PEARSON Education

ADDISON-WESLEY

Stephanie Knecht-Thurmann
Thomas Knecht

Windows Installer

Windows-Software installieren und verwalten

ADDISON-WESLEY
An imprint of Pearson Education

München • Boston • San Francisco • Harlow, England
Don Mills, Ontario • Sydney • Mexico City
Madrid • Amsterdam

3 Architektur und Inhalt eines Windows Installer-Pakets

Nachdem Sie in den beiden letzten Kapiteln einen kurzen Überblick über den Windows Installer und dessen Funktionsfeatures bekommen haben, wird Ihnen nun die *.msi*-Architektur näher vorgestellt. Danach lernen Sie die Unterschiede zwischen einer herkömmlichen Setup-Routine und der Installation eines *.msi*-Pakets sowie die daraus resultierenden Vorteile kennen. Darauf aufbauend erhalten Sie einen Einblick in den Aufbau von *.msi*-Paketen sowie die verschiedenen vom Windows Installer unterstützten Dateitypen.

3.1 Einführung in die Windows Installer-Architektur

Die Windows Installer Technologie basiert auf drei Komponenten: dem *Windows Installer-Client*, dem *Windows Installer-Dienst* sowie dem *Windows Installer-Paket* (*.msi*-Paket). Diese drei Bestandteile werden im Folgenden näher vorgestellt.

3.1.1 Der Windows Installer-Client

Der *Windows Installer Client* ist jede beliebige Applikation, die den Windows Installer zur Durchführung einer Aufgabe aufruft. Dieser Aufruf kann beispielsweise über eine Installer-basierte Applikation, über den Benutzeraufruf von SYSTEMSTEUERUNG/SOFTWARE/PROGRAMME ENTFERNEN bzw. /NEUE PROGRAMME HINZUFÜGEN oder auch über Softwareverteilungsmechanismen wie die des *Systems Management Server (SMS)* oder die Gruppenrichtlinien im Active Directory angestoßen werden.

Für diese Interaktionen zeigt der *Windows Installer Client* die Benutzeroberfläche für Installation oder Deinstallation an. Der *Windows Installer Client* benutzt den Windows Installer-Dienst jedes Mal, wenn Änderungen an der Konfiguration vorgenommen werden, d.h. wenn neue Daten geschrieben bzw. Daten entfernt werden oder Änderungen an der Registry erfolgen.

3.1.2 Der Windows Installer-Dienst

Der Windows Installer-Dienst wird unter Windows NT 4.0, 2000, XP und 2003 als Systemdienst ausgeführt und ist Bestandteil dieser Betriebssysteme bzw. kann als aktuellere Version nachinstalliert werden. Dieser Dienst greift auf die Inhalte des Installer-Pakets zu, um die Installation der Applikation zu verwalten. Dazu zählen das Kopieren der Dateien, Ändern der Registry, Erstellen von Verknüpfungen und das Anzeigen von Dialogboxen während der Installation. Beispielsweise sind im Installer-Paket verschiedene

Installationsanweisungen für den Installer-Dienst hinterlegt, je nachdem, ob bereits eine ältere Version der Applikation auf dem System vorhanden ist oder ob die Erstinstallation der Applikation erfolgt. Weitere Hinweise zum Windows Installer-Dienst finden Sie in Kapitel 4.1.

3.1.3 Das Windows Installer-Paket

Jedes Windows Installer-Paket, auch als *.msi*-Paket bezeichnet, besteht aus einer Datenbank. In dieser Datenbank befinden sich in Form zahlreicher Tabellen sämtliche Anweisungen für den Windows Installer-Dienst sowie sämtliche Daten, die für die Verwaltung des Programmstatus (Installation, Änderungen, Deinstallation) erforderlich sind. Mit Hilfe verschiedener Authoringtools können eigene *.msi*-Pakete erstellt und modifiziert werden.

3.2 Inhalt eines .msi-Pakets

In jedem Windows Installer-Paket sind die kompletten Informationen enthalten, die für die Installation und Deinstallation der Applikation sowie die Setup-GUI erforderlich sind. Das Installationspaket besteht aus einer *.msi*-Datei. Diese *.msi*-Datei ist eine relationale Datenbank mit diversen, untereinander verknüpften Tabellen. In ihr sind außerdem die internen und externen Quelldateien, Cabinet Files sowie optional Transform Files enthalten.

Tabelle 3.1 zeigt die Tabellengruppen, die in jedem *.msi*-Installations-Paket vorhanden sind. Die Inhalte der einzelnen Tabellen werden in Kapitel 8 ausführlich erläutert.

Tabellengruppe	Beschreibung
Core Table Group	Beschreibt die verfügbaren Features und Komponenten einer Applikation
File Table Group	Enthält die mit dem Installationspaket verbundenen Dateien
Registry Table Group	Enthält die Registryeinträge
System Table Group	Verfolgt die Tabellen und Spalten der Installationsdatenbank
Locator Table Group	Wird zum Durchsuchen der Registry, der Ordnerstruktur oder Installer-Konfigurationsdaten benutzt
Program Installation Group	Enthält die Eigenschaften, Verknüpfungen, Bitmaps usw., also alle Elemente, die für die Installation benötigt werden
Installation Procedure Group	Verwaltet die während der Installation von Standard und Custom Actions durchgeführten Vorgänge

Tabelle 3.1: Die Tabellengruppen in einem .msi-Paket

3.2.1 Konzept des Windows Installer

Für die Installation benutzt der Windows Installer ein Konzept, das auf Komponenten und Features basiert. Features sind einzelne Bestandteile der Gesamtinstallation, die der Benutzer auswählen kann. Beispielsweise besteht das Microsoft Office-Paket aus einer Reihe von Features wie z.B. *Word*, *Excel* oder auch Unterfeatures wie der Rechtschreibprüfung für *Word* oder *Excel*. Alle Features können einzeln vom Benutzer zur Installation ausgewählt werden.

