.

2 Schematischer Aufbau des Verteilungsprozesses

„Application (Software) development is a complex process which covers all the activities that have to be carried out from the end of the development itself on producer sites to the actual installation and maintenance of the application on consumer computers“ ([3]).

Es ist daher sinnvoll den Ablauf der Softwareverteilung so umfassend wie möglich zu erfassen. Als Beispiele für Aktionen dieses Ablaufs, sind an dieser Stelle die Paketierung der Software beim Hersteller, ihre Übertragung vom Hersteller zum Benutzer und die Installation, Aktualisierung und eventuelle De-Installation beim Benutzer zu nennen.

Aufgrund der Schwierigkeit der zugrunde liegenden vielschichtigen Vorgänge ist speziell die Softwareverteilung im größeren Umfeld in der Regel als kritisches Thema einzustufen. Dies liegt zum einen an den immer komplexer werdenden Anwendungen selbst und zum anderen am immer komplexer werdenden Umfeld. Es gibt ständig neue und unterschiedliche Versionen von Hardwarekomponenten, Betriebssystemen und Anwendungen die berücksichtigt werden müssen um einen reibungslosen Ablauf und ein ungehindertes Zusammenspiel zu gewährleisten. Auch der zeitliche Ablauf spielt bei der Softwareverteilung eine entscheidende Rolle. Laut [3] müssen daher folgende Fragen in größeren Unternehmen unbedingt beachtet werden: Soll die Verteilung auf allen Endgeräten gleichzeitig erfolgen („big bang“, [3]) oder lieber Schritt für Schritt nach festgelegten Bereichen? Ist die Konsistenz der Systeme gesichert und können Wechselwirkungen mit anderen Anwendungen auftreten? Wird die Produktivität des Unternehmens dadurch beeinträchtigt?

Es ist demnach in erster Linie wichtig den finanziellen und zeitlichen Aufwand und das damit verbundene Risiko so klein wie möglich zu halten und dennoch so flexibel wie möglich zu sein.

In [3] beschreiben die Autoren Coupaye und Estublier ein Modell mit drei Schichten als Grundlage für den Softwareverteilungsprozess. Dieses Modell ist auch in diesem Kapitel das Thema, da es den Sachverhalt deutlich und umfassend darlegt. Die Schichten sind die *Producer* (Hersteller-), die *Enterprise* (Unternehmens-) und die *User* (Benutzer-) Schicht.

Eine eigene *Enterprise-Schicht* ist in vielen Fällen nicht durch bestehende Technologien abgedeckt, da meistens nur der Verteilungsprozess zwischen einem Softwarelieferanten (producer) und einem Softwareabnehmer (user) und nicht der globale Prozess betrachtet wird. Große Unternehmen benötigen aber die Kontrolle über die Verteilung der Software. Die Software muss teilweise erst auf das Unternehmen abgestimmt, angepasst bzw. Erweiterungen eingebunden werden. Das Ganze muss abschließend noch getestet werden, bevor es dann in gesicherten Bahnen verteilt werden kann. Diese Faktoren können in einem

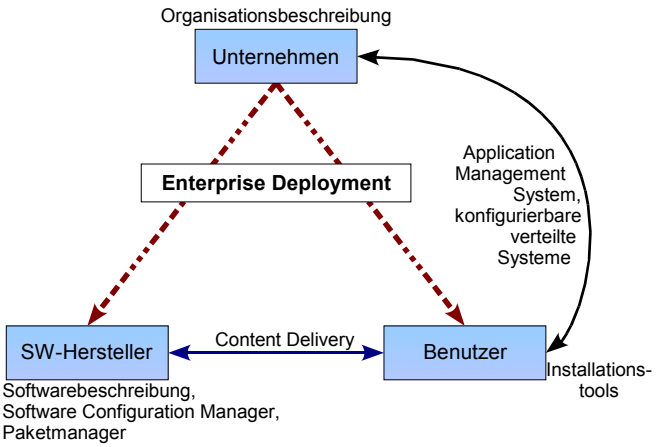


Abb. 2: *Enterprise Software Deployment (nach [3] Fig. 1)*

Zwei-Schichten-Modell nicht ausreichend dargelegt werden. Die Autoren betrachten die Anwendungsverteilung als einen Vorgang, dessen Aktionen einzelne Handlungsschritte verkörpern. Modelle und Richtlinien, die als Produkte bezeichnet werden, stellen eine Art Datenfluss zwischen Aktionen und Ressourcen (Personen, Hardware, Software) dar. Das Ganze ergibt ein, in sich geschlossenes System, da der Vorgang nicht ohne Aktionen, Produkte und Ressourcen ablaufen kann.

Die dazu erforderlich Abläufe werden in den nächsten drei Unterkapiteln beschrieben. Die zugrunde liegenden Modelle und Richtlinien sind im Anschluss daran ausführlich dargelegt.

2.1 Verteilung auf Herstellerseite (Producer Deployment)

Das Ziel des Herstellers einer Anwendung ist, diese geschlossen an den Kunden zu bringen. Dazu muss er sie nach ihrer Fertigstellung erst einmal zu einer Einheit zusammenschnüren (paketieren), um sie dann ausliefern zu können. Der Hersteller muss dazu die neue Anwendung dem Kundenkreis erst noch bekannt machen (advertise). Es muss aber auch dafür Sorge getragen werden, dass der Kunde darüber informiert wird, wenn alte Versionen abgekündigt werden und nicht mehr unterstützt bzw. durch neue Versionen ersetzt werden (unrelease & retire, [3]).

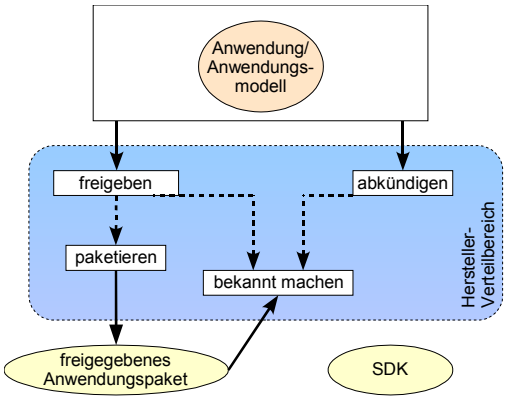


Abb. 3: *Verteilungsprozess auf Herstellerseite (nach [3] Fig.2)*

Das Bekanntmachen der neuen Anwendung (Advertising) kann beispielsweise per E-Mail, Newsletter oder direkt durch installierte Anwendungen/Dienste erfolgen. Die Vorgänge zur Bereitstellung der Anwendung werden meist durch Software Configuration Manager (Näheres siehe [4]) oder durch Paketmanager und automatisierte Installationslösungen (RPM [5], .deb/dpkg [6], MSI [7], InstallFromTheWeb [8], usw.) realisiert, die spezielle Programme für das Paketieren anbieten.

2.2 Verteilung auf Unternehmensseite (Enterprise Deployment)

Im Folgenden werden die Aktionen, die in einem Unternehmen beim Verteilungsprozess notwendig sind, beschrieben. Sie dienen der Vorbereitung zur tatsächlichen Verteilung der Anwendung zum Endbenutzer. Da dieser Bereich die organisatorischen Entscheidungen bezüglich des Verteilungsprozesses im ganzen Unternehmen widerspiegelt, ist er entscheidend für die Unternehmensseite [3].

Der erste Schritt nach dem Herausbringen einer neuen Anwendung ist die Übertragung zum Unternehmen (siehe **Abb. 2**). Diese Aktion kann vom Hersteller oder vom Unternehmen selbst ausgehen und wird abhängig davon dann unterschiedlich realisiert. Genauer wird auf die verwendeten Verfahren (Push und Pull) in Kapitel 3 eingegangen.

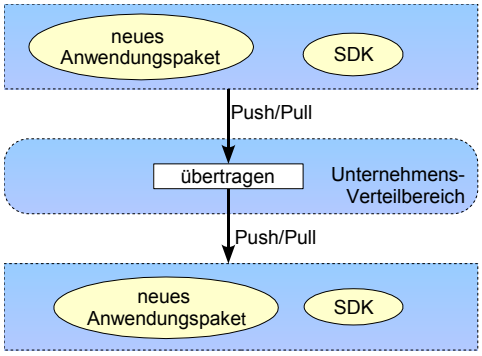


Abb. 4: *Transferaktion vom Hersteller zum Endkunden (nach [3] Fig.3)*

Hat das Unternehmen die Anwendung erhalten, muss sie diese als Erstes an die unternehmenseigene Umgebung anpassen. Dazu kann es notwendig sein, spezielle Erweiterungen einzubinden, Konfigurationen oder gar andere Anwendungen zu verändern, um die gegenseitige Verträglichkeit sicherzustellen – je nach Komplexität, Größe und Komplikationsgrad der neuen Anwendung. Daraufhin muss das Unternehmen die modifizierte Anwendung testen und anschließend wieder ein verteilbares Paket erzeugen, bevor die Übertragung zum Endbenutzer erfolgt. Dazu stehen dem Unternehmen in der Regel ähnliche Werkzeuge wie dem Hersteller zur Verfügung. Die Anpassung ist laut [3] einer der zeit- und aufwandsintensivsten Vorgänge des gesamten Ablaufs, da sie größtenteils nicht automatisiert werden können.

Der Zeitpunkt und die Art der Anwendungsverteilung werden im *predispose*-Vorgang festgelegt. Wie in **Abb. 3** dargestellt fließen generelle Richtlinien des Verteilungsprozesses und das allgemeine Unternehmensmodell in diesen Vorgang mit ein. In diesem Schritt werden die Organisationsstrukturen des Unternehmens durch das *Unternehmensmodell* auf den Verteilprozess abgebildet. Das *Predispose* ist also das direkte Bindeglied zwischen dem Unternehmen und dem Verteilungsprozess. In den *Verteilungsrichtlinien* ist festgelegt, welche Teilgruppen oder Anwendungspakete bevorzugt werden (Prioritäten), welche Abhängigkeiten erfüllt sein müssen und wie die Verteilung zeitlich ablaufen soll. Aus diesem *predispose*-Vorgang ergibt sich das erzeugte *Verteilungsmodell*, das, abhängig von Funktion und Tätigkeit bzgl. der Organisationsstrukturen die Verteilung von Version und Art der Anwendung für jedes Zielsystem/Benutzer regelt. Das Verteilungsmodell wird in [3] als Szenario gesehen, welches durch die Infrastruktur des Verteilsystems ausgewertet wird, um selbstständig ablaufende Verteilungsaktionen auf den Zielsystemen durchzuführen. Um den eigentlichen Verteilprozess starten zu können muss jetzt noch dem Zielsystem (Benutzer/Softwareagent, etc.) bekannt gemacht werden, dass eine neues Anwendungspaket bereitliegt.

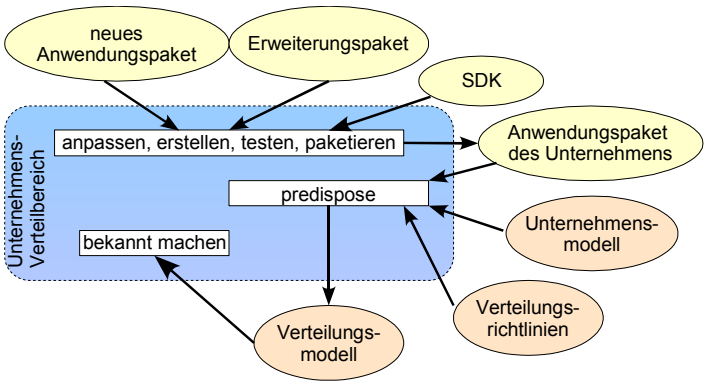


Abb. 5: *Unternehmensseitiger Verteilprozess (nach [3] Fig.4a)*

Die Autoren des Quellentextes [3] weisen auf Seite 3 darauf hin, dass die Aktivitäten des *predispose*-Vorgangs kaum von bestehenden Technologien abgedeckt werden, obwohl er ein Kernelement des *Enterprise Software Deployments* ist. Dem kann ich zustimmen, da sich seit Erscheinen des zugrunde liegenden Artikels in den vergangenen fünf Jahren, in dieser Beziehung kaum etwas geändert hat. Das liegt daran, dass das *Predispose* einer der umfassendsten und komplexesten Vorgänge der gesamten Softwareverteilung ist und keine einheitlichen Standards existieren, die die verschiedenen Modelle und Richtlinien vereinen.

2.3 Verteilung auf Benutzerseite (User Deployment)

Die Vorgänge, die im Bereich des Unternehmens ablaufen, liefern sowohl ein fertig verteilbares Anwendungspaket als auch ein Verteilungsmodell, das angibt, wann und wohin die Anwendung verteilt werden soll. Ihre Übertragung auf das System des Benutzers kann wie auch im vorherigen Kapitel (2.2 Unternehmensseitige Verteilung) über verschiedene Verfahren (Push und Pull) realisiert werden. Diese werden genauer in Kapitel 3.1 erläutert.

