brainwaregroup

Columbus Packaging Guide

User Manual

Module version 7.6

columbus

Issue: 12.18

Document titel	Columbus Packaging Guide - User Manual
Product version	7.6
Production and printing	Brainware Consulting & Development AG Sumpfstrasse 15 CH-6300 Zug
Release date	12.12.2018

Content

General

0.1 Document history

Version	Date	Author	Description
2.0.2	18.06.12	B. Nyffenegger	Release to QA
2.0.1	28.08.12	B. Nyffenegger	Detail control
2.0	18.07.12	B. Nyffenegger	Complete revision
1.3.2	15.02.11	H. Bergens	+ chapter *Reboot on demand*
1.3.1	07.06.10	B. Nyffenegger	(Release) + text revision + chapter *Dummy or placeholder packages* added
1.3	10.03.10	B. Nyffenegger	(Release) + chapter *Additional documents* + chapter *Scope of application* + Modification of name convention for log and configuration files. + The information of Info.rtf has been enhanced. + chapter *Variables resolution* + chapter *Information on SW-Pakets* + chapter *Additional tools in PackageStudio*
1.2	21.01.10	B. Nyffenegger	(Release) + chapter *Dummy* or placeholder packages + chapter Repetitive sections - %c_MachineCostCenter% variable removed
1.1	15.09.09	B. Nyffenegger	(internal) + Design adaptations, additional graphics, chapter *Repetitive sections* - %c_MachineCostCenter% removed
1.0	24.03.09	B. Nyffenegger	Release to project (external)
0.0	28.09.07	B. Nyffenegger	Initial creation (internal)

0.2 Supplementary documents

Please refer to the following manuals for additional information about Columbus Packaging Guide:

- Columbus **Technical Reference**
  Description of the script commands and variables as well as an overview of the Management Console configuration parameters (available in English language only).

0.3 Changes compared to the Packaging Guide 1.3.x

The present packaging guidelines have been developed with utmost care and consideration for existing projects. The further development of the peripheral systems, requirements and scope of application of Columbus (e.g. Shareless) have made the reform of the packaging guidelines unavoidable.

The major change relates to the use of the machine parts (Server and Client) of a SW-Paket:

In the future, all operations within the server script part will be processed. This ensures that, upon the completion of the server script part, the application has been completely installed and is ready for use.

The client script part is no longer used. User preferences have been minimized and will be processed in the user script part as before.

Note This change does not affect any SW-Pakets that were created using the Snapshot technology. They will be treated as before (delivery, configuration).

SW-Pakets are now divided into three complexity levels. These levels are called "platforms":

EASY: These SW-Pakets are created for the stand-alone application and can be used in every Columbus environment with no effort. No configuration settings (except for the deactivation of the autoupdate mechanisms) are made and no shortcuts or similar are changed.

ADVANCED: These SW-Pakets allow for carrying out settings by using a configuration file. The configuration file can be stored either centralized ((%C_ConfigPath%) or decentralized. No further efforts or requirements are needed for the Columbus environment. No additional, central variables are used.

EXPERT: All available script technologies and variables (from the customer, etc.) can be applied and used in this platform. Such SW-Pakets usually depend strongly on the customer and require extended knowledge of the packaging technology on the one side and of the customer environment on the other.

The target is a significant leaner structure and reduced complexity, but at the same time a higher interoperability of SW-Pakets over several SW releases.

CHAPTER 1

Basics on SW packages

In this chapter

1.1 Machine and user parts

SW-Pakete consist of a machine part and a user part. The two parts are executed with different user accounts:

- Machine parts on the one hand with a system account of Columbus service, and on the other hand with a local Admin-Account.
- User parts are executed in the context of each logged-in user.

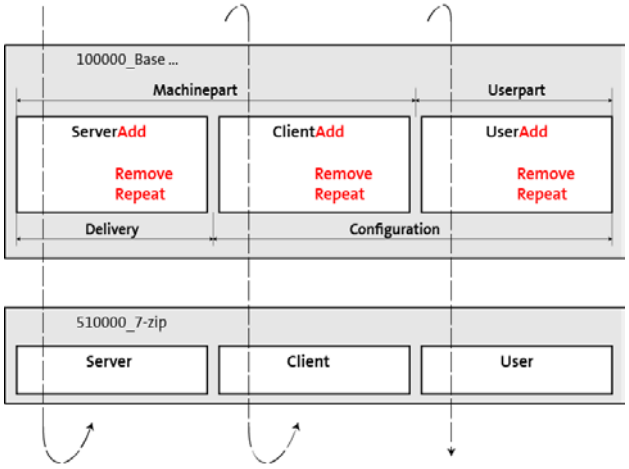
The machine part is subdivided into a server and a client part.

1.2 Installation sequence of the SW-Pakete

After assigning a SW-Paket in the Management Console, the Management Client is pushed to receive the new assignment from the database. After that, the SW-Pakets are processed in the following sequence:

- All server parts of the assigned SW-Pakets.
- All client parts of the assigned SW-Pakets (if the related server part is done).
- All user parts of the assigned SW-Pakets (if the related client part is done).

Wanted or unwanted EXIT marks can occur within the package parts. These parts are restarted after a restart or client push. The message related to an EXIT is also transmitted to the Management Console, similar to a successful completion.



1.3 Processing within the SW-Paket

Note	This chapter does not apply to any snapshot-based SW-Paket. These are created and processed as before.

Following the general trend towards the simplification of installation tasks, SW-Pakets for Columbus are also created with a clear and comprehensible processing. This orientation is especially recommended when using Shareless for SW deployment.

The aim is to process the machine-related actions only in the server section of a SW-Paket instead of in the Server and Client sections as before.

The advantage of this is that, after processing a Server section, the processed software is effectively available. This also ensures that requirements with a lower *Identifier* order are executed before installing an application.

In this sense, the Client section is used in any case for cleanup work.

User-specific actions are processed as before in the User section.

1.4 SW-Pakets used for Shareless

When SW-Pakets are used for Shareless, it must be especially ensured that a SW-Paket contains everything that will be used in the script parts. During Shareless processing, no access is possible to external systems or additional UNC paths.

Since the complete SW-Paket is first created locally as "Container", it corresponds to the runtime %_PkgSource% = %_PkgCache%. For instance, it makes no sense to copy data from %_PkgSource% to %_PkgCache%, since they are copied and stored again quasi locally in Shareless.

1.5 Conventions for SW-Pakets

1.5.1 Numbering (Identifier)

On the one hand, the SW-Pakets are numbered in order to ensure a clear identification. On the other hand, the numbering defines the sequence of installation / de-installation. The numbering starts at 100000 and ends with 999999. This ensures that new or special SW-Pakets can be executed prior to or after the existing packages.

In principle, there are increments of 100 within the number blocks in order to create new SW-Pakets. However, related applications may be combined in increments of 10.

Number block	Description
100000 – 299999	Basic configuration, system utilities, drivers, runtimes, prerequisites, player e.g. VMTools, ResourceKit, MS .net, C++, Java, Mediaplayer, etc.
300000 – 499999	Standard applications e.g. Microsoft, Adobe, SAP, etc.
500000 – 899999	Other applications e.g. Skype, Mozilla, VMware, etc.
900000 – 949999	Brainware products e.g. Packaging Tools, console, etc.
950000 – 999999	Final steps e.g. Final Config Package

In order to prevent duplicates or identifiers which are too similar, an identifier is centrally managed for each SW-Paket.

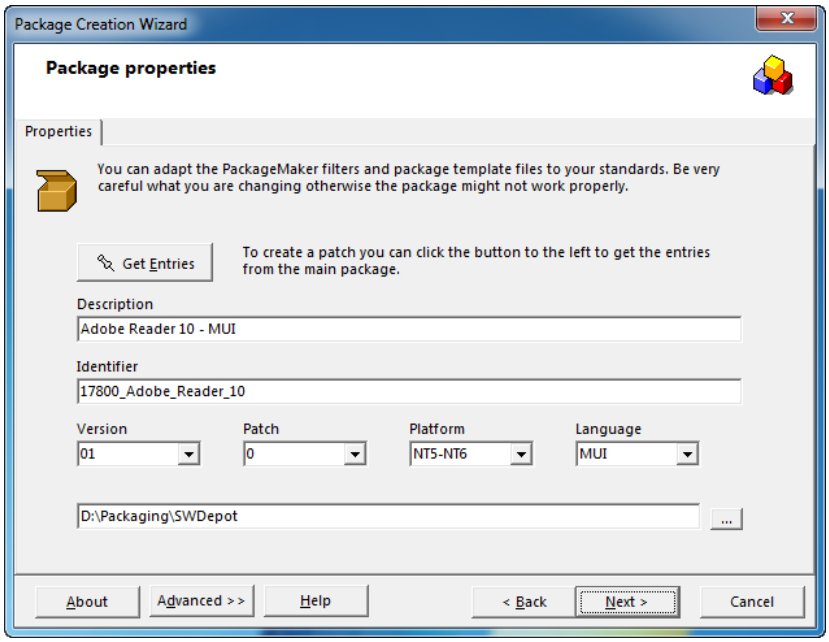
- In principle, the following applies:
  NUMBER_MANUFACTURER_PRODUCT_VERSION
- Information on the contents may be added:
  NUMBER_MANUFACTURER_PRODUCT_VERSION_CONTENTS
- For known manufacturers, an abbreviation may be used:
  Microsoft MS
  Brainware BW
- The version is specified together with the main version:
  7.03.09 7
  10.0.0.1 10

Examples:

Application	Identifier
Adobe Reader 10	321000_Adobe_Reader_10

Microsoft Office 2007	301100_MS_Office_2007_Enterprise
Microsoft Office 2007 Settings Package	301100_MS_Office_2007_Enterprise_Settings
Microsoft Office 2003 MUI Package	386100_MS_Office_2003_MUI_LangPack

- The combination of the above specifications results in the identifier of a SW-Paket.
- If the language code is changed for an existing SW-Paket, then the Management Console as well as the Management Client will consider this SW-Paket as a new entry.
- The language is no fixed component of the identifier, but is used for the technical identification of a SW-Paket.
- The name of the SW-Paket directory contains the identifier as well as the language.



Note The length of the identifier must not exceed 46 (or 50 minus 4 as buffer) characters.

Identifier (central)

In order to prevent duplicates or identifiers which are too similar, an identifier is centrally managed for each SW-Paket.