Jedes dieser Features besteht seinerseits aus verschiedenen Komponenten. Komponenten werden beispielsweise aus Ressourcen wie einzelnen Dateien (z.B. *winword.exe*), Registry-Schlüsseln, Verknüpfungen oder CLSIDs zusammengefügt. Eine Komponente kann vom Benutzer nicht separat ausgewählt werden. Installiert oder deinstalliert der Benutzer ein Feature, so werden alle Bestandteile der dazugehörigen Komponente gemeinsam hinzugefügt bzw. entfernt. Features können auch Komponenten gemeinsam benutzen. So verwenden beispielsweise die Rechtschreibprüfung in *Word* und *Excel* dieselben Komponenten wie Registry-Schlüssel oder CLSIDs.

Bei einer Installation, die nicht auf dem Windows Installer beruht, sieht das Installationskonzept komplett anders aus. Jede Applikation verfügt dabei über eine eigene ausführbare Datei oder ein Skript zur Ausführung der Installation. Für jede einzelne Applikation muss eigenen Installationsregeln, vor allem im Bereich der Versionierung, gefolgt werden. Es gibt keine zentrale Instanz, in der die Installationsregeln definiert waren. Das größte Problem ist jedoch der Umgang mit *dlls*. Kaum ein Installationsverfahren außer dem Windows Installer verwendet eine Nummerierung (Reference Counts) gemeinsam genutzter *.dll*-Dateien. Bei der Installation werden stattdessen durchweg vorhandene neuere *.dll*-Dateien durch ältere Versionen überschrieben, bei der Deinstallation werden gemeinsam genutzte Dateien entfernt, so dass zuweilen bestimmte Applikationen nach der Installation bzw. Deinstallation eines anderen Programms nicht mehr funktionieren, da eine *.dll*-Datei nicht mehr vorhanden ist.

Im Gegensatz zum nicht-Installer-basierten Installationsverfahren betrachtet der Windows Installer Dienst jede Applikation als aus drei logischen Teilen bestehend. Hierbei handelt es sich um Komponenten (Components), Produkte (Products) sowie Features.

3.2.2 Komponenten

Die Komponente ist zwar die kleinste, aber grundlegende Einheit der drei Bestandteile. Sie umfasst Dateien, Registry-Schlüssel und weitere Ressourcen, die zusammen installiert bzw. deinstalliert werden sollen. Wird eine Komponente zur Installation ausgewählt, werden sämtliche in ihr enthaltenen Ressourcen installiert. Eine Ressource kann immer nur Bestandteil einer Komponente sein. So ist es nicht möglich, dass zwei Komponenten eine Datei gemeinsam benutzen. Dabei ist es unerheblich, ob die Komponenten Bestandteil eines Produkts oder verschiedener Produkte sind. Jedes Produkt muss also selbst die Datei mit ausliefern. Jede Komponente besitzt somit quasi ihre eigenen Ressourcen. Die Komponenten sind für den Benutzer nicht sichtbar, lediglich der Entwickler weiß, aus welchen Komponenten sich seine Applikation zusammensetzt.

Innerhalb einer Komponente werden die Ressourcen weiter unterteilt. Es gibt Schlüsselpfade (Keypaths) und Komponentencode (Component Code). Als Keypath wird eine einzige Ressource innerhalb der Komponente bestimmt. In der Regel wird eine Datei gewählt, es kann jedoch auch ein Registrywert ausgesucht werden. Der Keypath gibt den Pfad zu einer gegebenen Komponente an. Wenn also eine Applikation den Pfad zu einer Komponente benötigt, so wird über den Windows Installer-Dienst der Pfad der Keypath-Ressource zurückgegeben. Die Keypath-Komponente spielt auch bei der Überprüfung einer Applikation durch den Installer eine wichtige Rolle. Wird diese Komponente nicht gefunden, wird die gesamte Komponente als beschädigt angesehen und repariert. Indem zwei oder mehr Ressourcen in eine Komponente aufgenommen werden, kann sichergestellt werden, dass eine dieser Ressourcen nie ohne die andere installiert oder deinstalliert werden kann.

Weiterhin muss garantiert sein, dass eine Komponente eindeutig ist. Eine bestimmte Komponente muss immer – unabhängig von der Applikation, zu der sie gehört – denselben Satz von Ressourcen beinhalten. Deshalb wird jeder Komponente eine GUID (Globally Unique Identifier) zugewiesen. Diese GUID wird auch als Component Code bezeichnet.

Aufgrund dieser Strukturierung ist es möglich, dass der Installer-Dienst die Applikationen auf dem Level der Komponenten verwalten kann, während bei anderen Installationsmechanismen die Dateien usw. direkt verwaltet werden.

