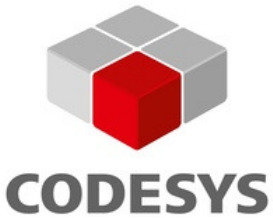


OEM Specification:
Packages

Document Version 7.0

tech_doc_e.doc / V1.2

CONTENT

1 Overview

Using the Package Management architecture is the preferred way of extending an existing CODESYS installation (or an installation of any other Automation Platform application) by additional features and configuration settings. The concepts behind packages are quite similar to those of typical Windows installer mechanisms. However, the Package Management is tailored to the Automation Platform architecture in an optimal way, which is – according to experience – not always the case with standard installers.

1.1 Main Features

The main features of the Package Management include:

- Installing package files to CODESYS or any other Automation Platform application. Different versions of the same packages can be installed side by side.
- Downloading packages file from a web-based server and installing it directly. ***Note:*** *This feature is currently not implemented yet.*
- Uninstalling packages that have been installed before.
- Listing all packages that have been installed.
- The Package Manager itself can be invoked by menu within CODESYS or as a standalone application with command line interface.

1.2 Package Contents

The following items can be part of a package:

- Plug-Ins
- Libraries
- Device descriptions
- Vendor descriptions
- Profiles
- Informational profiles
- Profile changes
- Files
- Additions to the menu configuration
- Additions to the toolbar configuration
- Additions to the shortcut configuration
- Additions to the view configuration
- Online help modules
- Entire menu configurations for a specific profile
- Entire toolbar configurations for a specific profile
- Entire keyboard configurations for a specific profile
- Options
- Library profiles

2 Package Files

This chapter describes the format a package file and its parts.

2.1 General

A package is a ZIP file with the file extension `.package`. It is required that this ZIP archive contains a top-level file named `package.manifest`. The format of this file is described in this chapter. All the other data that must be carried along with the package (binaries, files, icons, etc.) must also be part of that ZIP archive; the locations of those files and the folder structure within the archive is arbitrary.

2.2 package.manifest File Reference

The `package.manifest` file is an XML file that must conform to the schema that is described in this section.

2.2.1 Alphabetical Element List

2.2.2 Element Descriptions

Within the element descriptions, the cardinality of each child element is specified as follows:

- [0..1] The child element is optional and may appear once at maximum.
- [0..*] The child element is optional and may appear multiple times as well.
- [1] The child element is required and must appear exactly once.
- [1..*] The child element is required and may appear multiple times.

2.2.2.1 Package Element

Required root element of a package.manifest file.

```
<Package>
   <Strings>…</Strings>
   <General>…</General>
   <TargetDirectoryDefinitions>…</TargetDirectoryDefinitions>
   <Components>…</Components>
</Package>
```

Parent element: none

Attributes: none

Child elements:

Element	Description
Strings	[0..1]

Element	Description
	Contains localized string definitions. At any place in the package.manifest file where a user interface relevant string can be specified, there might be a reference into that section. The effective text is then chosen dependent on the current culture settings.
General	[1] Contains global information about the package.
TargetDirectoryDefinitions	[0..1] Some items in a package can be installed to a folder location that can be configured by the user. Those folder locations are described within this section.
Components	[1] Contains information about the groups of items that are installed by this package. A component can be activated or deactivated by the user, and required dependencies between components can be specified.

2.2.2.2 Strings Element

Contains localized string definitions. At any place in the package.manifest file where a user interface relevant string can be specified, there might be a reference into that section. The effective text is then chosen dependent on the current culture settings.

```
<Strings>
   <String Id="…">…</String>
   …
</Strings>
```

Parent element : Package

Attributes : none.

Child elements:

Element	Description
String	[0..*] Denotes a single localized string definition.

2.2.2.3 String Element

Denotes a single localized string definition.

At runtime, the lookup order is as follows: The current UI culture is determined (e.g. es-BR). First, a Localized element with culture es-BR is searched. If such an element is found, the string defined there is used. If not, the Localized element for the parent culture (here: es) is searched. If such an element is found, the string defined there is used. If not, the last step is repeated until the neutral culture is reached. In that case, the string specified in the Neutral element is used.

```
<String Id="…">
   <Neutral>…</Neutral>
   <Localized Culture="…">…</Localized>
   …
</String>
```

Parent element: Strings

Attributes:

Attribute	Description
Id	Required attribute. This is a unique identifier for the string.

Attribute	Description
	At any place in the package.manifest file where a user interface text can be specified, there are two options: either a normal text is specified, which is then used no matter which language the installer is currently running, or a string in the format `$xxx` is specified, which will look up the corresponding entry in this string table which has `Id="xxx"`.

Child elements:

Element	Description
Neutral	[1] Contains the default text that will be used when a lookup within the Localized elements was not successful.
Localized	[0..*] Contains the text for a specific language. It is not allowed that two Localized elements within the a String element occur for the same culture.