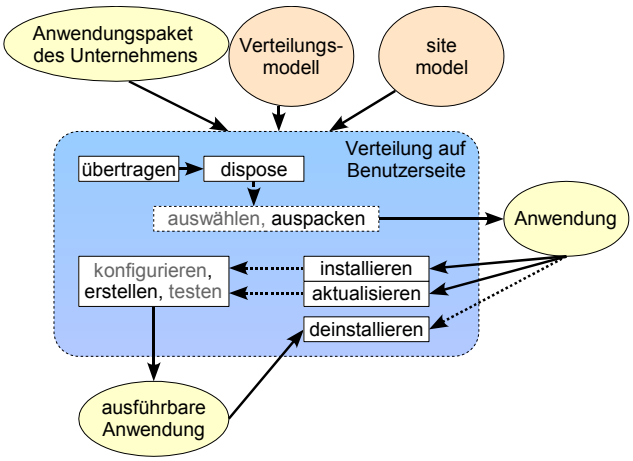


Abb. 6: *Tätigkeiten des Erstellungsprozesses auf der Benutzerseite (nach [3] Fig. 4b)*

Ist die Anwendung auf das System des Benutzers übertragen worden, übernimmt die *dispose* Aktivität die Vorbereitung für die nachfolgenden Prozesse – ähnlich dem *predispose* im Unternehmensmodell. Hier fließen das Anwendungspaket selbst, das Verteilungsmodell und das *„site model“*, welches die Hard- und Softwarekonfiguration des Benutzersystems spezifiziert, in die Vorbereitung ein. Der *dispose*-Vorgang besteht im Wesentlichen aus den zwei Teilaktivitäten *auswählen* und *auspacken*. Beim *Auswählen* werden nach dem *site model* die passenden Optionen für die Anwendung bestimmt (verfügbarer Hauptspeicher, Festplattenkapazität usw. – siehe [3]). Beim *Auspacken* wird lediglich das Anwendungspaket entpackt. Im nächsten Schritt kann die bereitgestellte und vorkonfigurierte Anwendung entweder installiert, aktualisiert werden. Es besteht auch die Möglichkeit alte Version zu deinstallieren. Die *Aktualisierung* bringt eine alte Version der Anwendung auf einen neuen, vom Hersteller oder Unternehmen herausgegebenen, Versionsstand. Dies können sowohl simple Programmupdates als auch komplett neue Versionen der Anwendungssoftware sein. Bei der Installation oder Aktualisierung wird die Anwendung dann in das Benutzersystem integriert. Dazu muss sie dem System angepasst (Programmpfade, Berechtigungen, Programmbibliotheken, etc.) und gegebenenfalls für das System neu erstellt (bei Linuxsystemen durchaus normal) und abschließend statische Tests durchgeführt werden. Oft ist es auch notwendig, dass System neu zu starten um den Installationsvorgang erfolgreich abschließen zu können. Dies ist im Hinblick auf die Überwachung des Ablaufs der Verteilung und der Installation im Speziellen wichtig (siehe Kapitel 4.1 – Reporting). Sind diese Schritte erfolgreich beendet, ist auf dem System eine ausführbare Anwendung und der Verteilungsprozess für dieses System damit abgeschlossen.

Der Verteilprozess auf Seite des Benutzers (bzw. die technische Seite des Verteilprozesses) ist nach Meinung der Autoren von [3] der Bereich, den von gängigen Programmen/Technologien am besten abgedeckt wird. Hier hat sich in den letzten fünf Jahren zu meinem Bedauern kaum etwas verändert. Produkte wie Microsofts System Management Server (SMS) [9], Altiris Client Management Suite [10] oder Novells ZENworks [11] decken nahezu alle Tätigkeiten des Erstellungsprozesses ab, kümmern sich aber nur beschränkt um die Umsetzung der hier genannten Strukturen, zumal meistens nur zwei Schichten – Enterpise und User – betrachtet werden. Die Umsetzung der Verteilungsrichtlinien hingegen ist zu einem viel zentraleren

Thema geworden, als es noch vor fünf Jahren der Fall war. Das sieht man auch an den einzelnen Produkten von Microsoft, Altiris, Novell oder LANDesk.

2.4 Modelle und Richtlinien

Modelle dienen der Vereinfachung und Veranschaulichung von (technischen) Vorgängen. In diesem Fall spiegeln die Modelle die Informationen wieder, die den Datenfluss der Aktionen beschreiben. Diese Daten werden von den im Folgenden dargestellten Modellen erfasst.

2.4.1 Anwendungsmodell

Das Anwendungsmodell ist die Abstraktion der Anwendungssoftware. Sie umfasst alle Informationen, die die Anwendung betreffen. Dazu gehörten die Dateien, die Dokumentation, etc. und eine Beschreibung der Architektur der Anwendung. Die Beschreibung umfasst ihre Komponenten, Voraussetzungen für Hardware und Software, interne und externe Abhängigkeiten zu anderen Programmen oder Programmteilen und diverse andere Informationen (Kontakte, Releasedatum, usw.). Das von Microsoft und Marimba entwickelte *Open Software Description Format (OSD)* [12] und das *Distributable Software Description Format (DSD)* [13] liefern hierfür beispielsweise eine formelle Grundlage. Aber auch die von verschiedenen *Application Management Systems* und *Software Configuration Systems* bereitgestellten Anwendungsmodelle erfüllen ihren Zweck. Die *Extensible Markup Language (XML)* eignet sich besonders gut zur Repräsentation der Modelle und lässt sich leicht mit anderen Tools auswerten. OSD und DSD sind sogar explizit mit XML als Grundlage entworfen. Sie haben die Möglichkeit Softwareabhängigkeiten in Form von gerichteten Graphen darzustellen, was gerade bei modularen Anwendungen notwendig ist (siehe dazu [12] und [13]).

2.4.2 Unternehmensmodell

Das Unternehmensmodell ist die Abstraktion des Unternehmens. Es beschreibt die hierarchischen Organisationsstrukturen des Unternehmens und beinhaltet Abteilungen, Unterabteilungen, Mitarbeiter und deren Positionen und Funktionen. Unternehmensmodelle lassen sich als Organigramme (Organisationsschaubilder) visualisieren. Das Unternehmensmodell kann aber auch die Prozesse und den Datenfluss zwischen Mitarbeitern darstellen, wie es z. B. in [3] angegeben wird: wenn zum Beispiel der Datenfluss für ein (Daten-)Paket von Abteilung A vorschreibt, dass Abteilung B dieses innerhalb von drei Wochen erhalten muss, dann müssen die Aktionen für dessen Verteilung nach Abteilung B innerhalb von drei Wochen fertig sein. Coupaye und Estublier bemängeln abschließend zwar, dass in den meisten Organisationsmodellen eine Verbindung zwischen dem Unternehmen selbst und dem einzelnen Benutzersystem (Rechner) fehlt, diese Beziehung ist aber in den meisten Firmen dadurch gegeben, dass jeder Mitarbeiter an einen eigenen Arbeitsrechner gebunden ist.

2.4.3 Verteilungsrichtlinien und -modelle

Die Verteilungsrichtlinien legen die Zustellungsart, den jeweiligen Empfänger und den Zeitpunkt fest, an dem verteilt werden soll. Wie aus **Abb. 3** ersichtlich wird, ergibt sich das Verteilungsmodell aus dem Unternehmensmodell, den Verteilungsrichtlinien und indirekt über Umwege aus dem Anwendungsmodell. In [3] ist das Verteilungsmodell das Szenario, das von der Verteilinfrastruktur (die beispielsweise eine Prozess-Engine beinhaltet) interpretiert wird, um die tatsächliche Verteilung zum Zielsystem durchzuführen. Ein besseres Verteilungsmodell wäre eine Kombination des bisherigen Modells und des Unternehmensmodells, sodass eine direkte Verknüpfung zwischen der Konfiguration einer Anwendung und der Benutzerseite entsteht. Dann würde dem verteilenden System direkt die Informationen zur Verfügung stehen, welche Konfiguration auf welcher Seite verteilt werden muss (vgl. [3]).

Kontrolle des Prozesses

Ein Verteilungsmodell kann gemäß [3] folgende unterschiedliche Richtlinien bestimmen, um den gesamten Vorgang zu überwachen:

- *Automatisch, manuell* oder *halbautomatisch.* Im automatischen Modus läuft der Verteilungsprozess komplett stillschweigend ab und führt die verschiedenen Tätigkeiten selbstständig durch. Im manuellen Modus wird der Benutzer zur Bestätigungen der Aktionen aufgefordert. Im halbautomatischen Modus kann der Benutzer vorher festlegen, bei welchen Aktionen er nach einer Bestätigung gefragt werden will. Der Rest läuft dann automatisch ab.
- *Wiederherstellbar* oder *ersetzbar:* Bei wiederherstellbar macht das System in seinem letzten konsistenten Stand weiter, falls es einen Absturz während der Verteilung geben sollte. Ist es ersetzbar, so führt es eine andere (ggf. die nächste) Aktion durch.
- *Einfach* oder *verzweigt:* Wenn im einfachen Modus eine Aktivität fehlschlägt, weil bestimmte Ressourcen fehlen, dann benutzt das Verteilsystem die Wiederherstellung oder die Ersetzung wie oben beschrieben. Im verzweigten Modus kümmert sich das verteilende System selbst um die Organisation der fehlenden Ressourcen.
- Ein *Protokoll* sollte erstellt werden, um nachvollziehen zu können, ob die Prozesse richtig arbeiten oder nicht (siehe dazu 4.1 – Reporting).
- usw.

Nach Meinung der Autoren ist auch hier eine Schwachstelle in der Umsetzung der Verteilungsrichtlinien bei gängigen Technologien zu sehen. Dies ist ein weiterer Unterschied den sie zwischen den Verteilungsmodellen, die aus gängigen Anwendungen hervorgehen, und dem des vorgestellten Enterprise Deployments, sehen.

2.4.4 Site Models

Site Models sind Abstraktionen der Zielsysteme (Desktop Computer der Benutzer) die dazu dienen, das Verteilungsmodell des Unternehmens auf die Clientsysteme individuell anzupassen und anzuwenden. Das *Site Model* stellt Hardware (Prozessor, Speicherausbau, Fest-

plattenkapazität, etc.) und Software (OS, Treiber, Anwendungen, bereits verteilte Komponenten, ...) der Clientsysteme dar. Die Informationen über die bereits verteilten Komponenten sind insofern wichtig, da sie von anderen Clients mitgenutzt werden können (shared access; siehe auch Kapitel 3 - Multicast).

Als Site Model kann auf das von der Distributed Management Task Force (DMTF) standardisierte *Common Information Model (CIM)* [14] zurückgegriffen werden. Eine Implementierung wird mittlerweile direkt von aktuellen Betriebssystemen bereitgestellt. Unter Windows (ab NT4.0 SP4) ist das die *Windows Management Instrumentation (WMI)* (siehe dazu [15] und [16]). Mit ihrer Hilfe kann man von der Ferne (remote) z. B. auf die Registry des Systems zugreifen, Prozesse starten und Hardwarekomponenten verwalten. CIM und seine Erweiterung *Web Based Enterprise Management (WBEM) [17]* wird nach heutigem Stand von allen ernst zunehmenden Anwendungen zur Softwareverteilung (LANDesk, SMS2003, Altiris, Tivoli) unterstützt. Die im zugrunde liegenden Artikel genannte *Application Management Specification (AMS)* wurde von CIM abgelöst.

Microsoft beispielsweise spezifiziert im Site Model des Systems Management Servers 2003 Computer, Benutzer, Gruppen und andere Ressourcen die vom SMS gehandhabt werden [9][18].

3 Techniken und Verfahren der Softwareverteilung

„Im Grunde kann man sich ein automatisiertes Softwareverteilungssystem als Transportmechanismus vorstellen. Es dient lediglich als Transportmittel, um einen Auftrag zur Softwareverteilung bzw. Konfiguration vom Softwareverteilungsserver zu einem Client zu transportieren. In welcher Art und Weise dieser Auftrag auf dem Client ausgeführt wird, ist abhängig von der verwendeten Technologie“ ([19]).

Da sich die Techniken und Verfahren zum einen in ihrer Handhabung und zum anderen in ihrem Einsatzbereich stark voneinander unterscheiden können, gilt es für ein Unternehmen abzuwägen, welche Verfahren am günstigsten einzusetzen sind.

Dieses Kapitel gibt einen Überblick über die, in den verschiedenen Produkten zum Einsatz kommenden, Technologien und erläutert diese. Es wird ein Einblick in verschiedene Installations-Methoden (Unattended Setup, Native Installation, MSI), Methoden zur Systemabbildung (SnapShot, Imaging/Cloning) und in Verfahren der Inventarisierung und des Patch-Managements gegeben. Auch auf die, diesen Methoden zugrunde liegenden, Kommunikationsmodelle und Techniken der Netzwerkebene wird eingegangen.

3.1 Netzwerkverfahren

Die Verteilung von Software in einem Netzwerk setzt bestimmte Verfahren und Techniken voraus. Diese schreiben vor, wie der Datentransfer zwischen den einzelnen Teilnehmern (Clients, Server) auszusehen hat oder in welcher Reihenfolge er ablaufen kann. Dabei geht es

nicht nur um den Transfer als solchen, sondern auch um die involvierte Kommunikation. Diese steuert den Fluss der Transferaktion, ihren zeitlichen (oder räumlichen) Ablauf und ihre Durchführung. Sie legen außerdem die Grundsteine für einen effektiven Einsatz der zur Verfügung stehenden Ressourcen.