In principle, the following applies:
NUMBER_MANUFACTURER_PRODUCT_VERSION

Information on the contents may be added:
NUMBER_MANUFACTURER_PRODUCT_VERSION_CONTENTS

For known manufacturers, an abbreviation may be used:

Microsoft	MS
Brainware	BW

The version is specified together with the main version:

7.03.09	7
10.0.0.1	10

Examples of package identifiers:

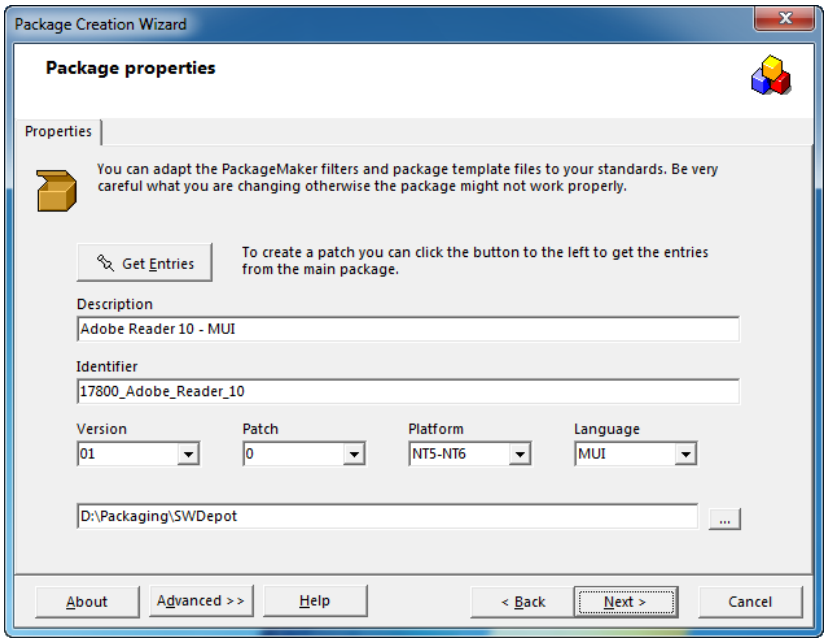
Application	Identifier
Adobe Reader 10	32100_Adobe_Reader_10
Microsoft Office 2007	30110_MS_Office_2007_Enterprise
Microsoft Office 2007 Settings Package	30110_MS_Office_2007_Enterprise_Settings
Microsoft Office 2003 MUI Package	38610_MS_Office_2003_MUI_LangPack

The combination of the above specifications results in the identifier of a SW-Paket.

If the language code is changed for an existing SW-Paket, then the Management Console as well as the Management Client will consider this SW-Paket as a new entry.

The language is no fixed component of the identifier, but is used for the technical identification of a SW-Paket.

The name of the SW-Paket directory contains the identifier as well as the language.



Note	The length of the identifier must not exceed 46 (or 50 minus 4 as buffer) characters.

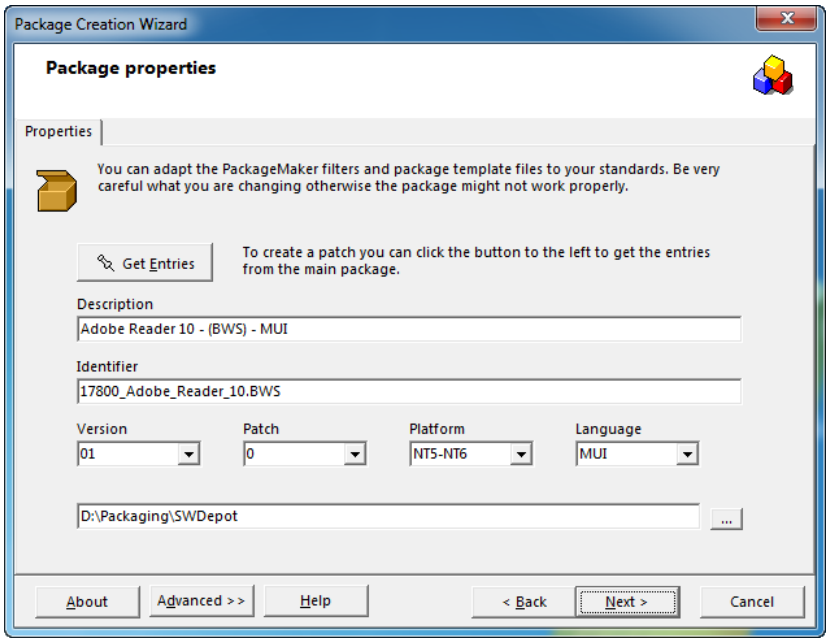
Identifier (branch office)

In order to ensure a clear identification, an abbreviation is added to the identifiers of SW-Pakets from branch offices.

A branch office creates a SW-Paket exclusively for its own use:
32100_Adobe_Reader_10.nnn Identifier.Abbreviation

Later on, a second and third branch office will need this SW-Paket. The SW-Paket can now be analyzed holistically and modified accordingly before it is added to the central SW release of the whole company.

321000_Adobe_Reader_10 Identifier according to whole company (central SW-Paket)



Note	The length of the identifier must not exceed 46 (or 50 minus 4 as buffer) characters.

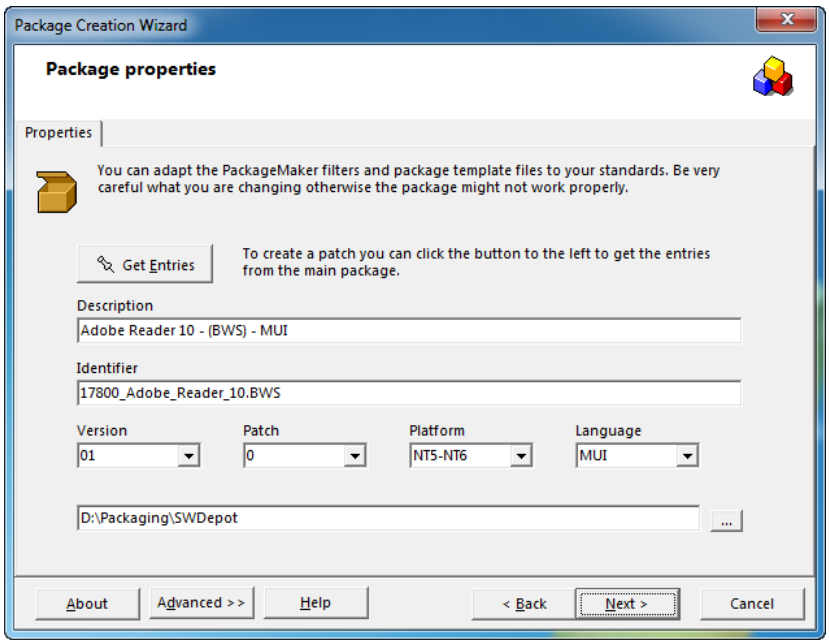
Identifier (centrally based)

In order to ensure a clear identification, an abbreviation is added to the identifier to mark central SW-Pakets which have been changed by a branch office.

A branch office changes an existing central SW-Paket exclusively for its own use:
321000_Adobe_Reader_10 321000_Adobe_Reader_10**.nnn**

This clearly identifies the origin of the SW-Paket. If a central SW-Paket is updated, it will be easy for the branch office to determine whether these changes have to be applied for their own SW-Paket.

Changes to central SW-Pakets have no direct effect on modified own packages.



Note	The length of the identifier must not exceed 46 (or 50 minus 4 as buffer) characters.

1.5.2 Description (central)

Name of the packed application. When the SW-Pakets are assigned in the Management Console, only this name is visible.

In principle, the following applies: Manufacturer Product Version - Language

Information on the contents may be added: Manufacturer Product Version Contents - Language

For known manufacturers, an abbreviation may be used:
Microsoft MS
Brainware BW

The version is specified together with the main version:
7.03.09 7
10.0.0.1 10

Examples of package descriptions:

Application	Description
Adobe Reader 10 German	Adobe Reader 10 - DEU
Microsoft Office 2007 English	MS Office 2007 – ENU
Microsoft Office 2003 Settings Package	MS Office 2007 Settings - UNI
Microsoft Office 2003 MUI Package	MS Office 2003 MUI LangPack - MUI

Note A modified description will be accepted automatically in the Management Console after carrying out a *Schedule Refresh Software Depot* by the Software Deployment Agent and will not trigger a reinstallation of the SW-Paket.

1.5.3 Description (branch office)

If one organization has several administration points (IT departments) which must create their own SW-Pakets, these can be distinguished from one another by using abbreviations. In principle, the same conventions apply as described above; in addition, the abbreviation is stored:

Manufacturer Product Version – (nnn) – Language

- Adobe Reader 10 - (BWS) - DEU

1.5.4 Language codes

Language codes for SW-Pakets

The language codes for SW-Pakets are used within the *Description* or the SW-Paket directory to distinguish among the application languages.

Language	Code
German	DEU
French	FRA
Italian	ITA
English	ENU
Multi-language	MUI
Universal	UNI

Language codes in the user part

The language codes for users are only used for the language configuration of applications within the **user part** (UserAdd). The language code **CANNOT** be used for a regional configuration / identification.

Language	Code
German	DEU
French	FRA
Italian	ITA
English	ENU

Language codes in the machine part

The language codes for computers are only used for the language configuration of applications within the **machine part** (ServerAdd, ClientAdd). The language code **CANNOT** be used for a regional configuration / identification.

Language	Code
German	DEU
French	FRA
Italian	ITA
English	ENU

1.5.5 Version

We recommended to use only one SW-Paket version: **01**

The function for further versions has been discontinued.

1.5.6 Patches

The master SW-Paket must always have the patch version **0**. This number is increased by each patch SW-Paket.
Max. length: **10 characters**

Patches within SW-Pakets may contain e.g. updates, upgrades or a modified configuration of the installed application.

Central patches are numbered as follows: 0100, 0200, 0300, etc.

Branch offices may add or modify their own patches.

Central SW-Paket	321000_Adobe_Reader_10	
Central patch	01	0100
Branch office patch	01	0100nnn100

Central SW-Paket	321000_Adobe_Reader_10		without patch
Branch office patch	01	0000nnn100	

* nnn corresponds to the company abbreviation of the branch office

1.5.7 Platform (complexity)

Contributes to a unique package ID, but has no technical function.
Max. length: **10 characters**

The platform of a SW-Paket is used for a summary in the Management Console and has no effect on the compatibility of a SW-Paket on a certain operating system. It describes the complexity of a SW-Paket:

Platform	Description
EASY	SW-Pakets with simple structure without using configuration files or central variables (runs on every Columbus environment without any effort
ADVANCED	The SW-Paket uses configuration files or central variables (no fixed values in the SW-Paket, all is controlled by a configuration file)
EXPERT	Any contents, technology, variable and dependency is allowed / included

Usage matrix:

Feature	EASY	ADVANCED	EXPERT
Check requirements	X	X	X
Snapshot-based			X
.msi, .exe (silent)	X	X	X
Variables in console		*(optional C_ConfigPath)*	X
Configuration files		X	X
Language selection			X
Basis for Package2Go	X	X	
Obtained through PCC	X	X	X

1.5.8 Status

The status of a SW-Paket shows the steps which have already been carried out or the future steps which have yet to be carried out on the SW-Paket.