2.2.2.4 Neutral Element

Contains the default text that will be used when a lookup within the Localized elements was not successful.

```
<Neutral>…</Neutral>
```

Parent element: String

Attributes: none

Child elements: none

2.2.2.5 Localized Element

Contains the text for a specific language. It is not allowed that two Localized elements within the a String element occur for the same culture.

```
<Localized Culture="…">…</Localized>
```

Parent element: String

Attributes:

Attribute	Description
Culture	Required attribute. The language of this Localized element. This must be a valid culture specifier (e.g. `zh-CHS`, `de-DE`, or `en`).

Child elements: none

2.2.2.6 General Element (Below Package Element)

Contains global information about the package.

```
<General>
   <Id>…</Id>
   <Version>…</Version>
   <Name>…</Name>
   <Vendor>…</Vendor>
   <Copyright>…</Copyright>
   <Description>…</Description>
   <Icon>…</Icon>
   <HTML>…</HTML>
   <RequiredInstallerVersion>…</RequiredInstallerVersion>
   <LicenseAgreement>…</LicenseAgreement>
   <ReadMe>…</ReadMe>
</General>
```

Parent element: Package

Attributes: none

Child elements:

Element	Description
Id	[1] A GUID (**g**lobally **u**nique **id**entifier) that uniquely identifies this package. Multiple versions of packages with the same Id may exist.
Version	[1] The version of this package. This string must be in one of the following formats: • major.minor • major.minor.build • major.minor.build.revision where major, minor, build, and revision are numbers between 0 and 255 each.
Name	[1] The human-readable name of the package. This string can be prefixed with `$`; in that case, a localized string will be looked up in the string table.
Vendor	[0..1] The vendor of the package. This string can be prefixed with `$`; in that case, a localized string will be looked up in the string table.
Copyright	[0..1] Copyright information of the package. This string can be prefixed with `$`; in that case, a localized string will be looked up in the string table.
Description	[0..1] A short description about the package and its contents. This string can be prefixed with `$`; in that case, a localized string will be looked up in the string table.
Icon	[0..1] Path to an icon file (.ico file format) that will be associated with the package. The path must point to a file within the package archive and must be specified relatively to the package.manifest file.
HTML	[0..1] Path to an HTML file that contains more detailed information than within the Description element. The path must point to a file within the package archive and must be specified relatively to the package.manifest file. It can

Element	Description
	be prefixed with $; in that case, a localized path will be looked up in the string table, thus supporting multiple HTML pages in different languages.
LicenseAgreement	[0..1] Path to an HTML file that contains the end user license agreement (EULA) for the package. If specified, the user must explicitly confirm this agreement before s/he can continue with the installation. The path must point to a file within the package archive and must be specified relatively to the package.manifest file. It can be prefixed with $; in that case, a localized path will be looked up in the string table, thus supporting multiple license agreements in different languages.
RequiredInstallerVersion	[0..1] Indicates the minimum assembly version of the PackageManagement plugin that is required to execute this package. The package cannot be installed with a package engine that is too old according to this specification. This string must be in one of the following formats: • major.minor • major.minor.build • major.minor.build.revision where major, minor, build, and revision are numbers between 0 and 255 each.
ReadMe	[0..1] Path to a TXT or RTF file that contains the read me for the package. The path must point to a file within the package archive and must be specified relatively to the package.manifest file. It can be prefixed with $; in that case, a localized path will be looked up in the string table, thus supporting multiple read me in different languages. The read me file will be displayed after the installation has been executed successfully.

2.2.2.7 Id Element (Below General Element)

A GUID (**g**lobally **u**nique **id**entifier) that uniquely identifies this package. Multiple versions of packages with the same Id may exist.

```
<Id>…</Id>
```

Parent element: General (below Package)

Attributes: none

Child elements: none

2.2.2.8 Version Element

The version of this package, or the version of a plug-in component. This string must be in one of the following formats:

- major.minor
- major.minor.build
- major.minor.build.revision

where major, minor, build, and revision must be non-negative 32 bit values each.

```
<Version>…</Version>
```

Parent element: General (below Package), MinimumPlugInVersion, ProfileChange

Attributes: none

Child elements: none

2.2.2.9 Name Element

The human-readable name of the package, component, target directory definition, or popup menu item. This string can be prefixed with $; in that case, a localized string will be looked up in the string table.

```
<Name>…</Name>
```

Parent element: General (below Package), General (below Component), TargetDirectoryDefinition, Popup

Attributes: none

Child elements: none

2.2.2.10 Vendor Element

The vendor of the package. This string can be prefixed with $; in that case, a localized string will be looked up in the string table.

```
<Vendor>…</Vendor>
```

Parent element: General (below Package)