3.1.1 Push und Pull

Es gibt im Wesentlichen zwei gegensätzliche Herangehensweisen, um den Transfer der bereitgestellten Anwendung auf das Zielsystem zu realisieren: das Push- und das Pull-Modell. Dabei spielt es generell keine Rolle, ob das Paket vom Hersteller zum Unternehmen oder vom Unternehmen zum Endbenutzer (siehe dazu Kapitel 2.2) übertragen werden soll. Beim Push-Modell werden die Pakete vom Quellsystem auf das Zielsystem „geschoben“, das Quellsystem übernimmt hier den aktiven Part, während beim Pull-Modell sich das Zielsystem um den Transfer kümmert und die Pakete aktiv von der Quelle „zieht“. Aus technischer Sicht, wie in [3] vorgeschlagen, zieht eine Kontrolle des Ablaufs der einzelnen Transferaktionen, aus Sicht des Herstellers gesehen, ein Push-Modell nach sich, während die Kontrolle über die Transferaktion, vom Endbenutzer aus gesehen, zu einer Pull-Architektur führt. Bezogen auf unser Drei-Schichten-Modell bedeutet das, dass das Unternehmen eher ein Pull-Verfahren einsetzen wird, um neue Software vom jeweiligen Hersteller zu beziehen und die angepasste Software dann per Push oder er auch per Pull verteilt, je nachdem wie das Systemumfeld (Netzwerkarchitektur, logische Struktur, zeitlicher Spielraum, Verfügbarkeit der Zielrechner) aufgebaut ist. Ein gängiger Weg ist, auf den Zielrechnern so genannte Softwareagenten zu installiert, die dann z. B. bei jedem Neustart oder in bestimmten, festgelegten Intervallen beim Server nachfragen, ob neue Pakete für dieses System vorgesehenen sind (Pull-Architektur). Welches Verfahren dann letztlich eingesetzt wird, hängt zum Großteil von der Umgebung ab, in der es verwendet wird.

Vorteile des *Push*-Modells

Für eine Push-Architektur spricht, dass sie von vornherein darauf ausgelegt ist, verfügbare Ressourcen optimal zu nutzen. Es ist daher abschätzbar, welche Anforderungen an das Netzwerk, die Server, die Clients, etc. durch die Transferaktionen gestellt werden und damit auch welche Kosten sich daraus ergeben. Durch die zentralisierte Kontrolle ist es möglich den zeitliche Ablauf (Scheduling) der einzelnen Transferaktionen gut zu koordinieren. Auf diese Weise lassen sich die verfügbare Bandbreite und die Serverressourcen optimal zuteilen, um z. B. auch bestimmte Gruppen vorrangig oder aber auch segmentweise zu bedienen. Engpässe können so vermieden werden.

Nachteile des *Push*-Modells

Um die Zielsysteme mit aktuellen Paketen versorgen zu können, benötigt das Serversystem ständig aktuelle Informationen über das Inventar seines Verteilbereichs und über die jeweiligen Zielsysteme (Was gibt es für Systeme? Was für Anwendungen sind installiert?). Miss-

stände in diesen Informationen, z. B. wegen neuen Hardwarekomponenten, werden nicht berücksichtigt und machen das System dadurch gegebenenfalls verwundbar. Um eine Push-Architektur für den Verteilungsprozess effizient einsetzen zu können, ist es laut Bradley in [20] daher unabdingbar, Methoden zu haben, die entweder sicherstellen, dass Geräte dem Bestandsverzeichnis (inventory listing) hinzugefügt werden, sobald man sie im Netzwerk einsetzt, oder die zuverlässig regelmäßige Aktualisierungen des Verzeichnisses durchführen. Aufgrund des starren Aufbaus bietet das Verfahren keine Möglichkeit, flexibel auf Änderungen und Ausnahmen zu reagieren. So stellen beispielsweise mobile Endgeräte (Laptops, PDAs) wegen ihrer Mobilität und der damit verbundenen Ungebundenheit zur Netzwerkinfrastruktur ein Problem dar. Ein weiterer Nachteil ist, dass bei einem zeitlich geregelten Ablauf feste Schranken eingehalten und im Gesamten koordiniert werden müssen. Eine Folge dessen ist ein nicht unerheblicher Aufwand für die Überwachung dieser Vorgänge.

Vorteile des *Pull*-Modells

Wie in [20] beschrieben sind die Vorteile, dass beispielsweise einfache, auch systemeigene Dienste (z. B. Logon- oder Startupskripte) verwendet werden können, um eine Verbindung zwischen dem Zielsystem und dem Server herzustellen. Auch sind tiefere Einblicke in die Beschaffenheit der Zielsysteme möglich. So können Einträge aus der Registrierdatenbank oder andere Einstellungen herangezogen werden, um den Bedarf an neuen (erforderlichen) Paketen für das System zu ermitteln und diese dann automatisch vom Server zu beziehen. Im Vergleich zum „pushen“ kann dies effizienter sein und auch mehr erfolgreiche Verteilprozesse nach sich ziehen. Das Pull-Verfahren bietet aber auch Vorteile in verteilten Umgebungen, in denen der Server über Subnetze und über Firewalls hinweg kommunizieren muss, um alle Zielsysteme zu erreichen. Dies ist bei Push-Verfahren oftmals nicht mit einem vertretbaren Aufwand realisierbar, wohingegen man für eine Pull-Umgebung einfach den Server zentral aufstellen kann (z. B. in einer DMZ – entmilitarisierten Zone), sodass ihn alle Zielsysteme erreichen können. Pull-Systeme können wegen ihres flexiblen Aufbaus gut mit Änderungen und Ausnahmen umgehen. Das zeigt sich gerade bei „ad hoc“ Geräten wie PDAs oder Notebooks.

Nachteile des *Pull*-Modells

Der Anspruch, der an die Infrastruktur und die Ressourcen (Netzwerkbandbreite, Serverkapazitäten) gestellt wird, ist wegen des flexiblen Aufbaus im Gegensatz zu Push-Verfahren kaum abschätzbar und die zur Verfügung stehenden Betriebsmittel können nicht optimal genutzt werden. Hinzu kommt, dass sich der Zeitpunkt für einzelne Transferaktionen nicht zeitlich festlegen lässt (kein Scheduling). Daher kann es passieren, dass sich viele Zielsysteme gleichzeitig aktualisieren wollen oder aber andere gar nicht (weil z. B. kein Reboot durchgeführt wird). Und dies kann wiederum zu Engpässen bei der Ressourcenvergabe bzw. zu nichtaktuellen Systemen führen.

Push	Pull
+ optimale Nutzung der Ressourcen	+ flexibler Aufbau und Einsatz
+ Kosten abschätzbar	+ Automatisierung auf Clientseite leichter realisierbar
+ zeitlicher Ablauf gut koordinierbar	+ flexible Ausnahmenbehandlung
+ läuft in geregeltem Rahmen ab	+ Änderungen leicht umzusetzen
- starrer Aufbau	+ Kommunikation in verteilten Umgebungen
- Äderungen aufwendig umzusetzen	- Anforderungen an das Umfeld schwer abschätzbar
- kann mit Ausnahmen schlecht umgehen	- kein Scheduling
- Kommunikation in verteilten Umgebungen	- Engpässe bei der Ressourcenvergabe möglich

Tabelle 1: *Vor- und Nachteile des Push- und Pull-Verfahrens*

Weitere detailliertere Ausführungen zu diesem Thema sind in [21] zu finden.

Während Push und Pull Modelle sind, die vorgeben, wem welche Rolle beim Transfer zuteil wird, legen die folgenden Techniken fest, wie dies auf der IP-Ebene umgesetzt werden kann.

3.1.2 Unicast/Multicast

In einem Netzwerk bezeichnet man die Kommunikation zwischen einem Sender und einer bestimmten Gruppe von Empfängern als Multicast (im Gegensatz zum Broadcast, das alle Empfänger des Netzwerkes anspricht). Neben Multicast gibt es im Internet Protokoll Version 6 (IPv6, [22] und [23]) außerdem noch *Anycast* (ein Sender, nächsten Empfänger aus einer erreichbaren Gruppe) und *Unicast* (ein Sender, ein Empfänger) als Pakettypen.

Das Unicast-Verfahren ist der Standard für Verbindungen zwischen zwei Rechnern und ist daher auch immer die erste Wahl. Microsoft setzt in seinem SMS2003 [9] generell nur Unicast ein (siehe [24]). Der Vorteil hierbei liegt in der zentralisierten Überwachung der Transfervorgänge. Dies bedeutet allerdings auch einen erhöhten Aufwand für die Verwaltung und natürlich eine größere Belastung der Ressourcen.

Multicast ist nur bei einigen wenigen Anwendungen einsetzbar (siehe weiter unten). Bei verteilten Standorten (WAN) ist ein Unicast-Verfahren zur Paketübertragung auf jeden Fall ratsam und wird z. B. auch von Altiris [10] speziell dafür eingesetzt.

Eine Verteilung mittels Multicast hat den Vorteil, dass mehrere Systeme gleichzeitig versorgt werden können. Als Nachteil kann sich die hohe Belastung für das Serversystem, das die Daten bereitstellt, und für das Netzwerk, über das sich der Verteilungsvorgang erstreckt, erweisen. Sind hier ausreichend Kapazitäten vorhanden, spricht jedoch nichts gegen einen Einsatz von Multicast. Altiris setzt z. B. ein Multicast Verfahren im Subnetzbereich ein, um dort benötigte Pakete schneller und koordinierter zu verteilen. Eine Methode die Belastung für die Ressourcen klein zu halten liefert das *Targeted Multicast*.

Targeted Multicast®

Dieses von LANDesk entwickelte und in der LANDesk Management Suite [25] vermarktete Verfahren zielt darauf ab, durch zielgerichtete Paketweitergabe innerhalb von Subnetzen, wertvolle Bandbreite einzusparen. Hierbei wird das Netzwerk in so genannte *Multicast-Domänen* unterteilt. Jede Multicast-Domäne besteht aus den Clients, die jeweils den Broadcast-Verkehr der anderen Clients hören können. Router blockieren normalerweise Multicast-Daten, sodass eine Multicast-Domäne meist einem realen Subnetz entspricht. Im Betrieb wählt der Managementserver einen Client in einer Multicast-Domäne aus – den Domänenrepresentanten, der das verteilte Paket schon in seinem Cache hat. Dieser übernimmt dann die Verteilung der anderen, konkret angegebenen Clients in diesem Subnetz per Multicast. Auf diese Weise lässt sich die Gesamtbelastung für das Netzwerk gering halten. Als Nachteil dieses Verfahrens steht dem Bandbreitengewinn ein erhöhter Aufwand für das Ermitteln des Domänenrepresentanten gegenüber. Dieser Nachteil wirkt sich extrem bei der Verteilung von kleinen Dateien aus, da hier der Administrationsaufwand für den Managementserver den Nutzen übersteigt. Für große Dateien hingegen ist das *Targeted Multicast* Verfahren eine willkommene Optimierung.

3.1.3 Peer-to-Peer

Als Peer bezeichnet man ein System, das sowohl Dienste in einem Netzwerk anbietet als auch Dienste anderer nutzt, also sowohl Server als auch Client in diesem Netzwerk ist. Mit einem Peer-to-Peer Netzwerkes lassen sich Aufgaben zwischen den einzelnen Rechnern eines Bereichs erledigen. Die Aufgaben müssen nicht mehr zentral von einem Server ausgehend erledigt werden und es werden dadurch Kapazitäten frei, und globale Ressourcen geschont. Peer-to-Peer eignet sich daher besonders für den Einsatz in Subnetzen, da sich der Traffic für den Transfer von Paketen dann auf dieses Teilnetzwerk beschränkt. Da die diesbezügliche Verwaltung der Pakete vom Server auf die Clientsysteme übertragen wird, folgt daraus, dass sich jeder Client selbst darum kümmern muss, welche Pakete er benötigt (autonom). LANDesk setzt dieses Verfahren beispielsweise in seinem *Peer Download*® um, um damit in Subnetzen den Bedarf der Clients aus dem Cache anderer Clients zu decken. Dazu überprüft der Client über den Server, ob er neue Pakete benötigt. Ist dies der Fall, so sendet er mittels Broadcast eine Anfrage an die übrigen Clients (Peers) in seinem Subnetz. Hat einer dieser Peers das Paket, so kann der Transfer im lokalen Subnetz erfolgen und es wird kein Datenverkehr über Router oder WAN-Verbindungen hinweg erzeugt. ([25], User's Guide)

Es wäre möglich, mittels Peer-to-Peer ein vollständig autonomes Softwareverteilungssystem zu realisieren. Ein solches System würde allerdings trotzdem einen zentralen Server zur Versionsverwaltung benötigen. Dies ist aber durchaus erwünscht, da sonst jeder Client neue Paketversionen in das Netzwerk einbringen könnte. Eine sinnvolle Kontrolle der Inhalte wäre so jedoch nicht realisierbar.

3.2 Unattendend Setup

Das Unattended Setup ist ein Verfahren, bei dem die Anwendung völlig unbeaufsichtigt vom Benutzer auf dem Zielsystem eingespielt werden kann. Dazu wird dem Setupprogramm eine Antwortdatei übergeben, die es anschließend auswertet. Die Installation kann dann völlig ohne Interaktion des Benutzers ablaufen. Dieses Verfahren ist weit verbreitet und existiert so lange es Installationsprogramme gibt. Oft wird auch zusätzlich auf eine detailierte, visuelle Darstellung des Installationsvorgangs verzichtet (silent install), das Setupprogramm läuft im Hintergrund ab.

Es gibt verschiedene Methoden, wie das Setupprogramm die Installationsparameter erhält. Sie können entweder dem Setupprogramm direkt auf der Kommandozeile mit übergeben werden oder aus einer Datei ausgelesen werden. Solche Antwortdateien werden meist vom Hersteller mitgeliefert und müssen nur angepasst werden. Bei komplexeren Anwendungen gibt es auch Frontends die die Parametereingabe erleichtern. Für die Installation von Windowssystemen (≥2000) stellt Microsoft ein solches Tool zur Verfügung (setupmgr.exe, mehr dazu in Kapitel 5.1 – RIS/ADS/Longhorn Deployment). Eine andere Möglichkeit ist, sich die Antwortdatei automatisch erzeugen zu lassen. Dazu wird das Installationsprogramm (in der Regel setup.exe) mit einem speziellen Parameter – bei *InstallShield* [26] ist dies beispielsweise „/r“ – aufgerufen. Es werden dann während des weiteren Installationsverlaufs alle Eingaben in einer Datei gespeichert. Diese kann dann für weitere Installationen verwendet werden.