The following states are set:

Status	Description
Raw Snapshot	Newly created SW-Paket. Will be edited by the Package Engineer.
Ready for QA	SW-Paket provided for QA.
Ready for Test	SW-Paket prepared for the integration test.
Ready for Pilot	Ready to be used in a pilot test.
Productive	The SW-Paket has been released for production.
Obsolete	The SW-Paket is no longer used (no new assignments).
Archive	The SW-Paket is uninstalled everywhere and is then deleted from the SW release.

CHAPTER 2

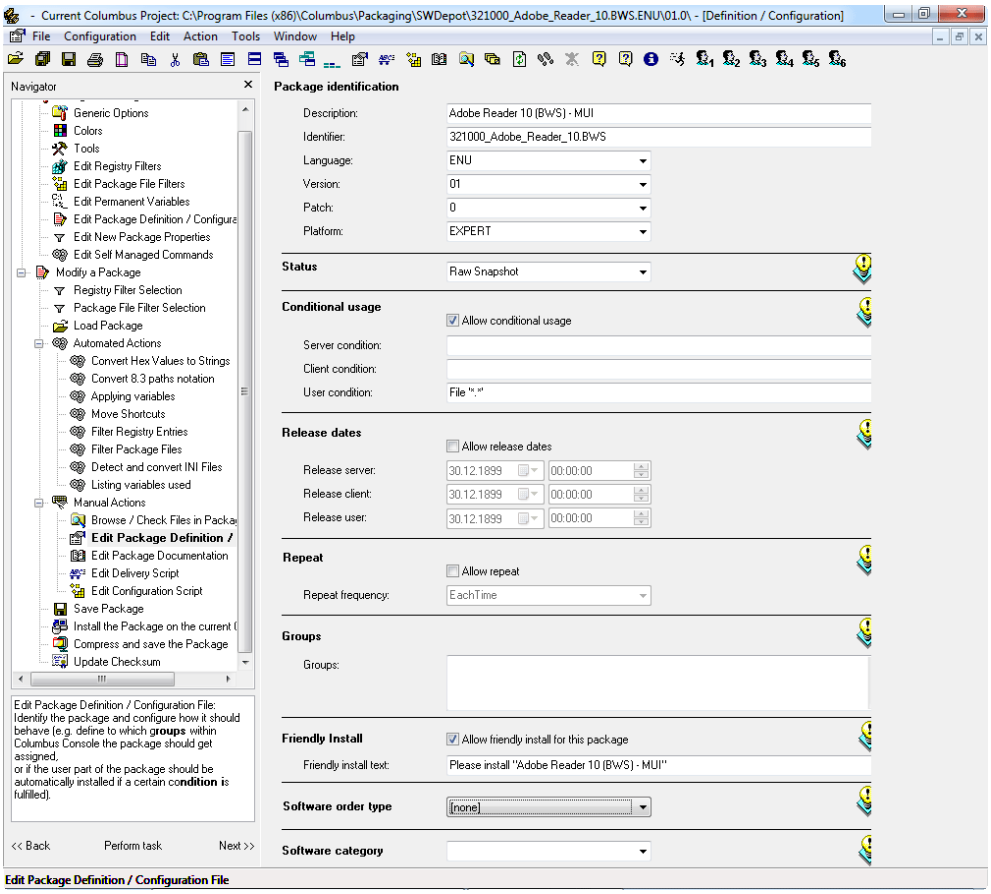
Processing SW-Pakete

In this chapter

Important	Columbus SW-Pakets are processed with Columbus PackageStudio only.

2.1 Package Definition / Configuration

In this summary, the definition of the SW-Paket is set, e.g. description and identifier. Furthermore, the behavior of the SW-Paket can be controlled on a Management Client. If this summary is modified, it has to be re-imported into the Columbus Database (Schedule Refresh Software Depot in Software Deployment Agent).

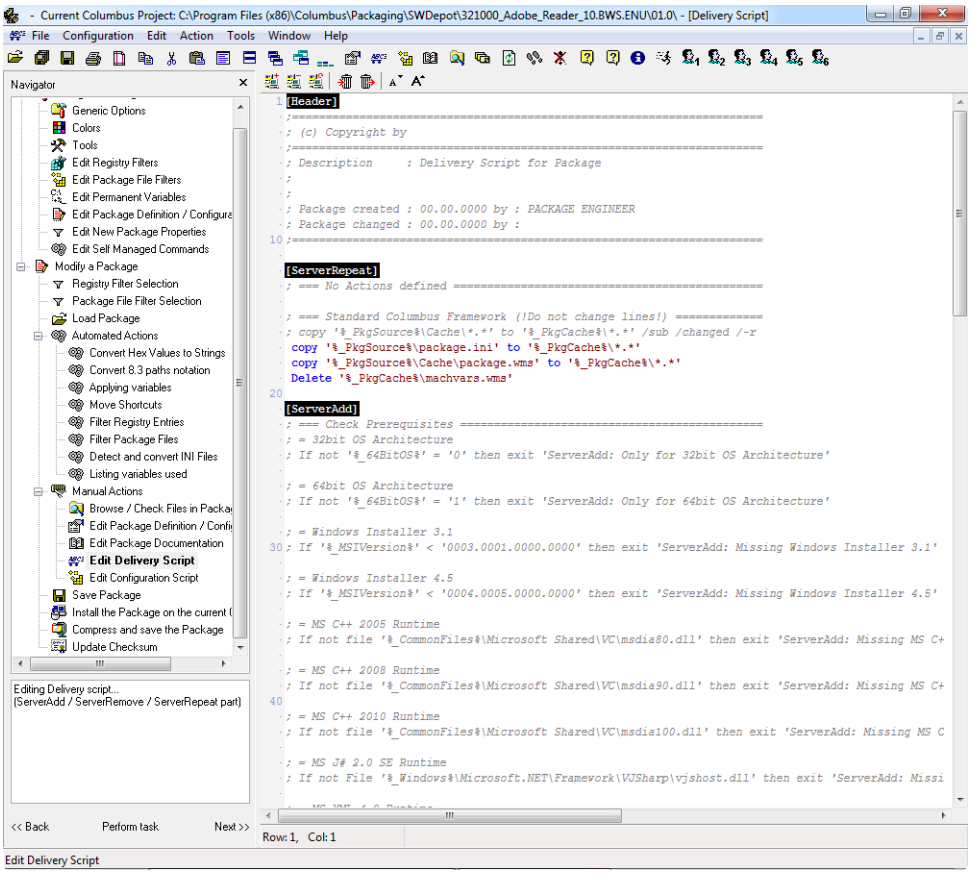


Section	Description
Package Identification	
• Description	Name of the packed application. When the SW-Pakets are assigned in the Management Console, only this name is visible.
• Identifier	Internal, alpha-numeric name of the SW-Paket.
• Language	Information for the SW-Paket management.
• Version	Version of the SW-Paket (**01** only)
• Patch	The master package must always have the patch version **0**. This number is increased by each patch package.
• Platform	Complexity of the SW-Paket.
Status	Status of the SW-Paket in the quality process
Conditional usage	
• Allow Conditional Usage	Defines whether the SW-Paket is to be assigned to the machine only and whether the user part should be executed for each logged-in user. Is set by default in order to facilitate the user administration.
• Server	Not used.
• Client	Not used.

Section	Description
• User Condition	**File '*.*'** Default setting, in order for the automated user assignment to work.
Release dates	Not used.
Repeat	
• Allow Repeat	Defines whether the SW-Paket is entered as Repeat. If yes, the sections `[ServerRepeat]` and `[Client Repeat]` will be executed at each start of Columbus. On each user login, the section `[UserRepeat]` will be executed. The Repeat sections have to be created manually in the SW-Paket.
• Repeat Frequency	Has to be set to "Each Time" in order to enable the above configuration.
Groups	In Columbus 7, this setting has to made directly in the Management Console.
Friendly Install	
• Allow friendly Install for this Package	If the Management Client has been configured on a machine in such a way that the user has to authorize the installation of a SW-Paket beforehand, the above mentioned configuration has to be set in the SW-Paket. Otherwise, the SW-Paket will be installed regardless of the client configuration.
• Friendly Install Text	This message will be shown to the user, when he is prompted for installation.
Software Order Type	In Columbus 7, this setting has to made directly in the Management Console.
Software Category	The SW-Paket can be divided in different categories from which a user can select, if he makes use of the software kiosk.

2.2 Delivery script

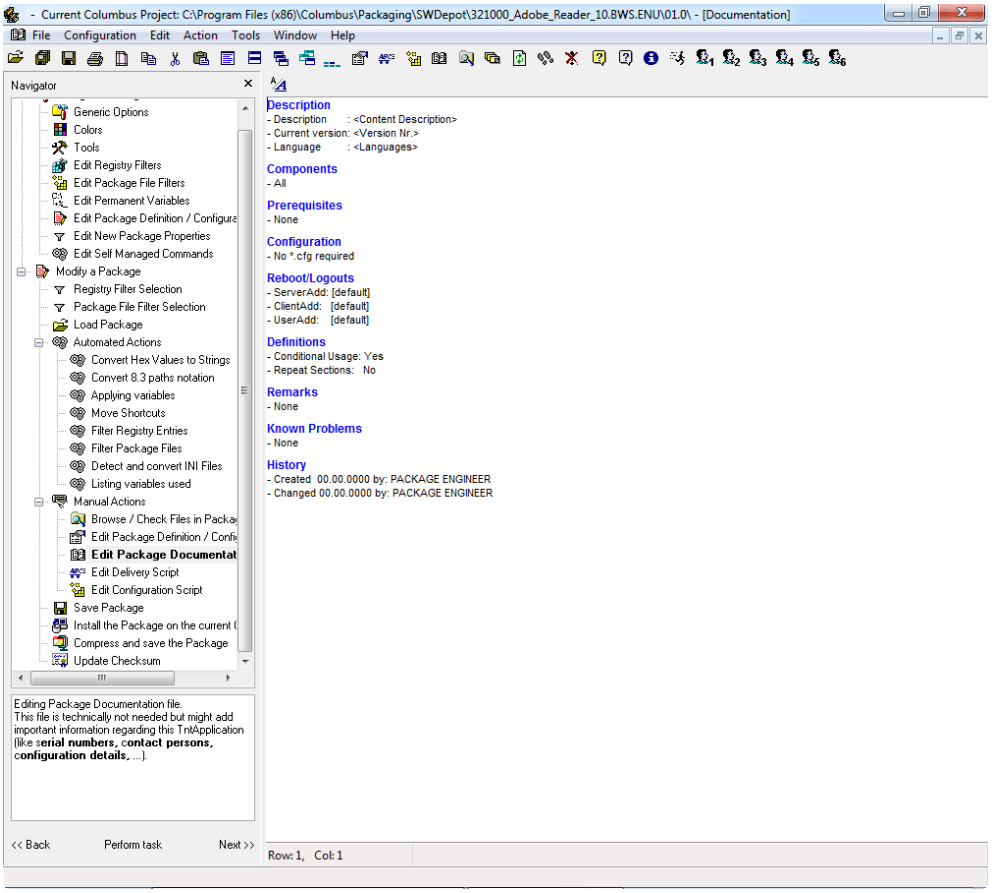
The delivery script contains the server section (part of the machine installation).