Attributes: none

Child elements: none

2.2.2.11 Copyright Element

Copyright information of the package. This string can be prefixed with $; in that case, a localized string will be looked up in the string table.

```
<Copyright>…</Copyright>
```

Parent element: General (below Package)

Attributes: none

Child elements: none

2.2.2.12 Description Element

A short description about the package, component, or target directory definition. This string can be prefixed with $; in that case, a localized string will be looked up in the string table.

```
<Description>…</Description>
```

Parent element: General (below Package), General (below Component), TargetDirectoryDefinition

Attributes: none

Child elements: none

2.2.2.13 Icon Element

Path to an icon file (.ico file format) that will be associated with the package. The path must point to a file within the package archive and must be specified relatively to the package.manifest file.

```
<Icon>…</Icon>
```

Parent element: General (below Package), General (below Component)

Attributes: none

Child elements: none

2.2.2.14 HTML Element

Path to an HTML file that contains more detailed information than within the Description element. The path must point to a file within the package archive and must be specified relatively to the package.manifest file. It can be prefixed with `$`; in that case, a localized path will be looked up in the string table, thus supporting multiple HTML pages in different languages.

```
<HTML>…</HTML>
```

Parent element: General (below Package)

Attributes: none

Child elements: none

2.2.2.15 RequiredInstallerVersion Element

Indicates the minimum assembly version of the PackageManagement plugin that is required to execute this package. The package cannot be installed with a package engine that is too old according to this specification. This string must be in one of the following formats:

- major.minor
- major.minor.build
- major.minor.build.revision

where major, minor, build, and revision are numbers between 0 and 255 each.

```
<RequiredInstallerVersion>…</RequiredInstallerVersion>
```

Parent element: General (below Package)

Attributes: none

Child elements: none

2.2.2.16 LicenseAgreement Element

Path to an HTML file that contains the end user license agreement (EULA) for the package. If specified, the user must explicitly confirm this agreement before s/he can continue with the installation. The path must point to a file within the package archive and must be specified relatively to the package.manifest file. It can be prefixed with `$`; in that case, a localized path will be looked up in the string table, thus supporting multiple license agreements in different languages.

```
<LicenseAgreement>…</LicenseAgreement>
```

Parent element: General (below Package)

Attributes: none

Child elements: none

2.2.2.17 TargetDirectoryDefinitions Element

Some items in a package can be installed to a folder location that can be configured by the user. Those folder locations are described within this section.

```
<TargetDirectoryDefinitions>
   <TargetDirectoryDefinition>…</TargetDirectoryDefinition>
   …
</TargetDirectoryDefinitions>
```

Parent element: Package

Attributes: none

Child elements:

Element	Description
TargetDirectoryDefinition	[0..*] This element contains a single target directory definition. File items that

	are part of the package can optionally be installed to a location that is selectable by the user. The appearance of this “virtual” folder is described by this element. See section 2.2.2.45, File Element, p. 28 for further information.

2.2.2.18 TargetDirectoryDefinition Element

This element contains a single target directory definition. File items that are part of the package can optionally be installed to a location that is selectable by the user. The appearance of this “virtual” folder is described by this element. See section 2.2.2.45, File Element, p. 28 for further information.

```
<TargetDirectoryDefinition>
   <Id>…</Id>
   <Name>…</Name>
   <Description>…</Description>
   <PromptUser>…</PromptUser>
   <DefaultValue>…</DefaultValue>
</TargetDirectoryDefinition>
```

Parent element: TargetDirectoryDefinitions

Attributes: none

Child elements:

Element	Description
Id	[1] This element contains a unique integer number identifying this target directory definition. At other locations in the package.manifest file, this definition can be referenced using `$` followed by this number.
Name	[1] The human-readable name of the target directory definition. This string can be prefixed with `$`; in that case, a localized string will be looked up in the string table.
Description	[0..1] A short description about the target directory definition. This string can be prefixed with `$`; in that case, a localized string will be looked up in the string table.
PromptUser	[0..1] This element contains a oolean value (`true` or `false`) which determines whether the user can change the default value of this target directory definition. If this element is omitted, `true` is assumed.
DefaultValue	[0..1] This element contains the default path for this target directory definition. If this element is omitted, the user must provide a value interactively. You can also use environment variables enclosed in percent signs (`%`). Beyond the variables provided by the operating system, there are some additional local environment variables: • AP_COMMON The location of the Common folder of the Automation Platform installation. • AP_PLUGINS: The location of the PlugIns folder of the Automation Platform installation. • AP_PROFILES: The location of the Profiles folder of the Automation Platform installation. • AP_ROOT: The location of the root folder of the Automation Platform installation. • DESKTOP:

	The location of the desktop folder. • DESKTOP_DIRECTORY: The location of the desktop folder. Equivalent to DESKTOP. • APP_DATA_CODESYS: The directory where user options are stored (of the current user). Brand labelling (i.e. a different product name) is automatically considered. • COMMON_APP_DATA_CODESYS: The directory where machine options are stored. Brand labelling (i.e. a different product name) is automatically considered.