Für die Verwaltung gemeinsam genutzter Dateien wird von herkömmlichen Installationsmechanismen – wenn überhaupt – eine Zählung gemeinsam genutzter Referenzen (shared reference count oder refcount) in der Registry durchgeführt. Diese Zählung bezieht sich jedoch nur auf gemeinsam verwendete Dateien, nicht aber auf andere Ressourcen wie z.B. Registry-Schlüssel. Der Windows Installer hingegen führt die Zählung der gemeinsam genutzten Referenzen auf Basis der Komponenten durch. Somit kann die Zählung für alle Ressourcen durchgeführt werden, und es ist sichergestellt, dass keine Ressource entfernt wird, bevor nicht die letzte Applikation deinstalliert ist, die auf diese gemeinsam genutzte Ressource zugreift. Die refcounts werden in Form von Client-Listen von Produktcode verwaltet. Der Windows Installer ist so in der Lage, alle Clients dieser Ressource zu erkennen und die Zählungen zu synchronisieren.

Die mangelhafte Zählung von Ressourcen, die wie bereits erwähnt nur auf Dateien, herkömmlicher Installationsverfahren beschränkt ist sowie das Nicht-Vorhandensein von Komponenten ergeben, dass bei der Deinstallation Reste der Applikation auf dem Computer verbleiben.

Der Windows Installer kann genau nachverfolgen, was genau eine bestimmte Komponente installiert hat und wann diese Komponente vollständig entfernt werden kann.

3.2.3 Produkt

Ein Produkt ist rein technisch gesehen die Zusammenstellung aller Features, die eine komplette Applikation oder besser gesagt ein komplettes Produkt ausmachen. So sind beispielsweise Microsoft Word oder Microsoft Excel bestimmte Produkte. Alle Features eines Produkts werden in einem Installer-Paket, d.h. einer *.msi*-Datei zusammengefasst.

Jedes Produkt verfügt über einen eindeutigen Produktcode. Über diesen Code werden die Clients vom Windows Installer-Dienst identifiziert, auf denen ein bestimmtes Produkt installiert ist. Zudem werden über den Produktcode vom Installer-Dienst auch die Applikationen ermittelt, die Clients bestimmter Komponenten sind. Dazu wird vom Windows Installer eine Liste verwaltet, die für jede Komponente über eine Liste von Client-Produkten verfügt.

3.2.4 Feature

Die Features sind – im Gegensatz zu den Komponenten – die Teile einer Applikation, die der Benutzer sehen kann. Bei einer benutzerdefinierten Installation kann der Benutzer auswählen, welche Features der Applikation in welcher Weise installiert werden sollen. Jedes auswählbare Programmelement entspricht dabei einem Feature. Wird ein Feature ausgewählt, werden automatisch sämtliche Komponenten dieses Features installiert. Im Gegensatz zu herkömmlichen Installationen, bei der lediglich entschieden werden kann, ob ein bestimmtes Feature installiert oder nicht installiert werden soll, bietet die Windows Installer-Technologie ein weit gefächertes Spektrum von Installationsoptionen wie z.B. „auf dem Arbeitsplatz installieren“, „bei der ersten Verwendung installieren“ oder „nicht verfügbar“. Die einzelnen Features spiegeln die verschiedenen Funktionen einer Applikation wider. Technisch gesehen setzt sich ein Feature aus einer Gruppe von Komponenten zusammen. Ein Feature kann auch weitere Features beinhalten. Man spricht in diesem Fall von Sub-Features. Features verfügen im Gegensatz zu Komponenten auch nicht über GUIDs, da sie nicht eindeutig sein müssen. Da die Verwaltung auf der Ebene der Komponenten durchgeführt wird, muss ein Feature nicht über den exklusiven Besitz einer Komponente verfügen.

Das folgende Schaubild (siehe Abbildung 3.1) veranschaulicht das Zusammenspiel von Produkten, Komponenten und Features innerhalb eines *.msi*-Pakets.

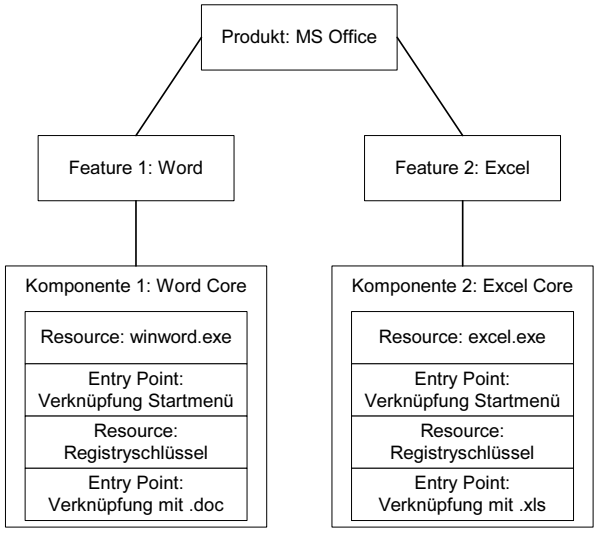

Abbildung 3.1: Das Zusammenspiel zwischen Produkt, Feature und Komponente

Das *.msi*-Paket befindet sich grundsätzlich im Root-Verzeichnis des Installationsmediums. Die Produktdateien werden entweder in einer internen *.cab*-Datei (Cabinet-Datei) oder separat zum *.msi*-Paket mitgeliefert (siehe Abbildung 3.3).