Section	Description
[Header]	Basic SW-Paket information
[ServerAdd]	This part of the SW-Paket is carried out first and contains in most cases an inspection of pre-installed SW-Paket or software. If all possible checks have a positive result, the application data and the other parts of the SW-Paket will be copied (advanced machine part and user part). Further tasks include: • Installation and repair • Changes in the registry
[ServerRemove]	This part is executed, if the SW-Paket in the Management Console is set to *Remove*. Tasks include: • De-installation of cleanup

2.3 Package Documentation

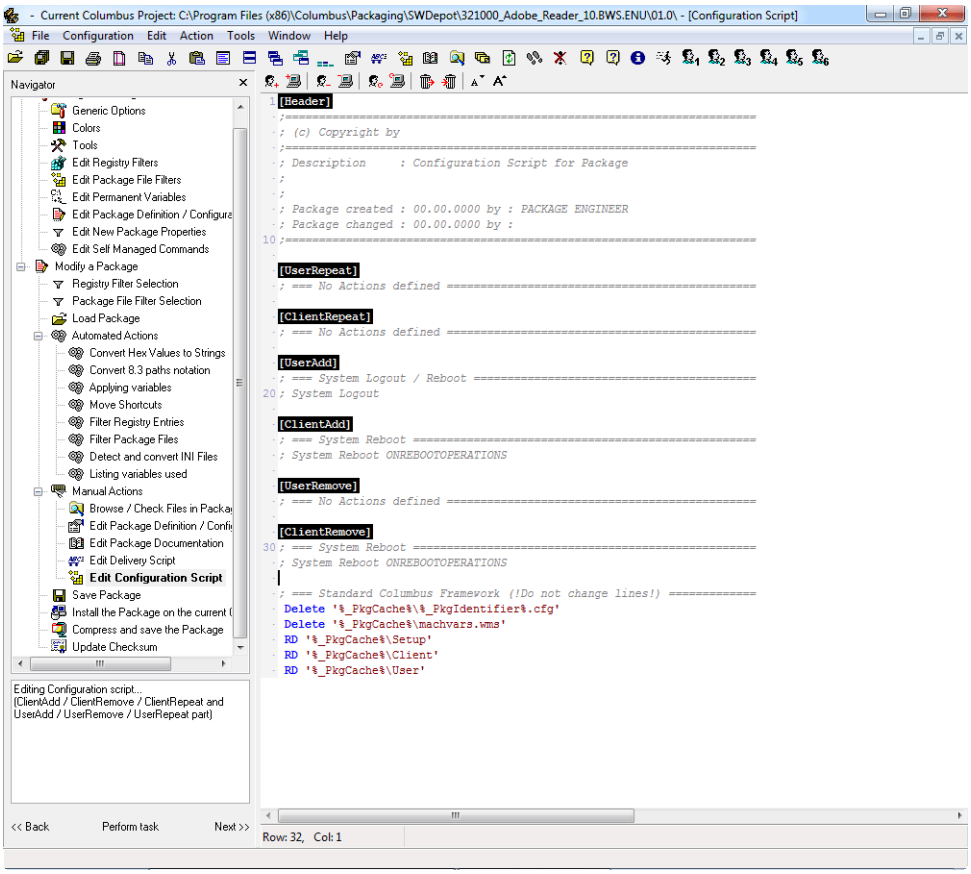
The specifications in this summary can be displayed in the Management Console by double-clicking on a SW-Paket. It contains information about the contents of the SW-Paket. The specifications vary according to the degree of detail and requirements of the quality process. If the documented information is more specific, it becomes easier to carry out modifications or enhancements on the SW-Paket later on.



Section	Description
Description	Name of the SW-Paket, current version of the installed software and language.
Components	An application can have several components; here it is specified which of them have been installed; e.g. Microsoft Office can consist of Word and Excel only, if Outlook has not been installed.
Prerequisites	The existence of another application or runtime (e.g. Microsoft .Net 1.1) may be a prerequisite for the application to be installed to be able to run on the system.
Configuration	SW-Pakets may be dynamized with variables. The required variables can be stored in the Management Console or in special configuration files.
Reboot	Qualifier stating whether a SW-Paket requires a system restart. A system restart can either be collected (System Reboot) or directly requested (System Reboot Immediate).
Remarks	Remarks on the SW-Paket (e.g. additional requirements)
Known Problems	Known problems which can occur with this SW-Paket.
Creator	Original creator of the SW-Paket. Very important if there are problems with the SW-Paket or a modification is intended.

2.4 Configuration script

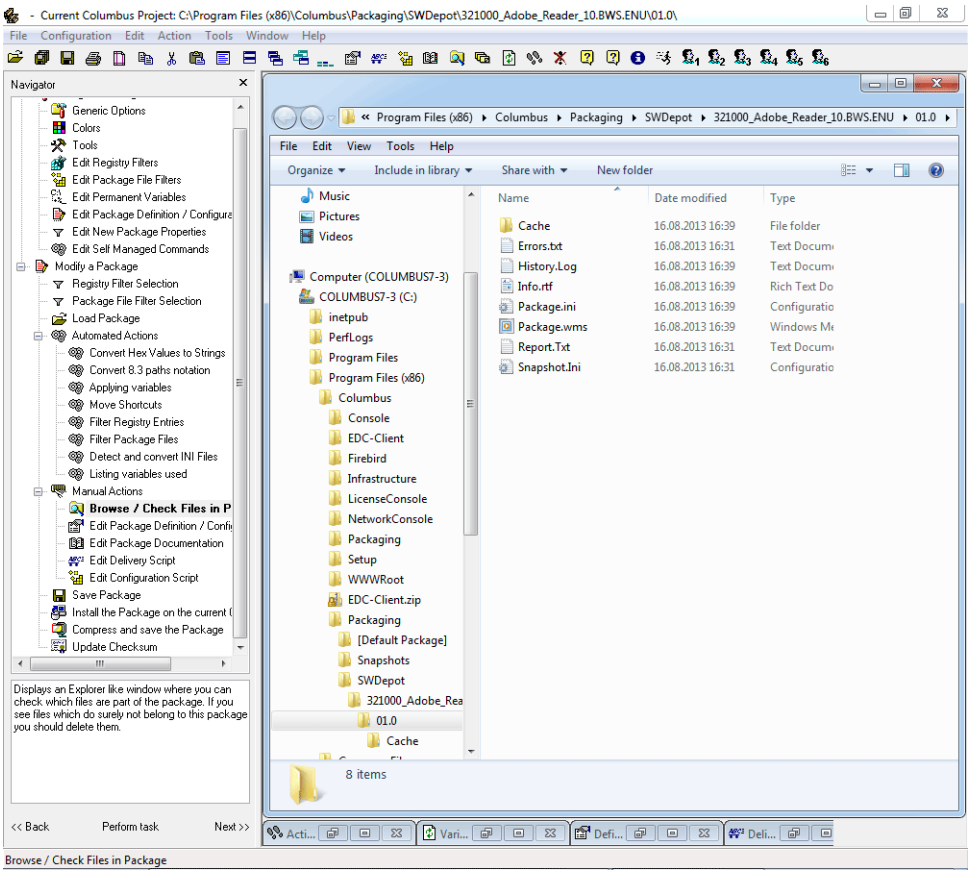
The configuration script contains the advanced part of the machine installation as well as the user installation part.



Section	Description
[UserAdd]	Takes care of the configuration / installation of the application for the user. The script commands are executed in the context of the current logged-in user. Further tasks: • Application data entries • User-specific variables • Shortcuts in the user profile
[UserRemove]	De-installation of data and configuration settings relevant for the user. Further tasks: • Removing user variables • Removing user configurations from the registry or deleting configuration files
[ClientAdd]	• Operations that were possibly not completed in the first machine part
[ClientRemove]	Contains the routine for deleting the application as well as the clean up of installation files. Further tasks: • Cleanup

2.5 Browse / Check files in the package

This summary shows the data structure of a SW-Paket. The files and directories can be either edited or deleted.



2.6 Remove in SW-Pakets

The following rules for a removal must be observed, regardless of the used packaging technology:

Task	Description
ServerRemove	The main application directory is deleted in ServerRemove. Please note that, if manufacturers write in the same main directory, only the directory for the corresponding application should be removed. Windows and System32 files are only removed in ClientRemove, if they are unique. File shortcuts to the removed software are released. HKLM\Software main key is deleted. The same rules apply as for ServerRemove. AllUser shortcuts are removed.
ClientRemove	Except for the script parts contained in the Columbus framework, usually no further action is required.
UserRemove	The User shortcuts are removed. In case of shortcut groups, entering If "%_Programs%\Adobe*.lnk" then RD "%_Programs%\Adobe" ensures that the shortcut directory "Adobe" is only deleted, if it contains no more shortcuts.

Task	Description
	HKCU\Software main key is deleted. The same rules apply as for ServerRemove.

2.7 SW-Paket Columbus Packaging Tools

In order to be able to create efficient and professional SW-Pakets in the productive environment, we recommend to distribute PackageMaker and PackageStudio with a Columbus SW-Paket. For this purpose, the brainwaregroup makes a SW-Paket available, containing the latest packaging tools and documents.