2.2.2.19 Id Element (Below TargetDirectoryDefinition Element)

This element contains a unique integer number identifying this target directory definition. At other locations in the package.manifest file, this definition can be referenced using `$` followed by this number.

```
<Id>…</Id>
```

Parent element: TargetDirectoryDefinition

Attributes: none

Child elements: none

2.2.2.20 PromptUser Element

This element contains a oolean value (`true` or `false`) which determines whether the user can change the default value of this target directory definition. If this element is omitted, `true` is assumed.

```
<PromptUser>…</PromptUser>
```

Parent element: TargetDirectoryDefinition

Attributes: none

Child elements: none

2.2.2.21 DefaultValue Element

This element contains the default path for this target directory definition. If this element is omitted, the user must provide a value interactively.

```
<DefaultValue>…</DefaultValue>
```

Parent element: TargetDirectoryDefinition

Attributes: none

Child elements: none

2.2.2.22 Components Element

Contains information about the groups of items that are installed by this package. A component can be activated or deactivated by the user, and required dependencies between components can be specified.

```
<Components>
   <Component>…</Component>
   …
</Components>
```

Parent element : Package

Attributes: none

Child elements:

Element	Description
Component	[0..*] A component is a group of single package items which are installed together. The user is able to activate or deactivate single components. However, within the package, dependencies and parent-child relationships between components can be specified. Futhermore, a component can require environmental conditions (e.g. the existence of a certain plug-in or version profile).

2.2.2.23 Component Element

A component is a group of single package items which are installed together. The user is able to activate or deactivate single components. However, within the package, dependencies and parent-child relationships between components can be specified. Furthermore, a component can require environmental conditions (e.g. the existence of a certain plug-in or version profile).

```
<Component>
   <General>…</General>
   <ChildComponents>…</ChildComponents>
   <RequiredComponents>…</RequiredComponents>
   <Dependencies>…</Dependencies>
   <Items>…</Items>
   <ProfileSelectionList>…</ProfileSelectionList>
</Component>
```

Parent element: Components, ChildComponents

Attributes: none

Child elements:

Element	Description
General	[1] Contains global information about the component.
ChildComponents	[0..1] A component may contain multiple child components. That way, a user-friendly hierarchy of the component structure can be defined.
RequiredComponents	[0..1] A component may require multiple other components to be installed. If a user activates this component, the required components are activated automatically as well. On the other hand, the user cannot deactivate a required component without deactivate this component as well.
Dependencies	[0..1] A component may require some environmental conditions, e.g. the existence of a specific plug-in version. Such dependencies can be specified using this element.
Items	[1] This element contains the collection of individual items to be installed (the “content” of the package).
ProfileSelectionList	[0..1] This element contains the collection of profiles which should be shown in dialog of profile change selection. It can also be specified if the dialog of profile change selection should be shown or not. ***Requires at least package manager version 3.5.1.0.***

2.2.2.24 General Element (Below Component Element)

Contains global information about the component.

```
<General>
   <Id>…</Id>
   <Name>…</Name>
   <Description>…</Description>
   <Icon>…</Icon>
   <Selectable>…</Selectable>
   <SelectedByDefault>…</SelectedByDefault>
</General>
```

Parent element: Component

Attributes: none

Child elements:

Element	Description
Id	[1] An integer value identifying this component. This value must be unique within the package.manifest file.
Name	[1] The human-readable name of the component. This string can be prefixed with `$`; in that case, a localized string will be looked up in the string table.
Description	[0..1] A short description about the component and its contents. This string can be prefixed with `$`; in that case, a localized string will be looked up in the string table.
Icon	[0..1] Path to an icon file (.ico file format) that will be associated with the component. The path must point to a file within the package archive and must be specified relatively to the package.manifest file. ***Note:*** *This information is currently not evaluated yet..*
Selectable	[0..1] A boolean value (`true` or `false`) indicating whether this component can be activated or deactivated by the user. If this element is omitted, `true` is assumed.
SelectedByDefault	[0..1] A boolean value (`true` or `false`) indicating whether this component is initially activated or not. If this element is omitted, `true` is assumed.