Parameter in den Antwortdateien können nahezu beliebige Werte annehmen – vom Namen des Programmverzeichnisses über Seriennummern bis hin zu Ländereinstellungen – je nachdem, was die Anwendung benötigt. Es gibt sogar spezielle Programme, die dann solche Eingaben während des Installationsvorgangs so eintragen, als ob der Benutzer tatsächlich vor dem Rechner sitzen würde. Selbst die Mausklicks werden vom Programm ausgeführt. Dieses Verfahren wird zur Zeit nur von wenigen Produkten wie etwa *baramundi Deploy* [27], *Prism Deploy* [28] oder Symantecs *ON Command CCM* [29] unterstützt. Als selbstständige Programme seien hier noch *WinRobots* und die Freeware *AutoIt* [30] und *AutoHotkey* [31] genannt.

```
[Unattended]
Unattendmode = FullUnattended
OemPreinstall = NO
TargetPath = *
Filesystem = LeaveAlone

[GuiUnattended]
TimeZone = "110"
AdminPassword = *
AutoLogon = Yes
AutoLogonCount = 1

[LicenseFilePrintData]
AutoMode = "PerServer"
AutoUsers = "5"

[Display]
BitsPerPel = 16
```

Abb. 7: *Auszug aus der Unattended.txt von WindowsXP*

Die starke Bindung einer unbeaufsichtigten Installation zu den Antwortdateien kann sich aber auch als Problem erweisen. In Einzelfällen kann es schwer sein, an die Informationen über die Einstellungsmöglichkeiten der Antwortdatei zu gelangen. Es sind auch nicht alle Parameter einer Anwendung über eine solche Antwortdatei zugänglich, da der Hersteller in der Regel nur grundlegende Eigenschaften für das Unattended Setup implementiert. Gibt es mehrere, unterschiedlichen Varianten ein und desselben Setups und damit mehrere Antwortdateien, kann dies zu Unübersichtlichkeit führen. Besser wäre es die Antworten aufgrund einer Vorlagendatei dynamisch zur Laufzeit zu erzeugen. Dadurch kann das Verfahren automatisiert werden, ohne dabei Flexibilität einzubüßen [19].

Ein Vorteil der unbeaufsichtigten Installation ist, dass auf die „Intelligenz“ des Setupprogramms zurückgegriffen werden kann. Da immer noch das originale Setup ausgeführt wird,

stehen auch die Mechanismen, die sich der Entwickler dafür ausgedacht hat, bereit. Das bedeutet, dass Fallunterscheidungen durchgeführt werden können, die Auskunft geben, ob genügend Platz auf einem Laufwerk frei ist oder ob bestimmte Parameter in der Registry eingetragen werden müssen, weil eine andere Version bereits installiert ist usw.

Der entscheidendste Vorteil einer bedienerlosen Installation ist, dass man so gleichzeitig mehrere Installationen auf verschiedenen Systemen durchführen kann, ohne dabei ständig von einem System zum nächsten rennen zu müssen. Sie eignet sich daher hervorragend zum Einsatz in der Softwareverteilung.

3.3 Native Installation

Ähnlich wie beim Unattended Setup ruft die Installationsroutine das Setup (meist „setup.exe“) auf, aber anstatt dessen Verlauf mit einer Antwortdatei zu steuern wird ein Skript verwendet. Dieses Skript enthält also die Befehle und Eingaben, die beim Setup getätigt werden müssen um die Installation erfolgreich durchzuführen. Die Fenster der Installationsroutine werden von dem Script erkannt und dieses führt dann die notwendigen Maus- und Tastatureingaben durch. Es werden die gleichen Installationsschritte nachgebildet, die ein Benutzer manuell beim Aufruf der „setup.exe“ ausführen würde.

Vorteile der Native-Installation-Methode

Es besteht auch die Möglichkeit, das Script mit Abfragen zu versehen, um je nach Installationsverlauf auf einem Clients unterschiedlich reagieren zu können. Das ist einer der grundlegenden Unterschiede zum Unattended Setup, denn dadurch kann die Installationsroutine wesentlich flexibler mit dem Client interagieren. Mit dieser Technik ist es zum Beispiel möglich, eine existierende, ältere Version der, zu installierenden Anwendung, vorher automatisch deinstallieren zu lassen - natürlich nur, sofern die, vom Hersteller angebotene Installationsroutine, das auch unterstützt bzw. erfordert. Zusätzlich besteht die Möglichkeit, dem Script clientspezifische Parameter zu übergeben. Somit kann man mit nur einem einzigen Script mehrere Installationsvarianten realisieren.

Auch Deinstallationsroutienen an Clients lassen sich mit diesem Verfahren durchführen. Damit kann man auch Software von Clients deinstallieren, die nicht mit dem Softwareverteilungssystem installiert wurde. Das kann beispielsweise unerwünschte oder lizenzpflichtige Software sein, die der Benutzer eigenhändig auf seinem Client installiert hat. Gerade in einer Zeit, in der Lizenzierung eine wichtige Rolle spielt und Lizenzkontrollen an Clients durchgeführt werden müssen, ist das ein nicht zu verachtender Vorteil dieser Installationsmethode.

Die Installation einer Applikation ist sowohl bei der Native Installation, als auch beim Unattended Setup, in der Regel unabhängig vom eingesetzten Betriebssystem. Daher eignet sie sich besonders gut für Unternehmen, die mehrere verschiedene Betriebssysteme parallel auf ihren Clients einsetzen.

Ein Merkmal dieses Verfahrens ist zudem die Flexibilität bei der Installation von Anwendungen, bei denen sich dieser Vorgang von System zu System stark unterscheidet. Die Unterschiede können sowohl durch Hardwarekomponenten, als auch durch Abhängigkeiten

von Softwarekomponenten bestimmt sein. Solche Installationen können mittels Snapshot oder Imagingverfahren meist nur auf baugleichen Clients verteilt werden. Beide Techniken werden weiter unten detaillierter beschrieben.

Durch die Native Installation lassen sich Anwendungen aber nicht nur verteilen und installieren, man kann damit auch Applikationen konfigurieren. Die Konfigurationsfenster von Anwendungen lassen sich mit dieser Methode genauso durchnavigieren wie die Installationsroutine. Damit sind tief greifende Kenntnisse der Registry, in der die meisten Anwendungen ihre Konfigurationen speichern, nicht mehr notwendig.

Nachteile der Native Installation

Zu beachten ist jedoch, dass die Skripte für diese Methode vom Hersteller des Softwareverteilungssystems oder dem Unternehmen, bei dem es zum Einsatz kommt, erst erstellt werden müssen. Dies kann, je nach Umfang und Komplexität eines Skriptes, eine gewisse Zeit in Anspruch nehmen. Dabei spielt auch die Anzahl der Installationsvarianten und Optionen der Installationsroutine der Applikation eine entscheidende Rolle. **[19]**

3.4 Snapshot-Verfahren

Die Snapshot-Methode ist auch als Differenzabbild- und SysDiff-Verfahren bekannt. Sie wird von vielen Softwareverteilungssystem unterstützt. Bei diesem Verfahren wird keine Installationsroutine ausgeführt, sondern lediglich die Veränderungen in der Registry und im Dateisystem auf den Client übertragen.

Vorgehensweise zur Erstellung eines „Snapshots“ nach [19]:

1. Manuelle Installation eines Clients in einen definierten Zustand(IST-Zustand), z.B. Betriebssystem inkl. Service Packs.
2. Aufzeichnung des IST-Zustandes → Abbild des IST-Zustandes
3. Manuelle Installation einer Applikation oder Durchführung einer Konfiguration (veränderter Zustand)
4. Aufzeichnung des veränderten Zustandes → Abbild des SOLL-Zustandes
5. Differenzbildung aus dem IST-und dem SOLL-Zustand. Darin enthalten sind alle Änderungen der Datei- und Verzeichnisstruktur des Clients, sowie Unterschiede in dessen Registry.

Vorteile der Snapshot-Methode

Dadurch, dass die aufgezeichneten Veränderungen der manuellen Installation nur auf den Client kopiert werden müssen, handelt es sich hierbei um eine schnelle Methode, Software zu verteilen.

Die Snapshot-Methode hat zudem einen positiven Nebeneffekt. Da alle Änderungen im Differenzabbild enthalten sind, können jederzeit Vergleiche gegenüber dem Stand des Clients gemacht werden. So können nachträgliche Änderungen der Applikation am Client, wie das

Fehlen einer wichtigen Systemdatei, erkannt werden. Diese Änderung kann dann, durch erneutes Einspielen des Snapshots, korrigiert werden. Intelligente Softwareverteilungssysteme kopieren dabei nur die, zur Korrektur notwendigen, Dateien. Somit lassen sich mit der Methode ausgerollte, aber nicht mehr funktionsfähige Applikationen, auf dem Client schnell wieder reparieren.

Auch Deinstallationen sind mit dieser Technik möglich. Dabei werden die, im Differenzabbild gespeicherten Änderungen, am Dateisystem und der Registry einfach wieder entfernt. Viele, von den Herstellern gelieferte Deinstallationsroutienen hinterlassen oft Reste der Anwendung in der Registry und im Dateisystem. Diese können dann bei späteren Installationen zu Problemen führen. Die Deinstallation mit der Snapshot-Methode führt dann meist zu einem saubereren Resultat auf dem Client.

Nachteile der Snapshot-Methode

Durch die Funktionsweise dieses Verfahrens, ergeben sich allerdings auch einige Nachteile. Eine dynamische Steuerung der Installationsroutine ist, im Gegensatz zur Unattended und Native Installation, hierbei nur sehr begrenzt möglich.

Da ein Snapshot nur aus Dateien und Änderungen am Dateisystem sowie der Registry besteht, kann man hieraus keine Varianten der Applikation erstellen. Damit ist ein Snapshot an das, auf dem Client eingesetzte Microsoft Betriebssystem und dessen Sprachversion gebunden. Das hat einen erheblichen Mehraufwand für Unternehmen zur Folge, in denen, aus dem eben genannten Grund, verschiedene Varianten der selben Anwendung verteilt werden müssen. Für jede dieser Varianten muss dann ein eigener Snapshot angelegt werden. Dazu gehört auch die Vorbereitung des Clients, die manuelle Installation und die Differenzbildung.

Es gibt auch Applikation, deren Installationsroutine sehr stark von der Hardware, dem Betriebssystem oder der, auf dem System laufenden Anwendungen abhängt. Diese lassen sich mit dem Snapshot-Verfahren nur mühsam verteilen. Schon ein kleiner Unterschied des Clients, auf dem der Snapshot erstellt wurde, zu dem Client, auf den die Applikation mit verteilt werden soll, können dazu führen, dass die Anwendung auf dem Zielsystem nicht richtig oder sogar überhaupt nicht funktioniert.

Ein weiterer Nachteil ist, dass die Deinstallation nur bei Applikationen funktioniert, die auch mit diesem Verfahren verteilt wurden. Somit können manuelle Installationen, die ein User auf dem Client gemacht hat, nicht wieder deinstalliert werden. Diese Notwendigkeit ergibt sich zum Beispiel bei lizenzpflichtigen Anwendungen.[19]

3.5 Windows Installer (MSI)

Hierbei handelt es sich um relationale Datenbanken, die den Zielzustand nach erfolgreicher Installation beschreiben. Im Gegensatz zu herkömmlichen Setups, erfolgt die Installation nicht mehr prozedural durch das Abarbeiten von Scripts, sondern regelbasiert. MSI steht für *Managed-Software-Installation* und kennzeichnet als Erweiterung ".msi" die Windows-Installer-Dateien.

Diese Methode kann als eine Art Mischung zwischen einem Snapshot und einem Unattended

Setup betrachtet werden **[19]**:

Snapshot deshalb, da das MSI-Paket im Grunde eine relationale Datenbank ist, die den Zustand des Rechners nach erfolgreicher Installation der Software beschreibt. In dem Paket ist ebenfalls festgelegt, nach welchen Regeln die Installation zu erfolgen hat.

Unattended Setup, weil der Aufruf der Installation mit der Übergabe einer Antwortdatei, des so genannten Transforms, erfolgen kann, und sich dann wie ein Unattended Setup verhält.

Dabei ist MSI wesentlich mächtiger als beide Methoden zusammen. Die MSI-Pakete kommen entweder direkt vom Hersteller der Anwendung oder müssen mit speziellen Tools selber erzeugt werden. Einige Softwareverteilungssysteme bieten die Möglichkeit, aus zuvor gemachten Snapshots, MSI-Pakete zu erstellen, um diese dann an die Clients verteilen zu können. Die Installation der MSI-Pakete wird durch den Windows Installer Dienst übernommen. Dieser ist ab Windows 2000 standardmäßig im Systemumfang enthalten. Unter Windows 95 / 98 und NT 4.0 muss dieser nachinstalliert werden. Treten bei der Installation des MSI-Paketes Fehler auf, stellt das automatische Rollback den Zustand des Clients wieder her, der vor dem Beginn der Installation vorzufinden war. Mit den MSI-Paketen kann zudem eine Deinstallation und Reparatur (Selfhealing-Funktion) der Applikationen durchgeführt werden. Sofern eine Applikation aus mehreren, unabhängigen Komponenten besteht, kann man diese auch bei der Erstinstallation weglassen und später installieren oder später auch andere Komponenten wieder deinstallieren.

3.6 Imaging/Cloning

Einige Softwareverteilungssysteme arbeiten nach dem „Imaging"- oder „Cloning"-Verfahren. Dabei wird von einem Referenzsystem ein bitgenaues Abbild der Festplatte, das Image, erstellt. Dieses Duplikat beinhaltet dann das Betriebssystem inklusive aller Applikationen und Anpassungen. Dieses kann dann auf weitere Clients mit identischer Hardware verteilt werden. Wie bei der Snapshot-Methode muss man dabei ein einzelnes System zuvor manuell installieren. Alle anderen Systeme können dann automatisch davon kopiert werden. Bei der manuellen Installation kann man dann auch schon benötigte Anwendungen integrieren, oder sonstige Anpassungen vornehmen, die dann auf die anderen Systeme mit übertragen werden sollen.