Selecting the required package template ([DefaultPackage])

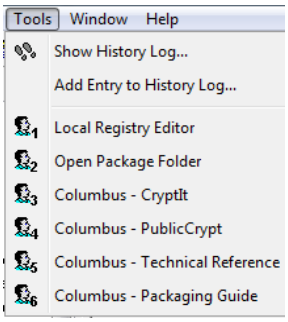
Several package templates can be found in the storage location of the packaging tools (e.g. C:\Packaging). These are identified as [Default Package].#PLATFORM#. Select the required package template as necessary by manually renaming the directory after [Default Package] (removing the suffix .#PLATFORM#).

Note	If no selection is made, the package template for EASY will be used.

2.8 Additional tools in PackageStudio

When the PackagingTools are installed in a Management Client using the SW-Paket Columbus *PackagingTools 7 – UNI*, the functionality of PackageStudio is supplemented with third-party applications.

The menu item **Tools** offers the following tools:



The same tools are also displayed in the tool bar of an open SW package:

Entry	Description
Local Registry Editor	"regedit" of the local machine, multiple start is possible
Open Package Folder	When a SW package is loaded, the path to %_PkgSource% is opened in Windows Explorer
Columbus – CryptIt	Cryptit.exe to encrypt passwords used with Columbus
Columbus – PublicCrypt	PublicCrypt.exe to encrypt passwords used with third-party manufacturers
Columbus – Technical Ref	Access to the *Technical Reference* of Columbus (explanation of all Columbus script commands and further useful information)
Project – Packaging Guide	Access to this document

CHAPTER 3

Details for SW packaging

In this chapter

3.1 SW packaging technology

If MSI or Silent Setup routines are provided by the manufacturer, they are transferred to a Columbus SW package and executed using the basic specifications.

Often, it is easier to carry out a packaging by using the Columbus snapshot technology, especially in those cases, when the MSI or Silent Setup routines do not contain certain configuration points or when the installation routine cannot be executed unattendedly / silently.

Configurations of applications are created and passed on with the Columbus snapshot technology.

The SW packages are created with a packaging template valid for the entire company.

The template can be found in the *Packaging Tools* under [Default Package].

3.2 Working with "Packaging Tools"

The term *Packaging Tools* comprises all programs and tools required for creating a SW package. In the project environment, each company will receive a SW-Paket, which contains a predefined SW packaging template.

The SW-Paket *902000_Columbus_PackagingTools.nnn.ENU* is supplemented by the corresponding abbreviation (Caution: identifier and description) and the package template contained will be adapted to the corresponding branch office, if required.

Contrary to the other applied guidelines, this SW-Paket will be modified in the main version and passed on to the branch offices (e.g. in case of new PackageStudio version).

3.3 Columbus script language

The script language used within the SW-Pakets (and also for OSDeploy jobs) is a mighty, yet easy to understand language. The grammar of the Columbus script language is similar to the proven batch file creation.

The commands and parameters are continuously enhanced in order to cope with the issues emerging on the market. Refer to the Command Help within Columbus PackageStudio after each update. The Columbus **Technical Reference** manual is used as documentation basis.

Ensure the correct usage of the script language, so that the SW-Pakets can be installed without errors.

3.4 Storage location of the installation source

Basically, a package directory is treated as a total package of an application. This package contains all necessary files and directories.

This applies for setup routines as well as for installation sources. Thus, it can be ensured that no important data will be lost or forgotten while distributing the SW-Pakets. Furthermore, this prevents changing the installation sources without prior change requests.

Note Installation sources are not stored on servers and shares without Columbus.

Within the SW-Paket, installation sources are stored in the following path:

#IDENTIFIER#\01.0\Server\Setup*.*

If an application cannot be installed via the network (access to %_PkgSource%), the delivery script offers an alternative by previously executing a copy procedure on a local storage.

```
Run '%_PkgSource%\Server\Setup\setup.exe' SHOW WAITDOWN TASK:5
```

3.5 Shortcuts

Basically, the original shortcuts of the setup routine are used.

If "Roaming Profiles" are used (user profiles stored on the server), the shortcuts for standard applications will be stored in the AllUser section.

3.6 Configuration of SW-Pakets

If configurations have to be defined variably (independent of site, license) in a SW-Paket, these specifications are summarized per SW-Paket and stored into a configuration file.

The configuration file is named according to the identifier of the related SW-Paket. (e.g. 177500_Adobe_Reader_10.cfg or 177500_Adobe_Reader_10**.BWS**.cfg)

The same is true for variable names (e.g. a_177500_Adobe_Acrobat_10_Serial or a_177500_Adobe_Acrobat_10**.BWS**_Serial).

3.6.1 Name convention of variables

In order to be able to clearly assign a variable and its value to one SW-Paket, the following basic codes for variable names are defined:

_	Are managed by Management Client.
a_	Is defined in a configuration file of an individual SW-Paket.
b_	Is defined by the basic configuration package.
c_	Is defined in the Management Console.
t_	Is used in a SW-Paket and deleted in the same section.

Depending on the application area, or if values have to be overwritten, a_ or b_ variables can be defined also within the Management Console.

3.6.2 Structure of a configuration file

If configurations have to be defined variably (independent of site, license) in a SW-Paket, these specifications are summarized per SW-Paket and stored into a configuration file.

```
; ===================================================================
; (c) Copyright by brainwaregroup
; ===================================================================
; Filename       : 177600_Adobe_Reader_10.cfg
; Version        : 1.0
; Change History          : None
; Created at     : 24.03.09 by : BWG/bdm
; Changed at     : 00.00.00 by :
; ===================================================================

; === Client Settings ================================================
[ClientSettings]
a_177600_Adobe_Acrobat_10_Serial=1234-5678-9100

; === User Settings ==================================================
[UserSettings]
a_177600_Adobe_Acrobat_10_Update=0
```

In the packaging template for ADVANCED and EXPERT you will find an example of such a configuration file under `#IDENTIFIER#\01.0\Config\_#PKGIDENTIFIER#.cfg`.

The configuration file can be stored either in a centralized or a decentralized storage.

Configuration files are only used for the ADVANCED and EXPERT platforms.

Centralized storage

A **centralized storage location** (see "**Variable values using the console structure**" on page 32) can be defined in environments with several Site Servers or where there is a need for a centralized control of SW-Pakets using the variable %C_ConfigPath%.

Decentralized storage

The configuration or control of a SW-Paket can also be decentralized by storing the configuration file in the SW-Paket.

The example of a configuration file is renamed and used as follows:
`#PKGIDENTIFIER#.cfg` > `410000_MS_Office_2010.cfg`

3.6.3 Variable values in a package

In order to create SW-Pakets which are independent of site and license, individual values within a SW-Paket will be replaced by a variable. These variables will be filled with valid values from the configuration file.

The following values in SW-Pakets are replaced by variables:

- Licensee (Registered Organization and Registered User)

- License number
- Machine type (Desktop, Notebook, Server)

3.7 Variables resolution - Columbus script language

The Columbus Script Interpreter triggers variables from left to right. Thus, nested variables are not possible.

Can be resolved

```
'%_PkgIdentifier%_License'
```

Read values from left to right	Resolved
%_PkgIdentifier%	"200000"
_License	"_License"
Effectively processed value:	200000_License

Cannot be resolved

```
'%a_%_PkgIdentifier%_License%'
```

Read values from left to right	Resolved
%a_%	" "
_PkgIdentifier	"_PkgIdentifier"
%_License%	" "
Effectively processed value:	_PkgIdentifier

3.8 Information on SW packages

In the moment of creating the SW packages, all specifications are defined, the network access is working and the order as well as the steps required for installation or packaging are known.

If a SW-Paket has to be modified at a later time, the above mentioned specifications are often missing.

Within the SW-Paket, additional information can be saved under Documents.

Thus, all the information can be saved for future tasks.

3.9 Repetitive sections

Individual SW packages can contain special sections which are repetitively executed.

In the delivery script, this is the ServerRepeat section; in the configuration script, these are the UserRepeat and the ClientRepeat sections.

In these special sections, the same Columbus script language is used as for the other sections.

3.9.1 Executing the sections

The repetitive sections are executed like their corresponding counterpart:

- ServerRepeat like ServerAdd (machine part)
- ClientRepeat like ClientAdd (machine part)
- UserRepeat like UserAdd (user part)

The repetitive sections are executed at any moment of a Columbus process call (push, machine start, user login, etc.).

During a user login in a terminal server environment, only the UserRepeat section is executed.

3.9.2 Creating the sections

If the sections are not included in the "Packaging Tools" template, they must be created manually.

In the script parts Delivery script and Configuration script, the new repetitive sections are created between the [Header] and the first [...Add] section.

3.9.3 Application examples

Section [ServerRepeat]

```
; === Start RemoteRegistry Service for Support ======================
Service 'RemoteRegistry' /Start /NoErrors

; === Change Password of local Administrator Account ================
ChangePW '' '%_UserAdministrator%' 'H#1234ABCD'
```

Section [ClientRepeat]: No action required

Section [UserRepeat]

```
; === Connect Networkdrive ==========================================
run 'net use w: \\SERVER\SHARE /PERSISTENT:NO' HIDE WAITDOWN TASK:5
```

3.10 Placeholder packages

In very rare cases, the application is not completely packaged. This can be due to the following reasons:

- The application or the setup routine must be activated manually
- Less than three installations are required
- Packaging would entail disproportionate costs

In such cases, a placeholder package is created which accurately matches the other SW packages having a valid identifier and a description.

The setup routine with installation instructions for manual installation can be included in the SW package.

The advantages of placeholder packages:

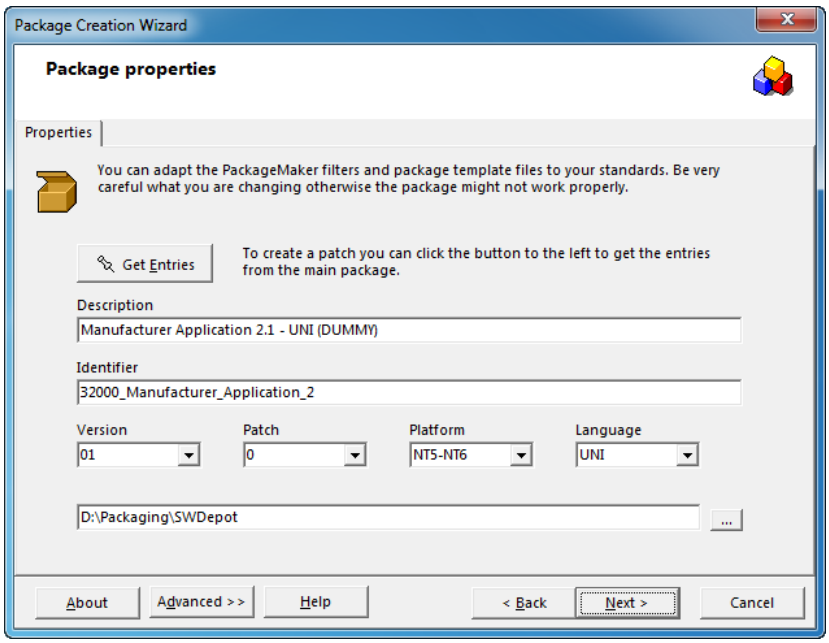
- When packaging is subsequently executed, the *wrapper* is already available
- The software to be installed on the computer is recognizable at a glance in the Management Console.