2.2.2.25 Selectable Element

A oolean value (`true` or `false`) indicating whether this component can be activated or deactivated by the user. If this element is omitted, `true` is assumed.

```
<Selectable>…</Selectable>
```

Parent element: General (below Component)

Attributes: none

Child elements: none

2.2.2.26 SelectedByDefault Element

A oolean value (`true` or `false`) indicating whether this component is initially activated or not. If this element is omitted, `true` is assumed.

```
<SelectedByDefault>…</SelectedByDefault>
```

Parent element: General (below Component)

Attributes: none

Child elements: none

2.2.2.27 ChildComponents Element

A component may contain multiple child components. That way, a user-friendly hierarchy of the component structure can be defined.

```
<ChildComponents>
   <Component>…</Component>
   …
</ChildComponents>
```

Parent element: Component

Attributes: none

Child elements:

Element	Description
Component	[0..*] A component is a group of single package items which are installed together. The user is able to activate or deactivate single components. However, within the package, dependencies and parent-child relationships between components can be specified. Furthermore, a component can require environmental conditions (e.g. the existence of a certain plug-in or version profile).

2.2.2.28 RequiredComponents Element

A component may require multiple other components to be installed. If a user activates this component, the required components are activated automatically as well. On the other hand, the user cannot deactivate a required component without deactivate this component as well.

```
<RequiredComponents>
   <ComponentId>…</ComponentId>
   …
</RequiredComponents>
```

Parent element: Component

Attributes: none

Child elements:

Element	Description
ComponentId	[0..*] An integer value identifying the required component. This value corresponds to the value specified in the Id element of a component.

2.2.2.29 ComponentId Element

An integer value identifying the required component. This value corresponds to the value specified in the Id element of a component.

```
<ComponentId>…</ComponentId>
```

Parent element: RequiredComponents

Attributes: none

Child elements: none

2.2.2.30 Dependencies Element

A component may require some environmental conditions, e.g. the existence of a specific plug-in version. Such dependencies can be specified using this element.

```
<Dependencies>
   <MinimumProfile>…</MinimumProfile>
   …
   <MinimumPlugInVersion>…</MinimumPlugInVersion>
   …
</Dependencies>
```

Parent element : Component

Attributes: none

Child elements:

Element	Description
MinimumProfile	[0..*] If specified, the component can only be installed to a certain target profile if all plug-ins within the target profile have equal or newer versions than specified in the minimum profile.
MinimumPlugInVersion	[0..*] If specified, the component can only be installed if a plug-in with equal or newer version is part of the target profile.

2.2.2.31 MinimumProfile Element

If specified, the component can only be installed to a certain target profile if all plug-ins within the target profile have equal or newer versions than specified in the minimum profile.

```
<MinimumProfile>
   <Path>…</Path>
</MinimumProfile>
```

Parent element: Dependencies

Attributes: none

Child elements:

Element	Description
Path	[1] This element describes at which location within the package archive the minimum profile is located.

2.2.2.32 MinimumPlugInVersion Element

If specified, the component can only be installed if a plug-in with equal or newer version is part of the target profile.

```
<MinimumPlugInVersion>
   <PlugIn>…</PlugIn>
   <Version>…</Version>
</MinimumPlugInVersion>
```

Parent element : Dependencies

Attributes : none

Child elements:

Element	Description
PlugIn	[1] Identifies the plug-in to be checked by its GUID (**g**lobally **u**nique **id**entifier).

Version	[1] Specifies the minimum version of the plug-in. This string must be in one of the following formats: • major.minor • major.minor.build • major.minor.build.revision where major, minor, build, and revision are numbers between 0 and 255 each.

2.2.2.33 PlugIn Element

Identifies the plug-in to be checked by its GUID (**g**lobally **u**nique **id**entifier).

```
<PlugIn>…</PlugIn>
```

Parent element : MinimumPlugInVersion

Attributes : none

Child elements: none

2.2.2.34 Items Element

This element contains the collection of individual items to be installed (the “content” of the package).

```
<Items>
   <PlugIn>…</PlugIn>
   …
   <Library>…</Library>
   …
   <DeviceDescription>…</DeviceDescription>
   …
   <VendorDescription>…</VendorDescription>
   …
   <Profile>…</Profile>
   …
   <InformationalProfile>…</InformationalProfile>
   …
   <ProfileChange>…</ProfileChange>
   …
   <File>…</File>
   …
   <AddMenuCommand>…</AddMenuCommand>
   …
   <AddToolbarCommand>…</AddToolbarCommand>
   …
   <AssignShortcut>…</AssignShortcut>
   …
   <AddView>…</AddView>
   …
   <OnlineHelpFile>…</OnlineHelpFile>
   …
   <OnlineHelpMerge>…</OnlineHelpMerge>
   …
   <MenuConfiguration>…</MenuConfiguration>
   …
   <ToolbarConfiguration>…</ToolbarConfiguration>
   …
   <KeyboardConfiguration>…</KeyboardConfiguration>
   …
   <Option>…</Option>
   …
   <LibraryProfile>…</LibraryProfile>
   …
   <VisualizationStyle>…</VisualizationStyle>
   …
   <VisualizationExtension>…</VisualizationExtension>
   …
   <Folder>…</Folder>
   …
</Items>
```