Dieses Verfahren wird meist nur zur Grund- oder Vorinstallation von Clients verwendet. Nach dem „Roll-Out“ werden dann die anderen, wesentlich flexibleren Methoden zur Softwareverteilung bevorzugt.

Vorteile des Image / Cloning Verfahrens

Der bedeutendste Vorteil dieser Methode ist die Geschwindigkeit. Mit keinem anderen Verfahren ist es möglich, das Betriebssystem inklusive der Applikationen so schnell auf baugleiche Systeme zu verteilen. Deswegen wird zur Verteilung der Images auch oft Multicasting (siehe Kapitel 3.1.2) eingesetzt um mehrere Clients zeitgleich betanken zu können.

Des Weiteren kann man auf dem Referenzclient bereits viele Einstellungen vornehmen, die in der Systemlandschaft des Unternehmens benötigt werden.

Somit ist es innerhalb von kürzester Zeit möglich, einen fabrikneuen Client mit dem Betriebssystem und allen nötigen Anwendungen auszustatten, um ihn gleich im Arbeitsalltag nutzen zu können.

Nachteile des Image / Cloning Verfahrens

Unglücklicherweise ist es auch ein sehr starres Verfahren, denn ein Image lässt sich nur auf einen hardwareidentischen Client verteilen. Um Clients, deren Hardware sich von der des Referenzclients unterscheidet, zu versorgen, muss man wieder eigens ein Image durch eine manuelle Installation erstellen. Eine Personalisierung oder Variation des Systems muss dann entweder nachträglich von Hand erledigt werden oder durch Nutzung anderer Technologien der Softwareverteilung geschehen. Solche Anpassungen sind aber in jedem Fall notwendig, da die Clients auf denen das Image verteilt wird, erstmal identisch sind. Das bedeutet sie besitzen die gleiche SID (Security Identifier), den gleichen Computernamen und die gleiche IP-Adresse (falls kein DHCP-Server zum Einsatz kommt). Das führt unweigerlich zu Kollisionen im Netzwerk. Einige Cloning-Tools wie Symantec Ghost (Enterprise Version, [32]) oder Altiris RapiDeploy [33] besitzen bereits Methoden, die nach dem Verteilen des Images diese Daten ändern. Es gibt aber auch ein Open Source Tool, NewSID von Sysinternals [34], das von Hand in den Verteilungsprozess integriert werden kann.

3.7 Inventarisierung und Licensing

Ein weiterer, zentraler Aspekt eines Softwareverteilungssystems ist das Inventory-Management. Darunter versteht man den Prozess, aktuelle Informationen über Hard- und Softwarekomponenten über das Netzwerk zu erhalten und in eine Datenbank zu speichern. Dadurch entfällt das mühsame manuelle Anlegen und Aktualisieren von Inventarlisten und die Daten sind stets aktuell. Die zentral abgelegten Informationen stehen so auch für Problemlösungen des User Help Desks oder als Informationsquelle für Abschreibungen zur Verfügung, wie in [35] geschildert wird. Der Hauptnutzen der gespeicherten Daten bleibt aber die Verwendung in der Softwareverteilung. Es können so schnell detailierte Informationen über die Beschaffenheit der einzelnen Systeme ausgelesen werden und so in die Planung des Verteilungsprozesses mit einfließen.

Licensing

Aber auch die Lizenzen der, zur Verfügung stehenden Softwareprodukte, lassen sich über Inventory-Managementwerkzeuge komfortabel verwalten. Das Softwareverteilungsprogramm kann somit bei Programminstallationen gegebenenfalls Lizenzen vom Lizenzserver automatisch anfordern oder diese beim Entfernen von Programmen auch wieder zurücknehmen. Die Möglichkeit Lizenzen im voraus für bestimmte Vorgänge zu reservieren ist außerdem gängig. Der Administrator hat so schnell einen Überblick über die eingesetzten, benötigten und noch verfügbaren Lizenzen.

3.8 Patch Management

Laut Schätzungen der Garner Group fanden 2005 mehr als 90 Prozent aller Angriffe auf Netzwerke über Sicherheitslücken statt, für die bereits Patches veröffentlicht waren. ([36]) Aus diesem Grund ist der Einsatz eines Werkzeuges für das Patch-Management eine logische Konsequenz.

Das Patch-Management durchlebt gerade in letzter Zeit einen regelrechten Hype. Fast jeder Hersteller für Softwareverteilungslösungen hat mittlerweile für diesen Teilbereich spezielle Zusatzmodule oder eigene Programlösungen zu bieten, wie etwa der *Microsoft Baseline Security Analyzer* (MBSA) [37].

Das gesetzte Ziel eines Patch-Management Tools ist es, möglichst automatisch bekannte Fehler zu beheben und kritische Sicherheitslöcher zu stopfen. Dabei darf die Stabilität eines Systems durch die Installierten eines Patches nicht beeinträchtigt werden. Nichts ist schlimmer als ein System, das nach der Installation eines Servicepacks instabiler läuft als zuvor. Deswegen ist gerade die Analyse der Abhängigkeiten zwischen Patches und Systemen wichtig. Da sich dies nur durch aufwendige Tests bewerkstelligen lässt, ist nach [36] ein manueller Ansatz wenig sinnvoll. Allein die Abhängigkeiten zwischen Einträgen in der Registrierdatenbank und den DLL Versionen auf Windowssystemen ist sehr komplex. Ein automatischer Analysevorgang kann hier wertvolle Zeit sparen und ist zudem meist korrekter.

Die Informationsgrundlage für die Analyse der Abhängigkeiten wird durch die Inventarisierung geschaffen. Hier werden die Patch-Stati und betroffene Anwendungen der einzelnen Systeme erfasst. Die anschließende Auswertung der Softwareabhängigkeiten ist ebenfalls sehr aufwendig. Es gibt diverse Bedingungen, die neben der zu aktualisierenden Softwarekomponente selbst (Betriebssystem, Anwendung, Treiber etc.) ausgewertet werden müssen, wie etwa die Voraussetzungen für die Installation des Patches, Alternativen dazu, neuere Softwareversionen, zusammenfassende Servicepacks et cetera. Der Anbieter eines Patches gibt diese Systembedingungen und die Dringlichkeit der Aktualisierung in der Regel an. Eine Patch-Managementlösung muss dies automatisch auswerten und berücksichtigen können.

Nach der Analyse lassen sich die Patches dann weit gehend automatisch verteilen und installieren. An dieser Stelle sollte auch das Desaster-Recovery ansetzen, um gegebenenfalls den Rechner nach fehl geschlagenen Installationen wiederherstellen zu können.

Ein Patch-Report bietet die Informationen über den Patchbedarf, die Dringlichkeit, die behobenen Fehler der Software und den Status der Patchinstallation. Ein gutes Tool zum Patch-Management sollte einen solchen Patch-Report liefern können.

Eine Patch-Managementlösung muss außerdem Rechner rollengemäß beliebig gruppieren können, um bestimmte Nutzergruppen und Verantwortungsbereiche zu trennen. Hier ist ein Softwareverteilungssystem mit Patch-Management gegenüber einer reinen Patch-Managementlösung im Vorteil, da das Rollenmodell dort bereits vorhanden ist und hierfür leistungsfähige Werkzeuge zur Verfügung stehen.

Auch aus anderen Gründen ist eine Integration des Patch-Managements in eine Software-Managementlösung sinnvoll: Einzellösungen sind in der Regel nicht plattformübergreifend einsetzbar; als Folge muss für jede, in einem Unternehmen unterstützte Plattform, ein gesondertes Werkzeug gepflegt oder gar implementiert werden. Softwareverteilungssysteme hingegen unterstützen überwiegend mehrere Plattformen und der Umgang mit diesen Werk-

zeugen ist bereits bekannt. Generell lässt sich das Patch-Management auch durch reine Softwareverteilungssysteme abdecken. Man muss dann allerdings auf spezialisierte Analysefunktionen und Reporte verzichten, da die Patches lediglich wie „normale“ Anwendungen behandelt werden können. Deshalb bieten einzelne Hersteller wie etwa Altiris [38], baramundi [27] oder LANDesk [25], zu ihren Lösungen ein ergänzendes Patch-Management an.

4 Weitere Aspekte eines Deploymentsystems

Neben den Technologien und Eigenschaften, die ein jedes Softwareverteilungssystem beinhalten muss, gibt es noch eine Reihe von Features die den Umgang erleichtern und die Sicherheit bei der Benutzung des Systems gewährleisten sollen. Zu diesen Features gehören Reporting, Hierarchiebildung, Hardwareunterstützung und Systemvorbereitung, Migration, sowie Methoden welche die Sicherheit eines Softwareverteilungssystems garantieren sollen. Einige von diesen Features sind in vielen Systemen bereits fest integriert, andere wiederum lassen sich nur bei einigen wenigen Softwareverteilungssystemen finden.

4.1 Reporting

Unter Reporting versteht man die Darstellung und Auswertung von Systeminformationen die ein Client an ein Softwareverteilungssystem sendet. Zum Reporting gehören Status-, Warn- und Fehlermeldungen, die vom Client an den Server gesendet werden. Oft wird der Begriff aber weiter gefasst und um die Auswertung von Inventardaten erweitert.

Zu den Systeminformationen gehören Hardware Inventardaten wie BIOS-Version, Hersteller und Typ des Mainboards oder auch die angeschlossenen Netzwerkadapter. Auch Software Inventardaten wie das Betriebssystem und die installierten Service Packs, sowie die installierten Anwendungen gehören zu den Systeminformationen, die vom Client gesendet werden.

Anhand dieser Daten kann man Gruppen von Clients bilden, wie zum Beispiel alle Systeme mit einem bestimmten Typ von Chipsatz oder alle Systeme bei denen die Festplattenkapazität langsam zu neige geht. Damit ist es möglich eine bestimmte Anwendung nur auf einer speziellen Gruppe von Clients zu installieren, oder andere Aktionen für diese Gruppe zu definieren

Ein wichtiger Teil der Systeminformationen sind die Status-, Warn- und Fehlermeldungen der Clientsysteme. Diese beinhalten unter anderem Informationen über den Verlauf, Erfolg oder auch Misserfolg einer Installation, die vom Softwareverteilungssystem angestoßen wurde. So kann man sich beispielsweise alle Clientsysteme anzeigen lassen, bei denen die Installation einer Anwendung fehlerhaft war und auf diesen Clients dann die Installation eventuell wiederholen lassen.

In den meisten Softwareverteilungssystemen sind bereits eine Vielzahl an Reports vordefiniert. Bei Bedarf können aber auch eigene Reports erstellt werden. Dafür müssen in der Regel Datenbankabfragen formuliert werden, die die gewünschten Werte abfragen. Diese Abfragen müssen je nach Softwareverteilungssystem entweder umständlich von Hand oder

bequem mit Hilfe von Abfrage-Assistenten erstellt werden.

Reporting ist ein wichtiges und sehr hilfreiches Feature, wenn es um die Wartung einer Unternehmensumgebung geht. Daher ist dieses Feature auch in fast allen Softwareverteilungssystemen zu finden.

4.2 Hierarchiebildung

Die Hierarchiebildung ist eine Strukturierung und Verteilung einer Installation eines Softwareverteilungssystems auf verschiedene Systeme. Meist sind diese Systeme Server, auf denen Teile des Softwareverteilungssystems installiert werden, damit die Systeme untereinander kommunizieren können. Diese Kommunikation kann dann entweder über das LAN oder gar über ein WAN vollzogen werden. In kleinen bis mittelgroßen Unternehmen ist diese Aufteilung meist nicht notwendig oder sogar überdimensioniert, da sich eine gewisse Anzahl an Clients problemlos von einem einzigen Server verwalten lassen.

4.2.1 Gründe für Hierarchiebildung

Es gibt mehrere Gründe für Hierarchiebildung bei einem Deploymentsystem.

Der wichtigste Grund sind verteilte Standorte in der Infrastruktur. Das Firmennetzwerk ist zum Beispiel auf mehrere Standorte und/oder Subnetze verteilt. Diese Standorte sind eventuell nur mit einem langsamen WAN miteinander verbunden. Um die Netzlast so gering wie Möglich zu halten installiert man an jedem Standort eigens einen Server, mit dem dieser Standort verwaltet wird. Damit kann ein Softwareverteilungssystem sowohl über LAN als auch über WAN effizient eingesetzt werden.

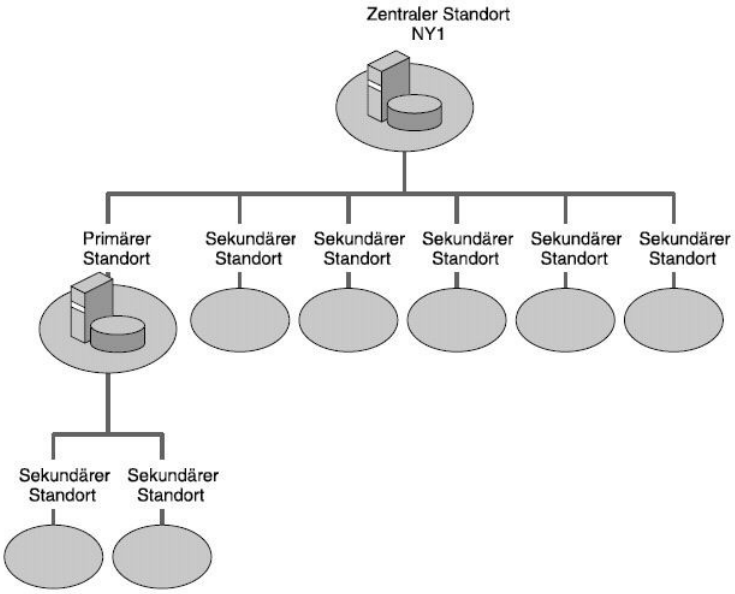
Auch Performancegründe können zur Verteilung der Anwendung führen. Ab einer gewissen Anzahl von Clients ist es sinnvoll diese auf mehrere Systeme zu verteilen, um die Auslastung der Server zu minimieren.