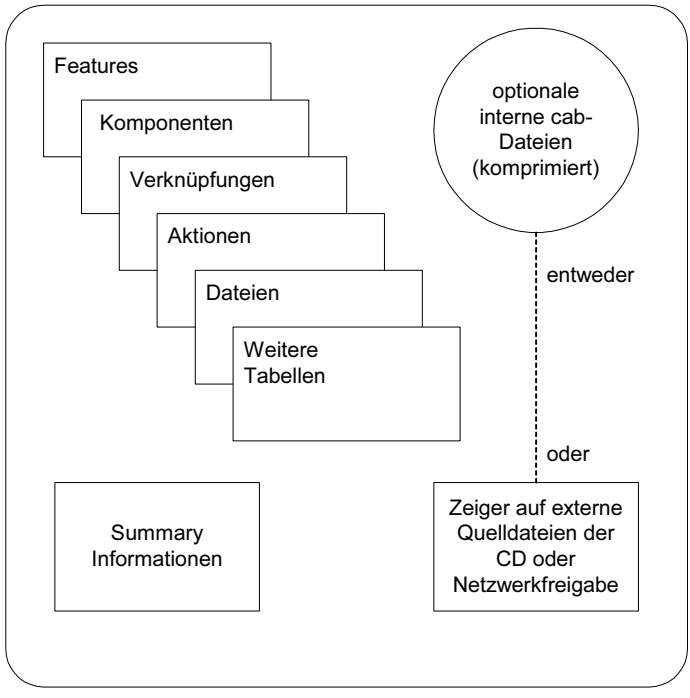

Abbildung 3.2: Aufbau und Inhalt eines Windows Installer-Pakets

Die Datenbankstruktur des *.msi*-Pakets spiegelt das Beziehungsgeflecht zwischen den Komponenten, Features und Ressourcen eines Produkts wider. Zudem ist die Struktur für Installationsvorgänge optimiert. Es wird dabei das *.msi*-Paket geöffnet, und die Informationen darin werden für die Installation ausgelesen. Die Windows Installer-basierte Installation von Microsoft Office 2000 und höher verwendet keine *.inf-* (Information/ Setup-File-), *.lst-* (Listing File-) und *.stf-* (Setup Information File-) Dateien mehr.

3.2.5 API

Zur Verwaltung der Produkte stellt der Windows Installer-Dienst eine eigene API bereit. Über diese API ist es möglich, Installer-Produkte und –Features zu installieren, konfigurieren und deinstallieren, die installierten Produkte, Features und Komponenten aufzuzählen sowie den Pfad zu installierten Komponenten zu bestimmen. Dabei ist der letzte Punkt der wichtigste. Programme können zur Laufzeit vom Windows Installer-Dienst Informationen über einen Pfad einer bestimmten Komponente anfordern. Damit besteht keine Notwendigkeit für hart verdrahtete, statische Pfadangaben, bei denen die Proble-

matik besteht, dass sie entweder auf fehlende Dateien zeigen oder auf verschiedenen Computern voneinander abweichen können.

3.2.6 Installation bei Bedarf von Features

Sobald bei einer Applikation, die noch nicht die Windows Installer-Technologie verwendet, Features benötigt werden, die bei der ersten Installation nicht gewählt wurden, so erfordert dies grundsätzlich das Beenden der Applikation und einen neuen Aufruf des Setups. Ein Benutzer müsste also bereits vor dem erstmaligen Ausführen der Applikation entscheiden, welche Features davon er jemals benötigen wird und welche nicht. Der erneute Aufruf des Setups ist bei Windows Installer-basierten Applikationen nicht notwendig, wenn später weitere Features nachinstalliert werden sollen. Wird ein Feature nachträglich benötigt, so ruft die Applikation den Windows Installer-Dienst auf, über den das neue Feature installiert wird. Der Benutzer muss also nicht erst die Applikation beenden und manuell das Setup erneut aufrufen. Es stehen also zu jeder Zeit quasi alle Features bereit – es sei denn, ein Feature wird bei der Installation explizit als nicht verfügbar markiert. Man spricht hierbei auch von Advertisement, also dem Ankündigen von Features.

Angenommen, bei der Installation von *Word 2003* wird nicht die Funktion Silbentrennung mitinstalliert. Die Silbentrennung stellt ein Feature des Produkts *Word* dar. Wird diese später benötigt, erfolgt eine Nachinstallation. Hierbei kann der Benutzer wahlweise zur Bestätigung der Installation aufgefordert werden oder nicht.

3.2.7 Installation bei Bedarf von Produkten

Im Gegensatz zur Installation bei Bedarf einzelner Features ist bei der Produktinstallation bei Bedarf die Unterstützung durch das Betriebssystem erforderlich. Da der Windows Installer erst ab Windows 2000 in das Betriebssystem integriert ist, ist unter Windows 95, 98 und NT deshalb diese Funktion nicht verfügbar. Bei der Feature-Installation wird die Windows Installer Management API verwendet, die Produkt-Installation setzt jedoch voraus, dass das Betriebssystem dieselbe *Windows Installer Management API* verwendet, um angekündigte Applikationen zu installieren. Ab Windows 2000 benutzen die Windows Shell sowie OLE (Object Linking and Embedding) die *Windows Installer Management API*.