Parent element: Component

Attributes: none

Child elements:

Element	Description
PlugIn	[0..*] This element causes the specified plug-in binary along with its dependencies to be installed.
Library	[0..*] This element causes the specified library to be installed.
DeviceDescription	[0..*] This element causes the specified device description to be installed.
VendorDescription	[0..*]

Element	Description
	This element causes the specified vendor description to be installed.
Profile	[0..*] This element causes the specified version profile to be installed. This profile will then be available as startup version profile. Note that this profile will be available at the next start of the application, so it will not be effective immediately.
InformationalProfile	[0..*] This element causes the specified version profile to be installed. This profile will then be available as informational version profile. Note that this profile will be available at the next start of the application, so it will not be effective immediately.
ProfileChange	[0..*] This element causes the specified changes in an existing version profile. Note that this change will be available at the next start of the application, so it will not be effective immediately.
File	[0..*] This element causes the specified file to copied somewhere into the target file system. The target path is either hard-coded or selectable by the user.
AddMenuCommand	[0..*] This element causes the specified command to be added to the menu configuration. Because of the fact that a menu configuration is maintained for each startup profile, the user must specify which profiles should be affected. Note that this profile will be available at the next start of the application, so it will not be effective immediately.
AddToolbarCommand	[0..*] This element causes the specified command to be added to the toolbar configuration. Because of the fact that a toolbar configuration is maintained for each startup profile, the user must specify which profiles should be affected. Note that this profile will be available at the next start of the application, so it will not be effective immediately.
AssignShortcut	[0..*] This element causes the specified command to be assigned with a shortcut. Because of the fact that a shortcut configuration is maintained for each startup profile, the user must specify which profiles should be affected. Note that this profile will be available at the next start of the application, so it will not be effective immediately.
AddView	[0..*] This element causes the specified view to be added to the view configuration. Note that this profile will be available at the next start of the application, so it will not be effective immediately.
OnlineHelpFile	[0..*] This element causes the specified help file to be added to the global online help system. Note that this copy action does not change the appearance of the help structure unless a corresponding OnlineHelpMerge item is also included within the package.
OnlineHelpMerge	[0..*]

Element	Description
	This element causes the specified merge file to be added to the global online help system.
MenuConfiguration	[0..*] This element causes an entire menu configuration file for a specific profile to be installed to the system. Note that this configuration will be available at the next start of the application, so it will not be effective immediately. ***Requires at least package manager version 3.4.1.0.***
ToolbarConfiguration	[0..*] This element causes an entire toolbar configuration file for a specific profile to be installed to the system. Note that this configuration will be available at the next start of the application, so it will not be effective immediately. ***Requires at least package manager version 3.4.1.0.***
KeyboardConfiguration	[0..*] This element causes an entire keyboard configuration file for a specific profile to be installed to the system. Note that this configuration will be available at the next start of the application, so it will not be effective immediately. ***Requires at least package manager version 3.4.1.0.***
Option	[0..*] This element causes an option to be added to the option storage. ***Requires at least package manager version 3.4.3.0.***
LibraryProfile	[0..*] This element causes a library profile file to be installed to the system. ***Requires at least package manager version 3.4.3.0.***
Folder	[0..*] This element causes the specified folder to copied somewhere into the target file system. The target path is either hard-coded or selectable by the user. ***Requires at least package manager version 3.5.1.0.***

2.2.2.35 PlugIn Element (Below Items Element)

This element causes the specified plug-in binary along with its dependencies to be installed. Please note that the plug-in will not be added to any version profile automatically. Typically, this step is performed by specifying an additional Profile, InformationalProfile, or ProfileChange element.

```
<PlugIn>
   <Path>…</Path>
</PlugIn>
```

Parent element: Items

Attributes: none

Child elements:

Element	Description
Path	[1] This element describes at which location within the package archive the plug-in binary is located. Please note that all required dependent binaries (interface components, GAC components) must be parallel to this path, otherwise the installation will fail.

2.2.2.36 Library Element

This element causes the specified library to be installed. Both source libraries (*.library) and compiled libraries (*.compiled-library) can be installed that way.

```
<Library>
   <Path>…</Path>
</Library>
```

Parent element: Items

Attributes: none

Child elements:

Element	Description
Path	[1] This element describes at which location within the package archive the library file is located.

2.2.2.37 DeviceDescription Element

This element causes the specified device description to be installed. Please note that this file must be a valid *.devdesc.xml file. It is not possible to convert a device description from any other format (*.eds, *.gs*, etc.).[1]

```
<DeviceDescription>
   <Path>…</Path>
</DeviceDescription>
```

Parent element: Items

Attributes: none

Child elements:

Element	Description
Path	[1] This element describes at which location within the package archive the device description file is located. This file must be in the *.devdesc.xml format.