Eine Hierarchiebildung kann aber auch einen, nur verwaltungstechnischen Grund haben. Zum Beispiel um verschiedene Abteilungen, die jeweils separat administriert werden, voneinander zu trennen.

4.2.2 Hierarchiemodelle

Es gibt zwei Arten von Hierarchiemodellen, die bei der Planung eines Softwareverteilungssystems angewendet werden können. Dabei sollte die Modellierung auf die Bedürfnisse des Unternehmens und den Verwaltungsanforderungen angepasst werden. Die beiden Modelle können aber je nach Bedarf auch miteinander kombiniert werden.

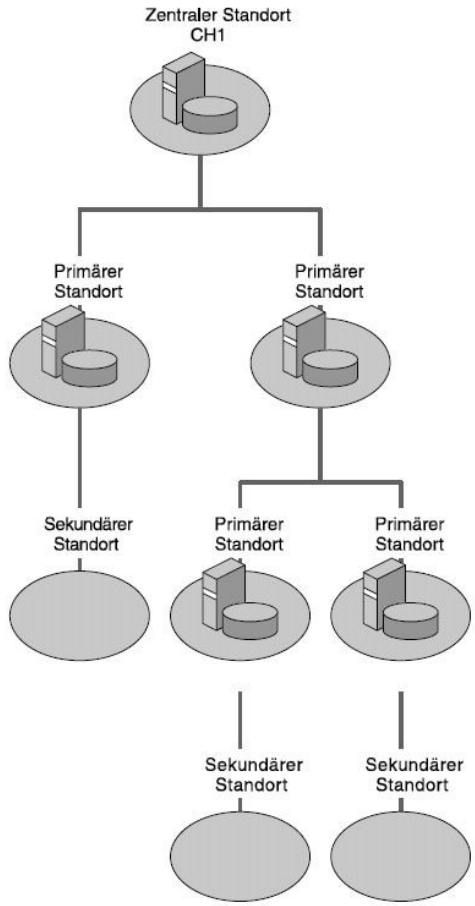
Bei beiden Modellen werden die Verwaltungs- und Konfigurationsdaten nach unten weitergegeben und die Ressourcen und Clientdaten zu den oberen Ebenen der Hierarchie befördert.



Flaches Hierarchiemodell

Das flache Hierarchiemodell besteht in der Regel aus bis zu drei Schichten.

Abb. 8: *Beispiel für eine flache Hierarchie [39]*



Tiefes Hierarchiemodell

Das Tiefe Hierarchiemodell besteht im allgemeinen aus drei oder mehr Ebenen.

Abb. 9: *Tiefes Hierarchiemodell [39]*

4.3 Hardwareunterstützung (System Preparation)

Bevor neue Software auf einem System installiert werden kann, muss sichergestellt sein, dass der Installationsvorgang auch ohne Probleme abläuft. Dazu ist es teilweise notwendig einige Anpassungen an dem System und/oder der Software durchzuführen. Während es bei der Installation von einfachen Anwendungen meist nur um eine Erfüllung von Ressourcenbedingungen (freier Festplatten- und Arbeitsspeicher, CPU-Leistung) und Softwareabhängigkeiten geht, müssen bei einer Betriebssysteminstallation wesentlich mehr Faktoren beachtet werden. Softwareabhängigkeiten spielen bei einer Neuinstallation keine Rolle, sondern nur die Abhängigkeiten zwischen der Hardware und dem Betriebssystem.

Um ein neues Betriebssystem (OS – Operating System) zu installieren, muss zunächst garantiert werden, dass dieses auch mit der gegebenen Hardware funktioniert. Um dies herauszufinden, gibt es grundsätzlich zwei verschiedene Methoden: Man kann die Hardwaredaten entweder aus einem Inventar-Verzeichnis auslesen, falls die Maschine bereits in diesem aufgelistet ist, oder das System mit einer speziellen Software starten, die die Hardwaredaten dann über das Netzwerk, in das Inventar-Verzeichnis einträgt. Sind die Daten schließlich ermittelt, kann entschieden werden, ob für die Installation des Betriebssystems noch spezielle Treiber für bestimmte Komponenten (Grafik, Sound, Storage, ...) eingebunden werden müssen. Zum Einbinden von Treibern für eine Windowsinstallation sollten Tools von der Verteilungssoftware angeboten werden, was jedoch bei kaum einem Hersteller der Fall ist. Man kann sich aber auch mit freien Programmen wie *nLite* (www.nliteos.com) behelfen und ein passendes ISO-Image des OS erzeugen oder gleich die Treiber selbst einbinden. Eine andere Möglichkeit wäre, ein passendes Betriebssystemimage von einer baugleichen, bereits installierten Maschine, abzuziehen und dieses dann als Grundlage für weitere Installationen zu verwenden. Näher wird in den Kapiteln 3.6 (Imaging/Cloning) und 5.1 (RIS/ADS/Longhorn Deployment) hierauf eingegangen.

Hat man die Software soweit auf die Hardware abgestimmt, muss im nächsten Schritt die Hardware angepasst werden. Hierbei müssen eventuell Einstellungen im BIOS modifiziert (Bootreihenfolge ändern, Wake-On-LAN aktivieren, HyperThreading ausschalten, Watchdog deaktivieren etc.) und unter Umständen die Festplatte eingerichtet (RAID) und partitioniert und formatiert werden. Beispielsweise sollte jeweils eine eigene Partition für das Betriebssystem und die Anwendungen und eine für die Benutzerdaten erstellt werden, um bei einer Neuinstallation lediglich die Systempartition formatieren zu können. Um diese Änderungen durchführen zu können, lädt man beim Starten des Systems meist ein spezielles Bootimage über das Netzwerk. Fast immer wird hierfür PXE (**P**reboot E**x**ecution **E**nvironment, [40]) als Verfahren eingesetzt (gibt es auch als Emulation auf einer Bootdiskette, falls die es nicht von der Karte direkt unterstützt wird). Dabei bezieht die Software des Netzwerkadapters zuerst von einem DHCP Server eine IP-Adresse und sendet dann ein Broadcast-Paket, auf das der PXE-Server antwortet. Dieser teilt der Software dann die Adresse des TFTP-Servers und das entsprechende Bootimage mit. Der Rechner kann dann mit dem heruntergeladenen Image genau wie von Diskette (oder CD) booten und entsprechende Tools ausführen, die die Modifikationen am System vornehmen. Mittlerweile gibt es sogar die Möglichkeit eine komplette Windowsumgebung über das Netzwerk zu starten (WinPE). Diese kann auch zur Hardwareerkennung genutzt werden, da sie über dieselben intelligenten Mechanismen wie das Betriebssystem verfügt. Oft lassen sich so auch spezielle Programme starten, die zur weiteren Konfi-

guration und Systemvorbereitung notwendig, aber unter DOS so nicht realisierbar sind. Beispielsweise wäre es denkbar einen Softwareagenten laufen zu lassen, der die Hardware dann in das Inventar einträgt (siehe oben).

4.4 Systemmigration

Unter Systemmigration versteht man die Übertragung von Benutzerdaten, sowie Anwendungs- und und Systemeinstellungen von einem System auf ein anderes.

Der Hauptgrund für eine Systemmigration ist eine Neuinstallation des Betriebssystems. Meist wird dabei ein älteres Betriebssystem durch ein neues ersetzt. Dabei kommt ein Überspielen des alten Systems nicht in Frage, weil man entweder die „Altlasten“ des in die Jahre gekommenen Betriebssystems nicht mitschleppen möchte oder aber einfach gleichzeitig auf neue Hardware umsteigen will.

Damit der Mitarbeiter wieder schnellstmöglich seiner Arbeit nachgehen kann, muss seine gewohnte Arbeitsumgebung auf das neue Betriebssystem übertragen werden. Bei einer Systemmigration geschieht das völlig automatisch. Der Mitarbeiter verschwendet keine wertvolle Arbeitszeit mit dem manuellen Ändern von Anwendungs- und Systemeinstellungen.

Folgendes kann durch die meisten Systemmigtationsanwendungen vom alten System auf das neue übertragen werden:

- Ausgewählte Dateien und Verzeichnisse
- Favoriten (Internet Explorer)
- Desktopeinstellungen, Desktophintergrund und Verknüpfungen
- Netzwerkeinstellungen
- Windows Systemeinstellungen (Windows Explorer, Taskleiste)
- Anwendungseinstellungen

Welche Anwendungseinstellungen jeweils migriert werden können, hängt vom eingesetzten System ab. Hier, als Beispiel, eine grobe Übersicht der Applikationen deren Einstellungen von Altiris PC Transplant übertragen werden können:

Adobe Acrobat, Adobe Photoshop, ICQ, einige Lotus Applikationen, eine Vielzahl von Microsoft Anwendungen, usw.

Die vollständige Liste kann in [41] eingesehen werden.

4.5 Sicherheitseigenschaften

Softwareverteilungssysteme verwalten eine Vielzahl von Clients in einem Unternehmen, wenn nicht gar alle. Daher ist es besonders wichtig, dass die Sicherheit dieser Systeme dadurch nicht beeinträchtigt wird. Durch den richtigen Schutz der Verwaltungssysteme kann sichergestellt werden, dass keine unbefugten Personen mit Hilfe dieser Systeme auf Clients des Unternehmens zugreifen oder diese gar außer Betrieb setzen können.

Da Softwareverteilungssysteme viele verschiedene Aufgaben haben und mit einer Vielzahl von anderen Computern verbunden sind, auf denen sie diese Aufgaben erledigen müssen, ist die Angriffsfläche für Unbefugte natürlich groß.

Demzufolge bieten so große und komplexe Systeme mehrere Angriffsflächen. Zum einen wäre da die Bedienoberfläche selbst, zum anderen könnte ein Angreifer die Netzwerkkommunikation belauschen und verändern. Es besteht auch die Möglichkeit, dass ein manipulierter Client versucht irgendwelche Rechte für andere Clients oder für das Softwareverteilungssystem selbst zu erschleichen.

Die Bedienoberfläche ist der Dreh- und Angelpunkt eines jeden Softwareverteilungssystems. Hat man erstmal uneingeschränkten Zugriff darauf, kann man über alle Clients verfügen, die dem System zugewiesen sind.

Ein Weg, die Gefahr eines Angriffs an dieser Stelle zu minimieren, ist die Trennung der Rechte für verschiedene Funktionen. Dadurch hat der Administrator die Möglichkeit, bestimmte Aufgaben auf verschiedene Personen zu verteilen. Diese haben dann jeweils nur die Berechtigung, bestimmte Funktionen in dem ihnen zugeteilten Aufgabenbereich, zu erledigen. Das Administratorkonto für das System muss nur noch zur Installation, Initialisierung und eventueller Wartung des Systems benutzt werden. Damit wird die Gefahr minimiert, dass ein Unbefugter unter Umständen an die Zugangsdaten des Administrators gelangt.

Die Verteilung und Verwaltung dieser Rechte integriert sich meist in das Microsoft Active Directory. Somit benötigen die Administratoren keine zusätzlichen Benutzerkonten, sondern können ihr Windows Benutzerkonto aus der Domäne im Unternehmen benutzen. Einige wenige Softwareverteilungssysteme, wie zum Beispiel LANdesk, unterstützen neben dem Microsoft Active Directory auch das Novell eDirectory und andere LDAP kompatible Verzeichnisse. ([42]) Sollte die IT-Infrastruktur eines Unternehmens nicht in einer Domäne realisiert sein, verwenden die Softwareverteilungssysteme das Windows Sicherheitssystem und die lokalen Benutzerkonten.

Zudem bieten manche Softwareverteilungssysteme die Möglichkeit, die Administratoren zu überwachen, somit kann der Missbrauch von Rechten verhindert werden.

Um Angriffe auf die Netzwerkkommunikation zu verhindern, gibt es mehrere Möglichkeiten. Meist wird eine asymmetrische Verschlüsselung nach dem Public-Private Key Verfahren eingesetzt. Bei der Übertragung von Installationspaketen zum Beispiel, werden diese signiert. Das Bedeutet, auf dem Server wird ein Hashwert von diesem Paket berechnet und mit dem Private-Key des Servers verschlüsselt. Dieser wird mit dem Paket an den Client geschickt. Der Client kann diesen Hashwert mit dem Public-Key entschlüsseln und vergleicht diesen Wert mit dem, den er selber vom Paket berechnet hat. Stimmen beide Werte überein, kann der Client sicher sein, dass das Paket beim Transport vom Server nicht manipuliert wurde. Dasselbe Verfahren kommt, wie in [39] beschrieben, auch bei der Übertragung von Kontonamen und Kennwörtern oder anderen sensiblen Daten zum Einsatz. Damit ist beispielsweise ein Man-In-The-Middle-Angriff ausgeschlossen.

5 Anwendungen im Detail

In diesem Teil soll ein kurzer Einblick in einige aktuelle Produkte gegeben und aufgezeigt werden, wie die behandelten Technologien dort eingesetzt werden. Zudem wird noch kurz auf die IT-Struktur der Universität Augsburg eingegangen. Da eine ausführliche Übersicht den Rahmen dieser Arbeit sprengen würde, sind die Produkte am Ende dieses Kapitels in einer tabellarischen Übersicht dargestellt. Weitere Produkte wie beispielsweise NetInstall, Prism Deploy oder ZENworks Desktop Management sind in [43] zu finden.