In order to distinguish such SW packages from the rest, the value (DUMMY) is entered in the description.

3.10.1 Name conventions for placeholder packages

The same conventions as for standard packages apply. A difference is only made on the description:

Manufacturer Product Version – Language (DUMMY)



3.10.2 Procedure for creating a placeholder package

- A new SW-Paket can be created and edited using the PackageStudio function **Create empty Package**.
- Adjust the data under **Edit Package Definition**.
- ... other operations as usual

CHAPTER 4

Variable values using the console structure

In this chapter

4.1 Information

The variables which can be created via the Management Console are used for the simple processing of identical SW-Pakets that have a different configuration.

In principle, the variable can be changed on any possible level. New variables are always created on the company level and are provided with a value on the required level (always inherited, so that parentage and availability are absolutely clear).

For reasons of clarity, the number of variables used in the Management Console should be limited to approx. 20. Configuration information on applications (e.g. TwixTel Server or Exchange Server) will not be treated as a variable in the Management Console but will be externally stored as a configuration file.

Only variables that will be used are set.

Note	This chapter only applies to SW-Pakets with the ADVANCED or EXPERT complexity.

4.2 Short summary

Variable	Short description	Possible values	Usage
c_ConfigPath	Path for the configuration files	\\Server\Share\Config \%c_Environment%	ADVANCED or EXPERT if centralized storage is required.
c_MachineType	Machine type	NOTEBOOK, WORKSTATION, SERVER	EXPERT
c_MachineLocation	Machine location	BERN	EXPERT
c_MachineLanguage	Machine language	DEU, ITA, FRA, ENU, JPN	EXPERT
c_MachineDep	Machine department	Accounting	EXPERT
c_UserDep	User department	Marketing	EXPERT
c_UserLanguage	User language	DEU, ITA, FRA, ENU, JPN	EXPERT

4.3 Detailed explanation per variable

4.3.1 c_ConfigPath

Definition

- Path to the configuration files, with regard to the environment.

Usage

- Always, if a distinction between development and production is required
- Geographical distinction

Contents

- \\SERVER\Share\Config\%c_Environment% Folder on the main Columbus server
- \\SERVER\Columbus\Config\%c_Environment% Arbitrary server share

4.3.2 c_MachineType

Definition

- To distinguish among machine types. Depending on the function of a machine.

Usage

- If, depending on the function, different configurations are required.
- e.g. On Notebooks, the virus scanner shall connect directly to the manufacturer's update site.
- e.g. no reboot after installation on servers

Contents

- Workstation Fixed work place
- Notebook Mobile work place
- Server Server or server functionality
- Citrix Citrix or terminal server

4.3.3 c_MachineLocation

Definition

- Location of the machine

Usage

- If, depending on the location, different configurations are required.
- e.g. Exchange Server in Bern or Zurich

Contents

- BE Bern
- LBST09 Längenbühlstrasse 9
- HH High-rise building
- Werkhalle 2 Second workshop

4.3.4 c_MachineLanguage

Definition

- Primary language of the machine

Usage

- For applications which do not allow for different languages settings per user
- For an easier handling of single-language environments, this variable can be set to *Company* level; thus, the machines do not have to be moved to corresponding sites, but can stay in the general container *computers*.

Contents

- DEU German International
- ENU English International
- FRA French International
- ITA Italian International
- JPN Japanese International

4.3.5 c_MachineDep

Definition

- Department of the machine

Usage

- If, depending on the department, different configurations are required.
- e.g. shortcut on desktop on all "workshop" machines.

Contents

- Workshop Machine in the workshop department
- Marketing Machine in the marketing department

4.3.6 c_UserDep

Definition

- Department of the user

Usage

- If, depending on the department, different configurations are required.
- e.g. shortcut on desktop on all "workshop" machines.

Contents

- Workshop User in the workshop department
- Marketing User in the marketing department

4.3.7 c_UserLanguage

Definition

- Primary language of the user

Usage

- For applications which do allow for different language settings per user
- For an easier handling of single-language environments, this variable can be set to *Company* level; thus, the users do not have to be moved to corresponding sites, but can stay in the general container *Users*.

Contents

- DEU German International
- ENU English International
- FRA French International
- ITA Italian International
- JPN Japanese International

4.4 Variables per level

4.4.1 On company level

Variable	Possible values	Assign
c_ConfigPath	**\\Server\Share\Config\%c_Environment%**	**Client**
c_MachineTyp	{LEER}	Client
c_MachineLocation	{LEER}	Client
c_MachineLanguage	**{LEER} ***	**Client**
c_MachineDep	{LEER}	Client
c_UserDep	{LEER}	User
c_UserLanguage	**{LEER} ***	**User**

* The language variables may be set uniformly on previous levels.

4.4.2 On location level

Variable	Possible values	Assign
c_ConfigPath	>> inherited	Client
c_MachineTyp	{LEER} >> inherited	Client
c_MachineLocation	**BE, FR, ZG**	**Client**
c_MachineLanguage	{LEER} >> inherited *	Client
c_MachineDep	{LEER} >> inherited	Client
c_UserDep	{LEER} >> inherited	User
c_UserLanguage	{LEER} >> inherited *	User

* The language variables may be set uniformly on previous levels.

4.4.3 On department level

Variable	Possible values	Assign
c_ConfigPath	>> inherited	Client
c_MachineTyp	{LEER} >> inherited	Client
c_MachineLocation	>> inherited	Client
c_MachineLanguage	{LEER} >> inherited *	Client
c_MachineDep	**Accounting, Support**	**Client**
c_UserDep	**Accounting, Support**	**User**
c_UserLanguage	{LEER} >> inherited *	User

* The language variables may be set uniformly on previous levels.

4.4.4 On machine type level

Variable	Possible values	Assign
c_ConfigPath	>> inherited	Client
c_MachineTyp	**Workstation, Notebook, Server, Citrix**	**Client**
c_MachineLocation	>> inherited	Client
c_MachineLanguage	**DEU, FRA, ITA, ENU, JPN ***	**Client**
c_MachineDep	>> inherited	Client
c_UserDep	>> inherited	User
c_UserLanguage	{LEER} >> inherited *	User

* The language variables may be set uniformly on previous levels.

4.4.5 On user type level

Variable	Possible values	Assign
c_ConfigPath	>> inherited	Client
c_MachineTyp	{LEER} >> inherited	Client
c_MachineLocation	>> inherited	Client
c_MachineLanguage	{LEER} >> inherited	Client
c_MachineDep	>> inherited	Client
c_UserDep	>> inherited	User
c_UserLanguage	**DEU, FRA, ITA, ENU, JPN ***	**User**

* The language variables may be set uniformly on previous levels.

4.5 Variable values of Management Client

The used Management Client offers other locally valid variables. e.g.: version of the operating system, existence of Windows components or version of the installed Microsoft Office Suite. You will find the variables with related values in the registry.

Variables to be used in all script parts (Server, Client and User):

```
HKEY_LOCAL_MACHINE\SOFTWARE\Brainware\Variables\Static
```

Variables to be used in the user part (User) only:

```
HKEY_CURRENT_USER\SOFTWARE\Brainware\Variables\Static
```

CHAPTER 5

Log and configuration files

In this chapter

Log files contain detailed information about success or failure of a setup routine. A log file is created for practically every action; this file is stored in a central location on the local computer.

5.1 Creating log files

If possible, log files will be written. This facilitates the tracking of installation or operation issues of applications.

The variable %_PkgIdentifier% is used to be able to clearly assign the log file to a SW-Paket. This ensures that the identifier which is valid in each case is used as file name:

```
"%_PkgCache%\%_PkgIdentifier%-INSTALL.Log"
```

Especially in case of MSI installation routines, log files will be created:

```
MSI '%_PkgSource%\Server\Setup\AcroPro.msi' 'ACTION="INSTALL"'
/LOG:"%_PkgCache%\%_PkgIdentifier%-INSTALL.Log" /NONE
```

Log files will be created separately for each MSI setup routine. If more than one .msi is executed in a SW-Paket, a log file will be created for each setup instance.

```
MSI '%_PkgSource%\Server\Setup\AcroSP1.msi' 'ACTION="INSTALL"'
/LOG:"%_PkgCache%\%_PkgIdentifier%-SP1-INSTALL.Log" /NONE
```

The name of the log file will be supplemented by the instance, either by a sequential number or a unique identifier.

A log file is also written and named accordingly, when the SW-Paket is removed.

```
MSI '%_PkgSource%\Server\Setup\AcroPro.msi' 'REMOVE="ALL"'
/LOG:"%_PkgCache%\%_PkgIdentifier%-UNINSTALL.Log" /NONE
```

5.2 Storage location of log files

The log files are stored on each client locally in the cache of the related SW-Paket. The storage location is defined by the variable %_PkgCache%.

The log files remain on the computer and will not be deleted, if the SW package is removed. Thus, it is possible to track at anytime, which setup routine was executed when and how.

5.3 Storage location of configuration files

In the packaging template for ADVANCED and EXPERT you will find an example of such a configuration file under #IDENTIFIER#\01.0\Config_#PKGIDENTIFIER#.cfg.

The configuration file can be stored either in a centralized or a decentralized storage.

Configuration files are only used for the ADVANCED and EXPERT platforms.

Centralized storage

A **centralized storage location** (see "**Variable values using the console structure**" on page 32) can be defined in environments with several site servers or where there is a need for a centralized control of SW packages using the variable %C_ConfigPath%.

Decentralized storage

The configuration or control of a SW package can also be decentralized by storing the configuration file in the SW package.