2.2.2.38 VendorDescription Element

This element causes the specified vendor description to be installed.

```
<VendorDescription>
   <Path>…</Path>
</VendorDescription>
```

Parent element: Items

Attributes: none

Child elements:

Element	Description
Path	[1] This element describes at which location within the package archive the vendor description file is located. This file must be in the vendor.xml format.

[1] The feature to install any other formats like *.eds or *.gs* with the package management will probably never be implemented. The reason is that there is a 1:1 relationship between package item and corresponding file on the target machine. This relationship is important for reference counting which is crucial for a clean uninstall strategy. However, converting those formats into *.devdesc.xml typically causes multiple *.devdesc.xml files to be generated, which contradicts to this relationship.

2.2.2.39 Profile Element (Below Items Element)

This element causes the specified version profile to be installed. This profile will then be available as startup version profile. Note that this profile will be available at the next start of the application, so it will not be effective immediately.

```
<Profile>
   <Path>…</Path>
   <CreateStartMenuLink>…</CreateStartMenuLink>
   <CreateDesktopLink>…</CreateDesktopLink>
</Profile>
```

Parent element: Items

Attributes: none

Child elements:

Element	Description
Path	[1] This element describes at which location within the package archive the version profile file is located.
CreateStartMenuLink	[0..1] If set to true, a link for this version profile will be created in the start menu. This link will start the development system with this profile. If missing or false, no start menu link will be created. ***Requires at least package manager 3.5.3.0.***
CreateDesktopLink	[0..1] If set to true, a link for this version profile will be created on the desktop. This link will start the development system with this profile. If missing or false, no desktop link will be created. ***Requires at least package manager 3.5.3.0.***

Attention: The CreateStartMenuLink and CreateDesktopLink elements only work if the following conditions are met:

- The Common folder of the installation must contain a file SetupInfo.xml, with the following contents:
  ```
  <SetupInfo>
     <IDEExecutable>…</IDEExecutable>
     <StartMenuFolder>…</StartMenuFolder>
     <IconFile>…</IconFile>
  </SetupInfo>
  ```
  Default values in a CODESYS installation are:
  IDEExecutable=CODESYS.exe
  StartMenuFolder=C:\ProgramData\Microsoft\Windows\Start Menu\Programs\3S CoDeSys\CoDeSys
  IconFile=CODESYS.ico
- The Common folder of the installation must contain an icon file with the same name as specified in the IconFile element of the SetupInfo.xml file.

2.2.2.40 CreateStartMenuLink Element

If set to true, a link for this version profile will be created in the start menu. This link will start the development system with this profile. If missing or false, no start menu link will be created.

Parent element: Profile

Attributes: none

Child elements: none

Requires at least package manager 3.5.3.0.
See also the requirements described in the Profile Element chapter.

2.2.2.41 CreateDesktopLink Element

If set to true, a link for this version profile will be created on the desktop. This link will start the development system with this profile. If missing or false, no desktop link will be created.

Parent element: Profile

Attributes: none

Child elements: none

Requires at least package manager 3.5.3.0.
See also the requirements described in the Profile Element chapter.

2.2.2.42 InformationalProfile Element

This element causes the specified version profile to be installed. This profile will then be available as informational version profile. Note that this profile will be available at the next start of the application, so it will not be effective immediately.

```
<InformationalProfile>
   <Path>…</Path>
</InformationalProfile>
```

Parent element: Items

Attributes: none

Child elements:

Element	Description
Path	[1] This element describes at which location within the package archive the version profile file is located.

2.2.2.43 ProfileChange Element

This element causes the specified changes in an existing version profile. If such an element is part of the installation set, then the user must specify which existing version profile(s) should be affected by this change. Note that this change will be available at the next start of the application, so it will not be effective immediately.

```
<ProfileChange>
   <PlugIn>…</PlugIn>
   <Version>…</Version>
</ProfileChange>
```

Parent element: Items

Attributes: none

Child elements:

Element	Description
PlugIn	[1] This element contains the GUID of the plug-in that should be added or modified in the target version profile.
Version	[1] The version of the plug-in as it should be changed in the target version profile. This string must be in one of the following formats: • major.minor • major.minor.build • major.minor.build.revision where major, minor, build, and revision are numbers between 0 and 255 each.

Element	Description
	If more than one ProfileChange request for the same plug-in component occurs for the same target profile, the newest version specification will be considered.