5.1 RIS/ADS/Longhorn Deployment

Mit den *Remote Installation Services* (RIS) liefert Microsoft mit seinen Server Produkten (Windows Server 2000 und 2003) eine kostenlose und einfache Methode, um kleinere Systemumgebungen automatisiert zu installieren. Der eigentliche Einsatzbereich ist die reine Installation von Windows-Betriebssystemen – Anwendungen können nicht selbstständig auf die Zielsysteme verteilt werden. Es ist jedoch möglich, spezielle Betriebssystem-Images zu erzeugen, in denen bereits nahezu beliebige Anwendungen (Word, Adobe Acrobat Reader, Winzip, etc.) integriert sind. Dazu muss zuerst ein System komplett installiert werden, um anschließend ein Snapshot-Image abziehen zu können. Das System lässt sich mit dem Werkzeug *riprep.exe* für den Snapshot-Vorgang vorbereiten. Dabei werden ähnlich wie mit dem Programm *sysprep.exe* alle personen- und rechnerbezogenen Daten entfernt und anschließend alle Dateien des Systems auf den Server kopiert. Da dieser Vorgang nur datei- und nicht imagebasiert abläuft und somit jede Datei einzeln kopiert wird, kann dies ziemlich lange dauern. Mit Hilfe des Programms *setupmgr.exe* kann man über eine grafische Oberfläche eine Antwortdatei für das Betriebssystemsetup erstellen. Diese kann dann für die Installation verwendet werden. Um eine Installation starten zu können, muss der RIS-Dienst in Bereitschaft versetzt werden. Der Client kann dann per PXE booten (siehe Kapitel 4.3) und den Installationsvorgang über den Server anstoßen. Da der Client hier Initiator ist, handelt es sich um eine Pull-Architektur. Im Folgenden läuft auf dem Clientrechner dann das normale Setup des Betriebssystems ab, welches die erzeugte Antwortdatei nutzen kann. Die Dateien und Einstellungen der zusätzlichen Programme werden einfach mit kopiert. Da dies wieder dateibasiert abläuft, dauert es ähnlich lange wie beim Erzeugen des Snapshots.

Automated Deployment Services (ADS)

Die *Autometed Deployment Services* [44] sind prinzipiell für die Installation von Serversystemen ausgelegt und bieten speziell für dieses Umfeld etliche Werkzeuge und Skriptmöglichkeiten an. Die Grundfunktionen wurden gegenüber RIS jedoch stark erweitert:

- 3-phasiges Deployment (Hardware Configuration Stage, Pre-OS Stage, Full-OS Stage)
- Jobs und Sequenzen für das Deployment definierbar
- WMI Interface
- Imagebasiertes Deployment
- Parallelinstallationen via Multicast

Auch wenn ADS auf Serversysteme ausgerichtet ist, lässt sich damit ein Client installieren.

Windows XP wird zwar offiziell nicht als Plattform unterstützt, funktioniert aber dennoch.

Longhorn Deployment/Windows Deployment Services (WDS)

Für die kommende Generation von Windows-Betriebssystemen hat sich Microsoft viele Gedanken zum Thema Deployment gemacht. Herausgekommen sind einige vielversprechende Ansätze und Möglichkeiten (Genaueres steht in [45] und [46]):

- Modularisierung des OS: dadurch besser anpassbar (z.B. separate Sprachpakete)
- Aufteilung in 3 Phasen: Planning, Engeneering, Implementation+Deploy
- Windows Imaging (WIM): Kompression, dateibasiertes Format, Kontainer, bootbar
- XML-basierte Version der unattended.txt, dadurch skriptfähig
- Imagebasiertes Setup (effektiver)
- spezielle Werkzeuge: Ximage, Setup Manager, User State Migration Tool, Application Compatibility Toolkit
- Verwendung von Best Practices

Dadurch soll sich die Anzahl der handzuhabenden Images reduzieren, diese besser unterstützt werden, die Installationen schneller und verlässlicher werden und die Anwendungskompatibilität soll erhalten bleiben.

Wenn Microsoft all diese Vorhaben in die Tat umsetzen kann, dann haben die Jungs in Redmond jedenfalls ein heißes Eisen in puncto Deployment im Feuer. Was es letztendlich allerdings zu leisten vermag, muss es erst noch gegen die Konkurrenz unter Beweis stellen.

5.2 Symantec Ghost/Ghost Solution Suite

Symantec Ghost ist eigentlich kein vollständiges Softwareverteilungssystem, sondern eher ein Imagingsystem zur schnellen Vorinstallation von Clients. Mit Ghost ist nur eine Verteilung von Images auf hardwarenahe Clients möglich. Eine nachträgliche Installation oder Deinstallation von Anwendungen auf dem Client ist mit diesem Produkt nicht möglich. Dazu muss ein neu angepasstes Image auf dem Client ausgerollt werden, wobei die vorhandenen Daten und Einstellungen auf dem Client überspielt werden würden. Allerdings bietet Symantec Ghost eine der schnellsten Methoden eine Vorinstallation auf einer großen Anzahl von Systemen durchzuführen.

Symantec liefert in seiner *Ghost Solution Suite* seit einiger Zeit auch ein Werkzeug (AutoInstall) aus, mit dem sich Änderungen, die etwa durch die Installation eines Programmes auf dem System gemacht werden, aufzeichnen lassen. Aus diesen Aufzeichnungen kann dann ein Softwarepaket erzeugt werden, dass dann für die Installation auf anderen (gleichen) Systemen benutzt werden kann. Auch gibt es mittlerweile Tools, um die Imagedateien zu modifizieren (Dateien hinzufügen/entfernen usw.), ein Inventory-Management und auch Reportingfunktionalitäten. Damit hat sich die starre Imaging-Lösung von einst in ein wertiges Softwareverteilungstool verwandelt.

5.3 Systems Management Server (SMS)

Microsoft gehört mit dem Systems Management Server 2003 (kurz SMS) zu den etablierten Anbietern im Bereich des Systemmanagements. Die Standardinstallation von SMS kann zusätzlich durch drei Feature Packs deutlich erweitert werden. Das Administration Feature Pack optimiert die Verwaltung des SMS 2003. Das Device Management Feature Pack dient dem Management von mobilen Geräten unter Windows CE 4.2 und Windows Mobile 2003. Das OS Deployment Feature Pack fügt Funktionen für die Verteilung von Betriebssystemen hinzu. Alle drei Feature Packs können kostenlos bei Microsoft heruntergeladen werden. Außerdem gibt es eine Kopplung zwischen dem SMS und Microsofts Patch-Management-Lösung (Software Update Services). [43]

Der Systems Management Server 2003 bietet grundlegende Funktionen eines Softwareverteilungssystems, allerdings vermisst man einige Features die bei der Konkurrenz zu finden sind, wie z.B. das Imaging. Auch werden nicht so viele Erweiterungen angeboten, die sich in die Oberfläche von SMS integrieren, wie etwa bei Altiris oder LANdesk.

5.4 Altiris CMS

Die Client Management Suite von Altiris baut auf dem Notification Server Framework auf und beinhaltet alle Module die ein modernes Softwareverteilungssystem benötigt. Zudem bietet Altiris über 50 weitere Module, die in den Notification Server integriert werden können. Eingeschlossen sind Funktionen für das Management von Servern, PDAs (Personal Digital Assistants) und weiteren IT-Assets. Ferner verweist der Hersteller gerne auf strategischen Allianzen mit Hardwareherstellern wie HP, Dell, IBM oder Fujitsu Siemens. [47]

Durch die große Anzahl an Erweiterungen die in die Altiris Client Management Suite integriert werden können, ist diese Lösung sehr flexibel und deckt eine Vielzahl von Einsatzgebieten neben der Softwareverteilung ab.

5.5 Baramundi CMS

Im Gegensatz zu Altiris ist das Augsburger Unternehmen Baramundi in die Kategorie der Anbieter von „klassischen“ Client-Management-Lösungen einzuordnen. Die Baramundi Management Suite ist eine Lösung für die Verteilung von Betriebssystemen und Software, sowie für die Inventarisierung. [43]

Die Baramundi Client Management Suite ist eine solide Lösung für kleine, mittlere und auch große Unternehmen und beschränkt sich auf die wesentlichen Funktionen eines Softwareverteilungssystems (OS-Install, Deploy, Inventory, Patch-Management, Scripting-Studio, Package-Studio).

5.6 LANDesk CMS

Die Management Suite von LANdesk umfasst alle Funktionen, wie Betriebssysteminstallation, Softwareverteilung, Asset Management und Patch Management. Ähnlich wie bei Altiris bietet LANdesk auch weitere Lösungen wie das Server- und Sicherheitsmanagement. [43]

Die grundlegenden Funktionen sind selbstverständlich in der Client Management Suite von LANdesk enthalten, darüber hinaus bietet die Lösung einige nützliche Features, wie zum Beispiel das Targeted-Multicast-Verfahren, die man bei anderen Produkten nicht vorfindet.

5.7 Fazit

Beim Vergleich der verschiedenen Produkte, wird schnell deutlich, dass es nicht das „beste" Produkt gibt. Dafür ist der gesamte Systemmanagement-Markt einfach zu umfangreich und die Bedürfnisse der Unternehmen zu verschieden. Vor der Auswahl einer Lösung muss daher die benötigte Funktionalität erörtert werden. Genügen die Basisfunktionen für Softwareverteilung, Fernsteuerung und vielleicht noch Inventarisierung, oder benötigt man definierte Prozesse für den Betrieb, spezifische Helpdesk-Lösungen und ein umfassendes, betriebswirtschaftlich getriebenes Asset Management. Erst dann kann die Produktauswahl getroffen werden. [43]

Aufgrund mangelnder Kenntnis der genauen Anforderungen und Bedürfnisse der Universität Augsburg im Bereich der IT-Infrastruktur ist es nicht einfach möglich eine gezielte Produktauswahl zu treffen. Allerdings kann hier schon mal eine grobe Vorauswahl von Softwareverteilungssystemen gemacht werden, die in das Umfeld der Universität passen würden.

Die Verwaltungs- und Netzwerkstruktur an der Universität Augsburg besteht aus mehreren Fakultäten die örtlich in manchen Fällen weit entfernt liegen, wie zum Beispiel die alte Universität in der Eichleitnerstrasse. Wünschenswert wäre ein Softwareverteilungssystem mit dem man die Fakultäten getrennt voneinander verwalten könnte, bei dem jedoch alle Daten im Rechenzentrum zusammenlaufen würden. Zudem sollte die Netzwerklast zwischen der alten und der neuen Universität niedrig gehalten werden. Das wäre am besten mit einem eigenen Server für die Softwareverteilung am Standort der alten Universität realisierbar, welcher wiederum mit dem Server im Rechenzentrum zusammen geschaltet ist.