The example of a configuration file is renamed and used as follows:

`#PKGIDENTIFIER#.cfg` > `41000_MS_Office_2010.cfg`

Example

```
; === Process Configfile ============================================
; If file '%_PkgSource%\Config\%_PkgIdentifier%.cfg' then copy
'%_PkgSource%\Config\%_PkgIdentifier%.cfg' to '%_PkgCache%\%_PkgIdentifier%.cfg'
; If file '%c_ConfigPath%\%_PkgIdentifier%.cfg' then copy
'%c_ConfigPath%\%_PkgIdentifier%.cfg' to '%_PkgCache%\%_PkgIdentifier%.cfg'
; If not file '%_PkgCache%\%_PkgIdentifier%.cfg' then exit 'ServerAdd: Missing
"%_PkgCache%\%_PkgIdentifier%.cfg"'
; Load Variables 'ClientSettings' '%_PkgCache%\%_PkgIdentifier%.cfg' /Volatile /Immediate
; If '%a_51050_7-zip_Packager_9_Value%' = '' then exit 'ServerAdd: Missing %%a_51050_7-
zip_Packager_9_Value%%'
```

5.4 Language setting of SW packages

The language is defined by a variable (%c_UserLanguage%) on the user level

The language is valid for one user. This configuration allows the use of different user languages on one computer.

If an application can only be installed for one language (no multi-language support), the corresponding language is selected in the machine variable named c_MachineLanguage.

If the application does not support all company languages, the default selection within a SW-Paket will be English.

Ex. Italian is not available:

```
If '%c_MachineLanguage%' = 'ENU' then run 'setup_english.exe'
```

CHAPTER 6

Software dependencies

In order to check whether a certain application exists, the .exe file, a .dll and / or the registry is directly checked in the ServerAdd part of a SW package.

Also, the static variables of Columbus can be used. The detailed variables for .net or other Windows components are only available with a PatchDeploy license.

Example

```
; = Windows Installer 4.5
; If '%_MSIVersion%' < '0004.0005.0000.0000' then exit 'ServerAdd: Missing Windows
Installer 4.5'

; = MS C++ 2005 Runtime
```

Note	For a consistent design of the SW release and in order to simplify a possible troubleshooting, the queries are only executed in the ServerAdd section.

Important	There will be no queries (except for the base and/or final package) which can be resolved by individual SW packages using a set variable.

Within such a comprehensively and widely used platform, it would not make much sense to create the component check (e.g. Internet Explorer) via a variable set by a SW package; thus, the SW package set by the variable would have to be installed in any case, despite the fact that the required component may already be part of the Windows installation files.

This causes even more problems, if different operating system versions are used within one platform (e.g. XP, 2003, Vista and 2008).

CHAPTER 7

Setup routines

In this chapter

The Windows Installer provides for a runtime environment for installation routines.

This system service can process installation routines in the form of MSI and MSP (patch) files.

Aside from Microsoft, the Installshield company is the biggest producer of MSI setup routines. Installshield works with its own script language and integrates it in the MSI setup routines. For the execution of setup routines created with Installshield, an InstallScript framework is required on the target computer.

This InstallScript framework changes from time to time and must therefore also be checked. Often, InstallShield setup routines cannot be installed because the framework of the target machine is obsolete or faulty. It is recommended to check the setup routine as well as the framework and to distribute, if required, the InstallScript Framework as a separate SW-Paket.

If you believe in the marketing documents, you may think that the processing or the distribution of MSI based SW-Paket is a piece of cake. In fact, also MSI setup routines have to be tested, re-defined or completely revised.

The configuration of setup routines can be done with so-called MST files; however, these files show some disadvantages, which can be critical in some cases:

- They can be created only with additional efforts and corresponding tools.
- Only the options that were provided by the manufacturer can be used.
- They are very difficult to edit later on.
- They cannot be modified during the installation.
- Higher effort, if customer-specific configurations should be passed on.
- They configure only the setup routine, but not the installed application.

Therefore, it is recommended to use the setup routine (MSI) released by the manufacturer directly and install it using standard parameters. Then, a comfortable and proven Columbus script routine can be used to carry out the desired configuration of an application.

Thus, the practical MSI installation routines and, if available, the repair routines, combined with the clearly defined Columbus script language, can be merged into a very good and comprehensive SW-Paket.

7.1 Columbus's own MSI command

MSI setup routines are mainly executed with the Columbus's own MSI command:

```
MSI '%_PkgSource%\Server\Setup\xyz.msi' 'REBOOT="ReallySuppress"'
/LOG:"%_PkgCache%\%_PkgIdentifier%-INSTALL.log" /NONE
```

Many MSI setup routines dispose of a repair function, which may be used for the re-installation of a SW-Paket.

First, it is checked whether the MSI setup routine is already installed; depending on the result, it is either completely installed or only the repair function is used:

```
If Installed 'MSIPRODUCT' '{74F6BA49-8345-44E6-A4E7-801497CC7C10}' then goto MSIReInst

MSI '%_PkgSource%\Server\Setup\xyz.msi' 'REBOOT="ReallySuppress"'
/LOG:"%_PkgCache%\%_PkgIdentifier%-INSTALL.log" /NONE

goto EndMSICheck

:MSIReInst
MSI '%_PkgSource%\Server\Setup\xyz.msi' 'oumsv' /REPAIR
/LOG:"%_PkgCache%\%_PkgIdentifier%-REINSTALL.log" /NONE

:EndMSICheck
```

Columbus's own MSI command evaluates the MSI error codes and informs the database about success or failure of the MSI installation. If the MSI setup routine could not be carried out correctly, the MSI's own error code will be displayed in the Management Console as error status.

7.2 With Windows Installer

If Columbus's own MSI command is unable to handle the installation routine, the MSI setup routine is executed by directly starting the Windows Installer.

```
Run 'msiexec /I "%_PkgSource%\Server\Setup\xyz.msi" REBOOT="ReallySuppress" /qn /L*v
"%_PkgCache%\%_PkgIdentifier%-INSTALL.log"' Show Waitdown Task:30
```

Many MSI setup routines dispose of a repair function, which may be used for the re-installation of a SW-Paket.

First, it is checked whether the MSI setup routine is already installed; depending on the result, it is either completely installed or only the repair function is used:

```
If Installed 'MSIPRODUCT' '{74F6BA49-8345-44E6-A4E7-801497CC7C10}' then goto MSIReInst

Run 'msiexec /I %_PkgSource%\Server\Setup\xyz.msi" REBOOT="ReallySuppress" /qn /L*v
"%_PkgCache%\%_PkgIdentifier%-INSTALL.log' Show Waitdown Task:30
GOTO EndMSICheck

:MSIReInst
Run 'msiexec /Foums "%_PkgSource%\Server\Setup\xyz.msi" /qn /L*v
"%_PkgCache%\%_PkgIdentifier%-REINSTALL.log "' Show Waitdown Task:30

:EndMSICheck
```

The above method does not verify whether the MSI setup routine has been carried out successfully. It can happen that a successful installation of the SW-Paket is displayed in the Management Console, although the application has not been installed at all.

In order to be able to check for errors when using the above installation method, the status is determined after the completion of the Windows Installer by checking the return value (ErrorLevel):

```
Run 'msiexec /I "%_PkgSource%\Server\Setup\xyz.msi" REBOOT="ReallySuppress" /qn /L*v
"%_PkgCache%\%_PkgIdentifier%-INSTALL.log ' Show Waitdown Task:30

If '%_ErrorLevelRun%' = '0' then goto NOERR
If '%_ErrorLevelRun%' = '3010' then goto NOERR
exit 'ServerAdd: Error "%_ErrorLevelRun%" occurred'
:NOERR
```

This check must be created for each called MSI setup routine in a SW-Paket; accordingly, the jump label NOERR has to be incremented to get the correct value for each check.

7.3 Windows Installer error codes

The MSI error codes apply either for the Columbus MSI command or for the direct call via Windows Installer. If the MSI setup routine is completed without error, **0** is returned.

Frequent error codes

Error code	Description
0	No errors
3010	System restart required
1603	Unknown, fatal error
1618	Another installation is already running
1619	MSI file cannot be called

For Information about other MSI error codes, refer to this web address: msdn.microsoft.com/en-us/library/aa368542.aspx

7.4 Other setup routines

Apart from the known MSI setup routine, you can also find other setup manufacturers and their products:

- Installshield
- InnoSetup
- Nullsoft
- Astarum

Also these setup routines often return a status in form of a value:

```
Run '%_PkgSource%\Server\Setup\setup.exe /s /l %_PkgCache%\%_PkgIdentifier%-INSTALL.log'
Show Waitdown Task:30

If '%_ErrorLevelRun%' = '0' then goto NOERR
If '%_ErrorLevelRun%' = '3010' then goto NOERR
exit 'ServerAdd: Error "%_ErrorLevelRun%" occurred'
:NOERR
```

The calls for silent or unattend methods vary depending on the setup manufacturers and sometimes even within a setup routine community.

Known parameters are:

- /s or /silent or /verysilent
- /u or /unattend
- /q or /quiet

Note Observe upper/lower case.

For further information about parameters, please contact the application or setup routine manufacturer.

7.5 Reboot on demand

A system restart that depends on the Return code can be executed as follows:

```
Run '%_PkgCache%\Client\setup.exe /s /l%_PkgCache%\INSTALL.log' Show Waitdown Task:30

If '%_ErrorLevelRun%' = '0' then goto End
If '%_ErrorLevelRun%' = '3010' System Reboot
If '%_ErrorLevelRun%' = '3010' then goto End
exit 'ClientAdd: Error "%_ErrorLevelRun%" occurred'
:End
```

CHAPTER 8

QA - Package Engineer

In this chapter

8.1 General

QA contents of the individual Package Engineer

These defaults are continuously renewed and enhanced to give the Package Engineer further support for an easy quality assurance. These defaults can serve as a check list, but they are not concluding.

Name convention

- Identifier created according to specification and language code observed
- NUMBER_MANUFACTURER_PRODUCT_VERSION
- NUMBER_MANUFACTURER_PRODUCT_VERSION.ABBREVIATION
- Language codes of the SW-Pakets observed.
- Package directory is IDENTIFIER.LANGUAGE
- Main package has the number 01.0
- Other patches are listed with 0100, 0200, etc.