Da ja eine Applikation, die nur angekündigt ist, noch nicht installiert ist, kann sie sich nicht selbst für die Installation verwenden. Bei der nachträglichen Installation von Features hingegen ist die Applikation an sich bereits vorhanden und kann über die entsprechende API die Installation des Features vornehmen. Beim Ankündigen von Applikationen werden lediglich einige wenige Bestandteile der Applikation installiert. Dazu zählen Verknüpfungen im Startmenü und auf dem Desktop, die Verbindung mit der Dateierweiterung sowie die OLE-Registrierung. Sobald die angekündigte Applikation beispielsweise über die Verknüpfung im Startmenü aufgerufen wird, wird vom Betriebssystem der Windows Installer-Dienst aufgerufen, um die Applikation zu installieren. Nach Ende der Installation wird dem Betriebssystem der Pfad der neu installierten Applikation mitgeteilt. Für die Features der Verknüpfungserstellung und Dateierweiterung ist die *Inter-*

net Explorer 4.01 SP1-Shell mit installiertem (jedoch nicht unbedingt aktiviertem) Active Desktop oder höher erforderlich.

Das Feature der Produktinstallation bei Bedarf ist ein wichtiger Bestandteil für die Intelli-Mirror-Technologie ab Windows 2000. Mit Hilfe dieser Technologie ist es möglich, beispielsweise in einem Active Directory Applikationen zuzuweisen oder anzukündigen. Im ersten Fall wird die Applikation bei einem Benutzer installiert. Bei der Ankündigung kann der Benutzer selbst entscheiden, welche bereitgestellten Applikationen er installieren möchte. In beiden Fällen ist eine Installation ohne lokalen Eingriff eines Administrators möglich.

3.2.8 Ressourcenstabilität zur Laufzeit

Außer der Installation bei Bedarf ist über die Windows Installer Management API auch die Reparatur von Applikationen zur Laufzeit möglich. Sobald eine installierte Applikation den Installer-Dienst aufruft, um einen Pfad aufzulösen, werden zwei verschiedene Prüfungen durchgeführt. Zunächst wird überprüft, ob die angeforderte Komponente und das Feature installiert sind. Ist dies nicht der Fall, wird das entsprechende Feature installiert bzw. bei einem angekündigten Produkt die Applikation installiert. Als zweites wird geprüft, ob sämtliche Komponenten des Features korrekt installiert sind. Hierbei wird der Keypath einer Komponente überprüft und dabei analysiert, ob die Komponente fehlerhaft ist oder nicht. Fehlt die Keypath-Ressource, wird automatisch zur Laufzeit eine Selbstreparatur durchgeführt. Hierzu muss lediglich Zugriff auf die Installationsquelle bestehen.

3.2.9 Rollback

Treten während der Installation schwer wiegende Fehler auf, so kann eine Windows Installer-basierte Installation vollständig rückgängig gemacht werden. Der Computer befindet sich nach erfolgtem Rollback wieder in exakt dem Zustand, der vor Beginn der fehlgeschlagenen Installation vorhanden war. Beim herkömmlichen Setup hinterließ eine abgebrochene Installation in der Regel die nicht komplett installierte Applikation auf dem System. Zudem bestand die Gefahr, dass vorhandene Applikationen nicht mehr funktionierten. Der Benutzer hatte also möglicherweise ein korruptes System, mit dem er nicht mehr arbeiten konnte.

Die Windows Installer-Technologie bietet für jeden während der Installation durchgeführten Schritt eine Undo-Möglichkeit. Auch bei der Deinstallation besteht diese Undo-Möglichkeit für jeden Schritt.

Dieses Rückgängigmachen aller Schritte bezieht sich auf die Wiederherstellung gelöschter oder überschriebener Dateien, Registry-Schlüssel und anderer Ressourcen. Alle Dateien, die während der Installation gelöscht oder modifiziert werden, werden in einen temporären Backup-Ordner gespeichert. Kann die Installation nicht ordnungsgemäß abgeschlossen werden, werden die Inhalte dieses Ordners wiederhergestellt. Ist die Installation erfolgreich verlaufen, werden die Inhalte des Ordners gelöscht. Somit ist sichergestellt, dass bei einer fehlgeschlagenen Installation der vorherige Zustand wiederhergestellt werden kann. Beim Rollback spricht man auch von *Transacted Installation*.

Ein Rollback kann lediglich bei erfolgloser Installation durchgeführt werden. Bei einer erfolgreichen Installation greift der als Sicherheitskonzept gedachte Rollback-Mechanismus nicht.

Nun noch ein Wort zu den Anforderungen an Speicherplatz für den Rollback-Mechanismus. Angenommen, eine Applikation soll upgedatet werden. Das Update erfordert 200 MB Plattenplatz, jedoch können 40 MB der bereits vorhandenen alten Applikation entfernt werden. In diesem Fall benötigt die Applikation eine Netto-Kapazität von 160 MB. Da während der Installation jedoch die 40 MB im temporären Backup-Verzeichnis vorgehalten werden, werden während der Installation die vollen 200 MB belegt. Erst nach erfolgreicher Installation werden die 40 MB gelöscht, und lediglich der Netto-Speicherplatz von 160 MB ist belegt. Steht nicht genügend Speicherplatz für die Dateien im temporären Backup-Verzeichnis zur Verfügung, so kann der Rollback-Mechanismus deaktiviert werden, damit die Installation dennoch durchgeführt werden kann. Treten in diesem Fall Probleme auf, kann die Installation jedoch nicht rückgängig gemacht werden. Das Rollback wird vom Windows Installer-Dienst auf sämtlichen Windows-Betriebssystemen unterstützt.