Produktübersicht Softwareverteilungstools

Hersteller	Altiris	Baramundi	LANDesk	Microsoft	Symantec	Microsoft	Microsoft	Microsoft
Produkt	**Client Managemen Suite 6.5**	**Management Suite 6.3**	**Management Suite 8**	**Systems Management Server 2003**	**Ghost Solution Suite**	RIS	ADS	Longhorn Deployment
Info	www.altiris.de	www.baramundi.de	www.landesk.de	www.microsoft.de	www.symantec.de	www.microsoft.de	www.microsoft.de	www.microsoft.de
Unterstützte Clients	Windows 9x/ME/NT 4.0 und höher	Windows NT 4.0, 2000, XP	Windows 9x/ME/NT 4.0 und höher, Mac und andere	Windows 9x/ME/NT 4.0 und höher	Windows 9x/ME/NT 4.0 und höher	Windows 2000, XP Prof., 2003 Server	offiziell nur Server Produkte ab Windows 2000	Windows Vista, XP(?), 2003(?)
Unterstützte Server	Windows 2000 und höher	Windows NT 4.0 und höher	Windows NT 4.0 und höher	Windows 2000 und höher		Windows Server ab 2000	Windows Server ab 2003	Windows Vista Server
Datenbank	MSDE oder SQL Server	MSDE oder SQL Server	MSDE, SQL Server, Oracle	SQL Server		Active Directory	MSDE	MSDE
Funktionen								
Softwareverteilung	✓	✓	✓	✓	✓ (eingeschränkt)	✓ (eingeschränkt)	✓	✓
Betriebssysteminstallation	✓	✓	✓	✓ (Feature Pack)	✓ (Sysprep)	✓	✓	✓
Imaging	✓ (RapiDeploy)	× (nur Schnittstelle vorhanden)	✓	✓ (Feature Pack)	✓	×	✓	✓
Snapshot	✓	×	✓	×	✓	✓	✓	✓
MSI-Unterstützung	✓	✓	✓	✓	×	✓	✓	✓
Inventarisierung	✓	✓	✓	✓	✓	×	×	✓
Lizenz-Management	✓	×	✓	×	×	×	×	k.A.
Patch Management	✓	✓	✓	✓ (eingeschränkt)	✓ (eingeschränkt)	×	✓ (eingeschränkt)	✓
Hierarchiebildung	✓	✓	✓	✓	×	×	×	k.A.
Eigener Packager	✓	✓ (Add-On)	✓	×	✓	×	×	✓
Reporting	✓	✓	✓	✓	✓	×	✓	✓
Migration	✓	✓ (Add-On)	✓	✓ (eingeschränkt, Feature Pack)	✓	×	×	✓
Hardware Konfiguration	DOS Tools, WinPE (ab 6.5)	k.A.	k.A.	WinPE (Feature Pack)	DOS Tools	WinPE Tools	Herstellerspezifische DOS Tools	WinPE (Vista)
Netzwerk								
Deployment-Methode	Imagebasiert, Native Installation, Unattended Setup	Native Installation, Unattended Setup	Imagebasiert, Native Installation, Unattended Setup	Imagebasiert, Native Installation, Unattended Setup	Imagebasiert	Dateibasiert, skriptbasiert	Imagebasiert	Imagebasiert
Unicast/Multicast	✓/✓	✓/ k.A.	✓/✓(Targeted Multicast™)	✓/×	✓/✓/ Directed Broadcast	✓/×	✓/✓	✓/✓
Deployment-Initialisierung	Server (Push)	Server (Push)	Server (Push)	Server (Push)	Server (Push)	Client (Pull)	Server (Push)	k.A.
PXE	✓	✓	✓	k.A.	✓	✓	✓	✓
WOL	✓	✓	✓	✓ (Add-On)	k.A.	×	k.A.	k.A.
gleichzeitige Deployments	k.A.	k.A.	k.A.	k.A.	k.A.	75	128	k.A.
Sonstiges								
Unterstützte Dateisysteme	NTFS, FAT32, FAT16	NTFS, FAT32, FAT16	NTFS, FAT32, FAT16	NTFS, FAT32, FAT16	Rohdaten	NTFS	NTFS, FAT32, FAT16	NTFS, FAT32, FAT16
GUI	Webkonsole, Win32 (Deployment Server)	MMC-Anwendung	Win32-Anwendung + Webkonsole	MMC-Anwendung	Win32 Anwendung	MMC-Anwendung	MMC-Anwendung	Win32 Anwendung
Bemerkungen								
Besonderheiten	Benötigt IIS mit ASP.NET 1.1	Kompakte Management Lösung, die sich auf das Wesentliche beschränkt	Multicast bei der Softwareverteilung	Kostenlose Feature Packs bei Microsoft downloadbar		für Clientsysteme konzipiert ab Windows 2000 verfügbar	für Server konzipiert, ab Windows 2003 verfügbar, WMI,	Application Compatibility Toolkit, Test Tools, Image-Management,
Vorteile	Notification Server ist eine Web-Applikation, kann von jedem Client mit Internet Explorer verwaltet werden, über 50 weitere Solutions für den Notification Server	Übersichtlich und skalierbar durch modularen Aufbau	Andere Imaging Tools können eingebunden werden, Management Suite durch zahlreiche andere Module erweiterbar	Viele Third Party AddOns verfügbar	Schnelles OS Deployment	kostenloser Zusatz zum Server-OS, schlichtes, schnell Handzuhaben	kostenloser Zusatz zum Server-OS, Powerful Mass Server Administration	kostenloser Zusatz zum Server-OS, weniger Images, bessere Unterstützung der Images, schnellere und verlässlichere Installationen, Erhaltung der Anwendungskompatibilität
Nachteile	Sehr komplexe Lösung	Kein Imaging, Mangel an weiteren Funktionen und Modulen		Kein Imaging, kein Lizenz-Management	Keine Hierarchiebildung	unkomfortables Management, keine echte Softwareverteilung	unkomfortables Management, eingeschränkter Funktionsumfang, hoher Einarbeitungsaufwand, nur Server Systeme ab Windows 2000	nur Visita OS unterstützt

***Tabelle 2:** Produktübersicht Softwareverteilungstools*

6 Ausblick/Conclusion

Betrachtet man die Lösungen zur Softwareverteilung auf dem heutigen Markt, stellt man fest, dass der Funktionsumfang der Produkte oft weit darüber hinaus geht. Funktionen wie Inventarisierung, Patch-Management oder Remote Control lassen sich in den meisten der hier betrachteten Produkte finden. Ebenso ist ein Hang zur Modularisierung der einzelnen Funktionalitäten auszumachen. Dadurch kann ein Unternehmen sich die, in der Unternehmensstruktur benötigten Pakete, aussuchen und damit die Kosten niedrig halten.

Auch bezüglich der Client-Typen haben sich die Werkzeuge von den reinen Intel-Desktops mit Windows in Richtung Linux weiterentwickelt.

Auf jeden Fall ist ein Trend zur Erweiterung der Funktionalität erkennbar. Vor allem das Patch-Management gewinnt in der jetzigen Zeit stark an Bedeutung. Es ist heutzutage fast unmöglich alle Patches oder Updates von Betriebssystemen oder sonstigen Anwendungen, ohne die Hilfe eines Patch-Management Systems auf die Unternehmensumgebung zu verteilen.

Alle der hier vorgestellten Techniken kommen in den Softwareverteilungssystemen zum Einsatz. Oft werden auch mehrere Technologien miteinander kombiniert, um dadurch die Schwächen einer Technologie durch die Stärken einer anderen zu ergänzen. Einige andere Technologien, wie das Multicasting werden zur Zeit leider nur in einigen Produkten eingesetzt.

Die Produktpalette auf dem Markt ist so umfangreich wie in kaum einem anderen Gebiet der IT-Branche. Der Trend von Eigenentwicklungen von Softwareverteilungssystemen ist in den letzten Jahren stark zurückgegangen [35]. Das ist als sehr positiv zu verzeichnen, denn die Unternehmen haben erkannt, dass es bereits solide Lösungen gibt, deren Kaufkosten sich langfristig amortisieren.

Für die Hersteller der Deploymentlösungen gilt, eine taugliche und einheitliche Umsetzung aller Modelle (producer, enterprise, deployment und site models) und vor allem des Predispose-Prozesses anzustreben. Standards wie das Open Software Description Format und das Common Information Model auf Basis von XML sind hier auf jeden Fall ein Schritt in die richtige Richtung.

Literaturverzeichnis

[1] it innovations GmbH: "Softwareverteilung", 2005, http://www.it-innovations.de/ks/Softwareverteilung/

[2] deron Systemhaus: "Marktstudie Client Management 2002 - Auszug", 2002, deron Systemhaus GmbH, http://www.deron.de

[3] Thierry Coupaye & Jacky Estublier: "Foundations of Enterprise Software Deployment", 2000, DASSAULT SYSTEMS & IMAG LSR Joint Laboratory, http://www.imag.fr

[4] RSP&A: "Software Engineering Resources - Software Configuration Management", 2004, http://www.rspa.com/spi/SCM.html

[5] R. P. Herrold: "RPM Package Manager", 2004, www.rpm.org

[6] Debian FAQ Authors: "The Debian GNU/Linux FAQ - Basics of the Debian package management system", 2005, www.debian.org/doc/FAQ/ch-pkg_basics

[7] Microsoft: "Windows Installer", 2005, www.microsoft.com

[8] Macrovision: "InstallShield - InstallFromTheWeb Client", 2005, www.installshield.com/client/

[9] Microsoft: "Systems Management Server", 2005, www.microsoft.com/smserver/

[10] Altiris: "Client Managemnet Suite", 2005, www.altiris.com/products/clientmgmt/

[11] Novell: "ZENworks Suite", 2005, www.novell.com/products/zenworks/

[12] Arthur van Hoff, Hadi Partovi, Tom Thai: "The Open Software Description Format (OSD)", 1997, www.w3c.org/TR/NOTE-OSD

[13] Richard Hall, Dennis Heimbinger, Alexander Wolf: "Specifying the Deployable Software Description Format in XML", 1999, http://serl.cs.colorado.edu/

[14] DMTF: "Common Information Model (CIM) Standards", 2005, http://www.dmtf.org/standards/cim/

[15] Microsoft: "Managing Windows with WMI", 1999, http://msdn.microsoft.com/library/default.asp?url=

[16] Microsoft: "Geheimnisse von Windows Management Instrumentation", 2004, http://www.microsoft.com/germany/technet/datenbank

[17] DMTF: "Web-Based Enterprise Management (WBEM)", 2005, http://www.dmtf.org/standards/wbem/

[18] Microsoft TechNet: "Management Architecture Guide: Version 2.0", 2005, http://www.microsoft.com/technet/itsolutions/cits/

[19] deron Systemhaus: "Technologien der Softwareverteilung - Auszug", 2003, deron Systemhaus GmbH, http://www.deron.de

[20] Tony Bradley: "Patch Management Tips - When to push or pull patches", 2005, searchwindowssecurity.teachtarget.com

[21] John Hagel & John Seely Brown: "From Push to Pull - Emerging Models for Mobilizing Resources", 2005, www.edgeperspectives.com

[22] Robert Hinden: "IP Next Generation Overview", 1995, http://playground.sun.com/pub/ipng/html/INET-IPng-

[23] IETF: "IP Version 6 (IPv6)", 2005, http://playground.sun.com/pub/ipng/html/ipng-main.

[24] Microsoft Technet: "Scenarios and Procedures for Microsoft Systems Management Server 2003: Plan", 2004, http://www.microsoft.com/technet/prodtechnol/sms/s

[25] LANDesk Software: "LANDesk Management Suite 8 - Installation and Deployment & User Guide", 2004, http://www.landesk.de/

[26] Macrovision Corporation: "InstallShield Windows", 2006, http://www.macrovision.com/products/flexnet_installshield/

[27] baramundi software AG: "baramundi Management Suite", 2005, http://www.baramundi.de/

[28] New Boundary Technologies: "Prism Suite", 2005, http://www.newboundary.com/products/prismdeploy/

[29] Symantec: "Symantec Website", 2006, http://www.symantec.com

[30] Jonathan Bennett: "AutoIt Script", 2005, http://www.autoitscript.com

[31] Autohotkey: "AutoHotkey", 2005, http://www.autohotkey.com/

[32] Symantec Corporation: "Symantec Ghost Reference Guide (v8.0)", 2003, www.symantec.com

[33] Altiris: "Altiris Deployment Solution 6.5 - Reference Guide", 2005, http://www.altiris.com/upload/deployment.pdf

[34] Mark Russinovich, Bryce Cogswell: "NewSID", 2005, http://www.sysinternals.com/Utilities/NewSid.html

[35] deron Systemhaus: "Client Management 2004 - Auszug", 2004, deron Systemhaus GmbH, http://www.deron.de

[36] Frank Bartel: "Patch-Management statt Patchwork", LANline 20/2003, Konradin Verlagsgruppe AG, http://www.netigator.de/netigator/live/fachartikelarchiv/index.php3?obj=LL&np=archiv

[37] Microsoft: "Microsoft Baseline Security Analyzer (MBSA)", 2006, http://www.microsoft.com/technet/security/tools/mbsahome.mspx

[38] Altiris: "Client Managemnet Suite", 2005, http://www.altiris.com/products/clientmgmt/

[39] Microsoft: "Microsoft Systems Management Server 2003 - Konzepte-, Planungs- und Bereitstellungshandbuch", 2004, http://www.microsoft.com/downloads/details.aspx?FamilyID=bd2b3619-4704-4c19-a00b-628e65f6f826&DisplayLang=de

[40] Intel & SystemSoft: "Preboot Execution Environment (PXE) Specification 2.1", 1999, http://www.pix.net/software/pxeboot/archive/pxespec.pdf

[41] Altiris: "Altiris PC Transplant - list applications supported", 2005, http://www.altiris.com/docs/products/PCTapps.pdf

[42] LANDesk Software: "LANDesk White Paper: LANDesk Management Suite 8 vs. SMS 2003", 2004, www.landesk.com

[43] Martin Kuppinger: "Einzelne Tools reichen nicht mehr aus", Infoweek.ch 2/2005, Compress Information Group AG, www.infoweek.ch

[44] Microsoft: "Choosing Between ADS and RIS for Bare-Metal Deployments and Re-Deployments", 2003, http://www.microsoft.com/windowsserver2003/technologies/management/ads/default.mspx

[45] Paul Thurrott: "Longhorn Setup and Deployment Strategies", 2004, http://www.winsupersite.com/showcase/longhorn_setup.asp

[46] Tim Sinclair, Manu Namboodiri: "Windows - Codename "Longhorn" Deployment: Tools, Technologies and Best Practices", 2005, http://www.microsoft.com

[47] Johann Baumeister: "Funktionsstand im Client-Management", LANline 9/2005, Konradin Verlagsgruppe AG, http://www.netigator.de/netigator/live/fachartikelarchiv/index.php3?obj=LL&np=archiv

Abbildungsverzeichnis

Tabellenverzeichnis

Autorenverzeichnis

Die folgenden Kapitel wurden von Florian Mücke geschrieben:

- Kapitel 1 – komplett (mit Abstract)
- Kapitel 2 – komplett
- Kapitel 3 – komplett (bis auf 3.3 – 3.6)
- Kapitel 4 – nur 4.3 (Hardwareunterstützung)
- Kapitel 5 – nur 5.1 (RIS/ADS/LD) + Tabelle, 5.2 (Ghost Solution Suite)

Die folgenden Kapitel wurden von Adalbert Ochotta geschrieben:

- Kapitel 3 – nur 3.3 (Native Installation), 3.4 (Snapshot), 3.5 (MSI), 3.6 (Imaging/Cloning)
- Kapitel 4 – komplett (bis auf 4.3)
- Kapitel 5 – komplett (bis auf 5.1)
- Kapitel 6 – komplett

Erklärung

Wir erklären hiermit an Eides Statt,

- dass wir die vorliegende Studienarbeit selbstständig angefertigt,
- keine anderen als die angegebenen Quellen benutzt,
- die wörtlich oder dem Inhalt nach aus fremden Arbeiten entnommenen Stellen, bildlichen Darstellungen und dergleichen als solche genau kenntlich gemacht und
- keine unerlaubte fremde Hilfe in Anspruch genommen habe.

Königsbrunn, 25. Januar 2006

Florian Mücke, Adalbert Ochotta