Structure of SW package

- Query of the preconditions in ServerAdd
- Removing the _PkgCache files in ClientRemove
- No fixed specifications used (variables set instead)
- Dependencies to other SW packages or applications observed

SW package - Remove

- Remove of the application completed.
- Log files are preserved (for conclusion in case of problems)

Variables

- Standard variables used (%_Windows%, %_SystemDrive%, etc.)
- Global variable used (%c_Machinetype%, %c_UserLanguage%, etc.)
- Temporary variables deleted in the corresponding section (%t_ColData%, etc.)
- Company name and user names replaced by variable (RegOwner, RegUser)
- 8.3 variable set

Script language

- Columbus script command syntax observed.

ini files or similar configuration files

- Contents checked for fixed paths and replaced with text file /Replace

Shortcuts

- Shortcuts written in UserAdd and ClientAdd
- Shortcuts removed in UserRemove and ClientRemove
- If shortcut groups are removed, check for other existing shortcuts.
- Possibly, the directory contains other shortcuts of similar programs

Documentation

- Info.rtf correctly filled in (description, version, specification for *.cfg file, indicate pre-requests, regperm and xcacls documented,
- *.cfg created and completed, if required
- SW-Paket prerequisites transferred from ServerAdd in the documentation
- Query of the supported platforms (WORKSTATION, SERVER, etc.) transferred

Technical

- Miscellaneous

Functionality

- Does the application work with limited user rights?
- Does the application work in another environment (hardware, operating system)?
- Are the defined settings used?
- Are there any errors in the application?

8.2 SW-Paket self-test - Procedure

SW-Pakets that are made available for the QA must have been previously tested. These tests comprise also the correct representation and annotation of the Columbus script.

The sender of a SW-Paket which is received by QA is held responsible for its quality. This applies also, if a SW-Paket has *only* been corrected by the sender.

98% of the SW-Paket can be tested on a V machine without negative side effects on the test results. There are some SW-Pakets which are HW dependent or require additional peripherical devices (USB, serial).

Tests on virtual machines can be carried out very easily thanks to the snapshot function of VMware. The *Reset* option should be used very often.

However, the V machine must be reinstalled frequently, in order to make use of the most up-to-date OS release or basic configurations.

The following procedure is recommended:

Installation of the required operating system on a test machine

- MS Windows 7 SP1 English
- MS Windows 2008 SP1 English

Installation of minimum basic packages

- (e.g. Base-Config, Windows MUI, Final-Config)

Assigning the SW-Paket

- Are missing applications or components displayed? (.net or MS Office)

Adding missing components

- Assign missing components (one after another, always check between installations)

Reset machine, change user (default user, no admin rights)

- No user login
- Wait for the status message in the Management Console.
- Login with limited user, checking the results

Re-installation of the SW package

- With logged-in user
- Without logged-in user

Reset of the machine, changing the machine language

- Same procedure as above
- It is recommended to provide one V machine per language

Reset of the machine, changing the user language

- Same procedure as above
- It is recommended to provide one user account per language

Reset of the machine, assigning the default SW group

- Same procedure as above
- Check compatibility

8.3 SW package self-test - Frequent errors

The test procedure above is a proposition, which may vary according to the SW package or the applications involved. Depending on the experience of the Package Engineer, he can use his own test routines.

The following errors and their consequences occur frequently:

One computer for all

- Test on the same computer or snapshot
- Previous setup faults
- Unpredictable test results

Assigning the default SW group (prior to beginning the test)

- No checking of possibly missing components

Test only with administrative account

- No statement regarding end user

Test only with logged-in user

- SW packages must be installable without logged-in user.

CHAPTER 9

Info.rtf template

Description

This package installs:

- Description: <Content Description>
- Current version: <Version Nr.>
- Language: <Languages>

Components

- All

Prerequisites

- None

Configuration

- No *.cfg required

Reboot/Logouts

- ServerAdd: [default]
- ClientAdd: [default]
- UserAdd: [default]

Definitions

- Conditional Use: Yes
- Repeat Sections: No

Remarks

- None

Known Problems

- None

Creator

- Created 00.00.0000 by: PACKAGE ENGINEER
- Changed 00.00.0000 by: PACKAGE ENGINEER

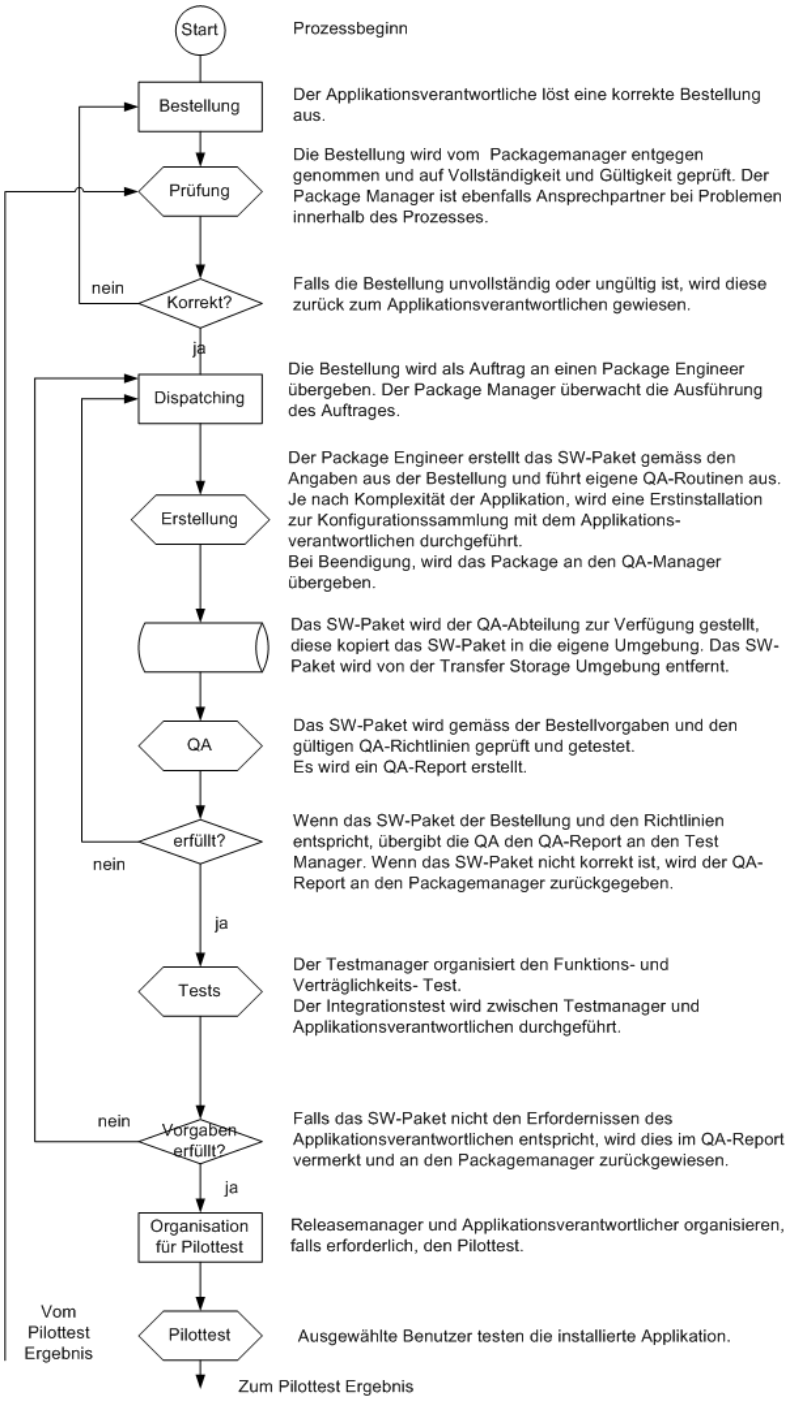
CHAPTER 10

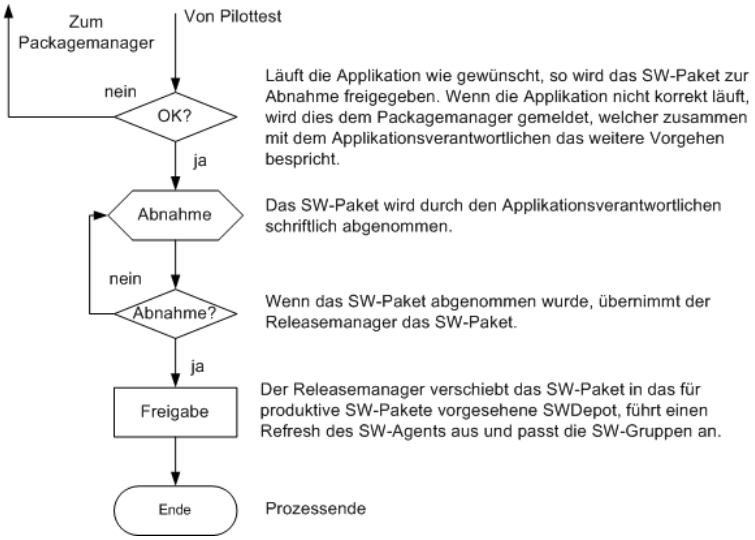
SW release management

In this chapter

Note	Is treated in a separate document.

10.1 Process graphic (example)





10.2 Process roles

Process role	Description
Package Manager	• First contact for questions or trouble • Monitors the packaging process • Distributes and monitors the packaging orders • Contact of the Packaging Engineer
Application supervisor	• Prepares the specification for the creation of the SW package (the technical part of a SW package order) • Accepts the SW package in written form, before it is transferred into the productive environment by the Release Manager.
Release Manager	• Monitors the SW release status • Defines time and date for releases • Is copied on all information regarding the process
Package Engineer	• Creates SW packages according to order and applicable guidelines. • Makes component and system tests. • Works based on 4-eye-principle
QA Manager	• Checks and tests the SW package, but not the application • Does not modify the SW package. • Creates the QA report • Gives recommendations • Reports to the Package Manager
Test Manager	• Checks and tests the application • Does not modify the SW package. • Edits the QA report • Reports to the Package Manager and Application Supervisor
Pilot test user	• Selected users which test the application thoroughly during the pilot test. • Report to the int. QA Manager and the Application Supervisor

CHAPTER 11

Glossary

In this chapter

11.1 Abbreviation

Location	Abbreviation
Geneva	GVA
Lausanne	LAU
Columbus Training	CTR
brainwaregroup	BWG

11.2 Terms

Term	Description
brainwaregroup	Brainware is the manufacturer of Columbus Software Management (SwM) and comprises the divisions Development and Consulting. The Columbus product range is mainly developed in Switzerland. Further information: www.brainwaregroup.com
Columbus	Generic term for the software management application
Management Console	Control unit of the Infrastructure Service
Management Client	Installed instance on a computer. Needed for installation / de-installation, inventory, patch management, etc.
Infrastructure Service	Server instance with several agents and the Columbus Database.
PackageMaker	Application for creation of SW-Pakets using before and after snapshots which then will be processed in PackageStudio.
PackageStudio	Application for processing existing Columbus SW-Pakets.
VMware	VMware or the VMware WorkStation application offers the possibility to install Windows and other operating systems on a virtual machine which allows creating SW-Pakets. Further information: **www.vmware.com** (**http://www.vmware.com**)