3.2.10 Installationsquelle

Für jede Installation bei Bedarf, Neuinstallation oder weiteren Konfigurationen muss der Windows Installer-Dienst Zugriff auf die Installationsquelle haben. Handelt es sich bei der Installationsquelle um eine Netzwerkfreigabe, so ist der Installer-Dienst in der Lage, ein alternatives Netzlaufwerk zu benutzen, wenn die Verbindung zu seiner ursprünglichen Installationsquelle nicht hergestellt werden kann. Um dieses Feature nutzen zu können, muss der Administrator bei der Verteilung der Applikation eine Liste alternativer Installationsquellen der Applikation angeben. Ist keine der in der Liste hinterlegten Quellen verfügbar, so besteht die Möglichkeit, dass der Benutzer manuell seine verfügbaren Laufwerke durchsucht und eine Quelle angibt. Soll diese Möglichkeit nicht gegeben werden, kann sie in den Gruppenrichtlinien deaktiviert werden. Wie auch beim Rollback wird für diese Funktionalität keine bestimmte Windows-Betriebssystemversion vorausgesetzt.

3.2.11 Updates und Patches

Zur Identifizierung verwandter Produktgruppen können vom Softwareentwickler für jede Gruppe Upgrade Codes (GUIDs) festgelegt werden. Werden diese zusätzlich mit einer Produktversion kombiniert, identifiziert der Upgrade Code eindeutig ein bestimmtes Produkt. Somit ist es möglich, ältere und neuere Versionen einer bestimmten Applikation zu bestimmen. Über diese Identifikation kann der Softwareentwickler bestimmen, ob beispielsweise eine ältere Produktversion entfernt werden soll oder ob eine ältere Version installiert werden darf, wenn bereits eine neuere vorhanden ist.

Auch für die Anwendung von Patch-Dateien (*.msp*-Dateien) verfügt der Windows Installer-Dienst über einen eigenen Mechanismus. Patch-Dateien werden für ein Update oder eine Reparatur der Applikation verwendet. Nach der Installation bleibt die Patchdatei auf dem System vorhanden und wird gemeinsam mit den Originaldaten der Installationsquelle für Installationen bei Bedarf und die Selbstreparatur benutzt. Bei Netzwerk-

installationen wird die Patchdatei in der Regel auf den administrativen Installationspunkt der Installationsquelle und nicht auf die Applikationen der einzelnen Benutzer angewendet. Über die von Microsoft bereitgestellten *QFE-(Quick Fix Engineering-)*Patches können die Clients vom Administrator sowohl über die Kommandozeile als auch über die API benachrichtigt werden. Soll hingegen ein vollständiges Servicepack eingespielt werden, so muss eine herkömmliche Installation von dem aktualisierten Installationspunkt als Quelle durchgeführt werden. Der Windows Installer-Dienst behandelt diese Installation als ein herkömmliches Update.

3.2.12 Anpassungen über Transform Files

Für die individuelle Anpassung der Installation mussten Administratoren das Setup-Skript modifizieren. Bei ähnlichen Änderungen an mehreren verschiedenen Skripten mussten die Änderungen für jedes einzelne Skript separat durchgeführt werden. Beim Windows Installer ist eine Modifikation des Installationspakets zum Zeitpunkt der Installation über ein Transform File möglich. In ein einzelnes Transform File können beliebig viele Anpassungen aufgenommen werden. Ein Transform File kann auf beliebig viele Installationspakete angewendet werden, sofern die gewählten Modifikationen auf das Paket anwendbar sind. Hierzu muss die Komponente, deren Keypath über das Transform File geändert wird, in dem Installationspaket vorhanden sein. Wie Patchdateien verbleiben auch Transform Files auf dem Computer. Die Transform Files werden bei jeder Konfigurationsänderung der Applikation erneut angewendet. Dies gilt auch für eine erneute Installation der Applikation. Ein Transform File kann jedoch nur bei der ersten Installation angewendet werden. Soll eine bereits installierte Applikation über ein Transform File modifiziert werden, so muss zunächst die gesamte Applikation deinstalliert und danach mit dem angewendeten Transform File neu installiert werden.

3.2.13 Umgebungen mit eingeschränkten Berechtigungen

Für zahlreiche Arbeitsplätze sind die Benutzerberechtigungen so weit eingeschränkt, dass ein Benutzer über keinerlei Schreibrechte für das Dateisystem und die Registry verfügt. Diese eingeschränkten Berechtigungen verhindern zwar versehentliche oder absichtliche Änderungen an der Konfiguration, aber bei der Installation neuer Applikation ergibt sich in derartig gesicherten Umgebungen ein Problem, da für die Installation einer Applikation häufig Administratorrechte vorhanden sein müssen. So mussten für die Installation entweder dem Benutzer erweiterte Berechtigungen zugeteilt werden, oder aber der Administrator selbst musste die Installationen vornehmen. Die Windows Installer-technologie schafft auch für diese Problematik Abhilfe. Unter Windows NT 4.0, 2000, XP und 2003 kann der Windows Installer-Dienst in zwei verschiedenen Kontexten ausgeführt werden. Zum einen kann er im Benutzerkontext ausgeführt werden, zum anderen aber auch unter dem Konto LOKALES SYSTEM, das über erweiterte Berechtigungen verfügt als ein gewöhnliches Benutzerkonto. Über die Gruppenrichtlinien kann bestimmt werden, dass für Applikationen Installation, Deinstallation und Reparatur unter dem Konto des lokalen Systems durchgeführt werden sollen. Somit bleibt die durch eingeschränkte Benutzerrechte abgesicherte Umgebung erhalten, während gleichzeitig dennoch die Installation von Applikationen ermöglicht wird. Für Windows NT 4.0 gilt jedoch die Einschränkung, dass entweder alle oder gar keine Applikationen unter

dem Konto des lokalen Systems installiert werden dürfen. Unter Windows 2000 und höher kann für jede Applikation separat festgelegt werden, in welchem Kontext die Installation erfolgen soll.

3.2.14 Windows Installer und SMS/Active Directory

Die Windows Installer-Technologie spielt auch in Umgebungen eine wichtige Rolle, in denen die Softwareverteilung zentral über den Systems Management Server (SMS) oder das Active Directory verwaltet wird. Der SMS-Server unterstützt ab der Version 2.0 Windows Installer-basierte Applikationen. Für die Softwareverteilung an Benutzer und Computer über die Gruppenrichtlinien des Active Directory müssen die zu verteilenden Applikationen im Windows Installer-Format vorliegen. Ist dies nicht der Fall, ist eine Repaketierung erforderlich.

3.3 Unterschiede gegenüber einem Setup-Programm

Bei einem Setup-Programm benutzt jede Applikation unterschiedliche Regeln für die Installation. Diese nicht standardisierten Regeln führen oftmals zu Problemen und Fehlern beim Setup. Das größte Problem sind hierbei die älteren Dateiversionen, die eine neuere Version überschreiben. Auch die Nutzung von gemeinsamen Dateien war oftmals problematisch. Applikationen lieferten teilweise keine korrekten Informationen über die gemeinsam genutzten Dateien. Wurde nun eine Applikation deinstalliert, wurde die gemeinsam genutzte Datei möglicherweise entfernt, und eine andere Applikation war nicht mehr funktionsfähig. Auch das Hinzufügen, Updaten oder Deinstallieren einer Applikation konnte bei einer herkömmlichen Setup-Routine zu Problemen führen, da auch hier jede Applikation ihre eigenen Regeln anwendete. Genau wie bei der Installation stellten auch hier Dateiversionen und gemeinsam genutzte Dateien das größte Problem dar.

Ein *.msi*-Paket hält sich an strikte Regeln. Diese Implementierungsregeln sind bereits fest im Betriebssystem implementiert. Dazu muss ein Installationspaket lediglich als Windows Installer-Paket gebaut und deklariert werden.

Bei gemeinsam genutzten Dateien wendet der Windows Installer eine besondere Technik an. Nachdem eine bestimmte Datei erstmalig installiert worden ist, registriert der Windows Installer deren Präsenz. Installieren Sie nun weitere Applikationen, die ebenfalls Windows Installer-basiert sind, werden diese Applikationen zu der so genannten *Client List* hinzugefügt. Diese Liste enthält alle Applikationen, die diese Datei benötigen. Diese Liste wird bei jeder Installation, Deinstallation oder auch bei Updates aktualisiert. Bei der Deinstallation wird keine Komponente der Applikation entfernt, für die noch Einträge anderer Applikationen in der Client List enthalten sind. Dieses Feature ist zwar auch bei anderen Setup-Technologien verfügbar, jedoch wird die Koordinierung wesentlich komplexer und schwieriger, da jeder Softwarehersteller eigene Installationsregeln anwendet.

Auch Fehler, die im Laufe der Installation auftraten, konnten zu einem instabilen Systemzustand führen, oder im besseren Falle „nur“ Dateileichen im Dateisystem und der Registry zurücklassen. Der Windows Installer stellt bei einer fehlgeschlagenen Installation über das Rollbackverfahren wieder den Zustand vor Beginn der Installation her. Dies ist besonders interessant bei dem Update einer Applikation. Trat während des Setups ein Problem auf, waren meist weder die ursprüngliche, noch die aktualisierte Version verwendbar, und eine Neuinstallation wurde fällig. Bei einem Windows Installer-basierten Setup steht bei einem Updatefehler die bisherige Version weiterhin uneingeschränkt zur Verfügung.

3.3.1 Anpassungen einer Applikation bei Setup und Windows Installer

Anpassungen einer Applikation können sich sowohl auf den Installationsprozess an sich, z.B. die Durchführung einer Silent Installation ohne notwendige Benutzereingaben, als auch auf die Anpassung der zu installierenden Features beziehen. Bei Setup-Programmen war eine dementsprechende Anpassung bestenfalls über die Modifikation der Datei *setup.inf* möglich. Anderenfalls mussten diese Applikationen zeitaufwändig mit Drittanbieterprogrammen bearbeitet werden, um die Anforderungen des Unternehmens zu erfüllen. Auch Kommandozeilenparameter waren nicht standardisiert. Dasselbe galt für Computer- und Benutzerbezogene Installationen. Der Windows Installer benutzt einheitliche, applikationsneutrale Befehle und Parameter. Zudem können bestimmte Eigenschaften, nämlich öffentliche Eigenschaften des Windows Installer, als Variablen in der Kommandozeile verwendet werden. Zusätzlich sind featurebezogene Anpassungen über Transform Files möglich. Diese Dateien ändern das standardmäßige Installationsverhalten einer Applikation.

Das einheitliche Installations- und Paketformat sämtlicher Windows Installer-basierten Applikationen erlaubt somit die gemeinsame Nutzung aller Installer-Eigenschaften und Kommandozeilenparameter für alle Applikationen.

Bei einem herkömmlichen Setup-Programm sind im Setup-Programm selbst zwei Bestandteile enthalten: zum einen die Beschreibung der Änderungen, die durch die Installation am System vorgenommen werden, und zum anderen der Code selbst in Form eines Installationsskripts, der die Änderungen durchführt.

Mit der Windows Installer-Technologie erfolgt eine Trennung der Beschreibung und Ausführung. Die Beschreibung der durchzuführenden Änderungen ist in dem *.msi*-Paket in Form einer Datenbank enthalten. Der Code, der für die Durchführung der Änderungen zuständig ist, wird über den Windows Installer-Dienst implementiert (siehe Abbildung 3.5).

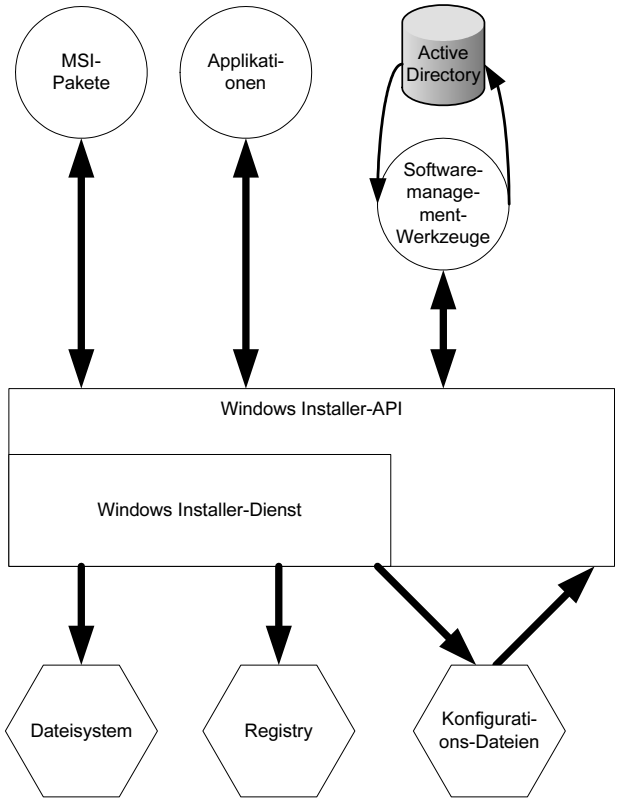

Abbildung 3.3: Die an einer Installation beteiligten Elemente

Sie sehen in Abbildung 3.5, dass das *.msi*-Paket den Sollzustand der Installation beschreibt. Auch eine repaketierte Applikation oder ein Softwaremanagement-Werkzeug wie das Active Directory können auf die Windows Installer-API zugreifen. Über diese API werden die Programme und Features installiert, Statusabfragen durchgeführt und Pakete erstellt. Der Windows Installer-Systemdienst schließlich führt den Installationsvorgang an sich aus, wobei Änderungen am Dateisystem und der Registry vorgenommen werden. Über die Konfigurationsdateien wird bestimmt, was auf dem Computer wie installiert wird.

3.3.2 Dllcache

Der Windows Installer entfernt bei der Deinstallation einer Applikation nicht alle zu ihr gehörigen *.dll*-Dateien, sondern behält auf dem System diejenigen bei, die auch von anderen Applikationen benutzt werden. Damit ist die Funktion dieser Applikationen gewährleistet. Nur wenn bei der Installation eine ältere *.dll*-Datei durch eine neuere Version überschrieben wurde, wird bei der Deinstallation die neuere *.dll*-Datei beibehalten und nicht wieder durch die ältere Version ersetzt.

Der Speicherort des dllcache befindet sich im Verzeichnis `%Systemroot%\System32\dllcache`. In diesem Ordner sind u.a. auch *.dll*-Dateien enthalten. Die Systemdateien dieses Ordners kann Windows wiederherstellen, nachdem diese überschrieben oder geändert worden sind. Befinden sich die Dateien im Ordner \DLLCACHE, ist zum Wiederherstellen die Windows-Installations-CD nicht erforderlich.

Der Inhalt dieses Ordners kann theoretisch gelöscht werden, allerdings kann die Wiederherstellung dann nur über die CD erfolgen. Die CD muss dann also bereitliegen. Um den Pfad des dllcache zu ändern, bearbeiten Sie in der Registry unter dem Pfad `HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\Setup` den Eintrag DRIVERCACHEPATH.

3.4 Dateierweiterungen des Windows Installer

Neben den *.msi*-Dateien können über den Windows Installer auch weitere Dateitypen installiert werden (siehe Tabelle 3.3).

Dateierweiterung	Abkürzung	Beschreibung
.msi	Managed Software Installation	Installationspaket
.mst	Managed Software Transform	Transform File
.msp	Managed Software Patch	Patchdatei
.msm	Managed Software Merge Module	Mergemodul

Tabelle 3.2: Die Dateierweiterungen des Windows Installer

Weiterhin spielen die in Tabelle 3.5 gezeigten Dateierweiterungen eine Rolle. Diese Dateien werden in späteren Kapiteln näher vorgestellt.

Dateierweiterung	Abkürzung	Beschreibung
.pcp	Patch Creation Properties File	Binäre Datenbankdatei im Format einer .msi-Datei, jedoch mit einem anderen Datenbankschema
.idt	Installer Database Table	Textdatei einer exportierten Tabelle der Windows Installer-Datenbank
.cub	-	Validierungsmodul

Tabelle 3.3: Weitere wichtige Dateierweiterungen im Zusammenhang des Windows Installer