2.2.2.44 PlugIn Element (Below ProfileChange Element)

This element contains the GUID of the plug-in that should be added or modified in the target version profile.

```
<PlugIn>…</PlugIn>
```

Parent element: ProfileChange

Attributes: none

Child elements: none

2.2.2.45 File Element

This element causes the specified file to copied somewhere into the target file system. The target path is either hard-coded or selectable by the user.

```
<File>
   <TargetFolder>…</TargetFolder>
   <Path>…</Path>
   <IgnoreArchiveFolder>…</IgnoreArchiveFolder>
   <Overwrite>…</Overwrite>
</File>
```

Parent element: Items

Attributes: none

Child elements:

Element	Description
TargetFolder	[1] This element contains the path to the target folder where the file should be copied to. If this is a string, that path will be directly used. If this is a `$` followed by an integer value, it is a reference into the target directory table (see section 2.2.2.17, TargetDirectoryDefinitions Element, p. 14) and the user will be able to select the desired target path interactively. You can also use environment variables enclosed in percent signs (`%`). Beyond the variables provided by the operating system, there are some additional local environment variables: • AP_COMMON The location of the Common folder of the Automation Platform installation. • AP_PLUGINS: The location of the PlugIns folder of the Automation Platform installation. • AP_PROFILES: The location of the Profiles folder of the Automation Platform installation. • AP_ROOT: The location of the root folder of the Automation Platform installation. • DESKTOP: The logical Desktop rather than the physical file system location
Path	[1] This element describes at which location within the package archive the file is

Element	Description
	located.
IgnoreArchiveFolder	[0..1] An element with boolean value (`true` or `false`) which determines whether the archive path is ignored and the installation routine behaves as if the file would be top-level within the package archive.. If this element is omitted, `false` is assumed.
Overwrite	[0..1] This element describes if the file should be overwritten or not if it already exists in the target folder where the file should be copied to.

2.2.2.46 TargetFolder Element

This element contains the path to the target folder where the file should be copied to. If this is a string, that path will be directly used. If this is a `$` followed by an integer value, it is a reference into the target directory table (see section 2.2.2.17, p. 14) and the user will be able to select the desired target path interactively.

```
<TargetFolder>…</TargetFolder>
```

Parent element: File, Folder

Attributes: none

Child elements: none

2.2.2.47 AddMenuCommand Element

This element causes the specified command to be added to the menu configuration. Because of the fact that a menu configuration is maintained for each startup profile, the user must specify which profiles should be affected. Note that this profile will be available at the next start of the application, so it will not be effective immediately.

```
<AddMenuCommand>
   <Command>…</Command>
   <InsertionPath>…</InsertionPath>
   <InsertionPath2>…</InsertionPath2>
   <InsertionPosition>…</InsertionPosition>
   <BeginGroup>…</BeginGroup>
</AddMenuCommand>
```

Parent element: Items

Attributes: none

Child elements:

Element	Description
Command	[1] This element contains the type GUID of the command that should be added to the menu configuration.
InsertionPath	***Package manager version <3.4.1.0:*** [1] ***Package manager version ≥3.4.1.0:*** [0..1] This element contains information about the location in the menu structure where the command should be added. It is a string which is a concatenation of the popup menu item names; the parts of that "path" are divided by double colons (`::`, e.g. `Edit::Advanced`. If this popup menu path does not yet exist, it will be created automatically. This specification can be prefixed with `$`; in that case, a localized string will be looked up in the string table.

Element	Description
	It is recommended to use the InsertionPath2 element instead because that element allows an exact specification *where* the elements of the popup path should be created. Using InsertionPath, they will always be created at the end of each level, which is not desired in most cases. ***Package manager version ≥3.4.1.0:*** The elements InsertionPath and InsertionPath2 are mutually exclusive. Exactly one of the two must be specified.
InsertionPath2	[0..1] This element contains information about the location in the menu structure where the command should be added. If this popup path does not yet exist, it will be created automatically. The elements InsertionPath and InsertionPath2 are mutually exclusive. Exactly one of the two must be specified. ***Requires at least package manager version 3.4.1.0.***
InsertionPosition	[1] Use that element to give information about where within the popup menu specified in the InsertionPath element the command should be located. It can be at the first or the last position, or before or after an already existing menu item.
BeginGroup	[0..1] An element with boolean value (`true` or `false`) which determines whether a separator item should be added before the new menu item. If this element is omitted, `false` is assumed.

2.2.2.48 AddToolbarCommand Element

This element causes the specified command to be added to the toolbar configuration. Because of the fact that a toolbar configuration is maintained for each startup profile, the user must specify which profiles should be affected. Note that this profile will be available at the next start of the application, so it will not be effective immediately.

```
<AddToolbarCommand>
   <Command>…</Command>
   <InsertionPath>…</InsertionPath>
   <InsertionPosition>…</InsertionPosition>
   <BeginGroup>…</BeginGroup>
</AddToolbarCommand>
```

Parent element: Items

Attributes: none

Child elements:

Element	Description
Command	[1] This element contains the type GUID of the command that should be added to the toolbar configuration.
InsertionPath	[1] This element contains information about which toolbar the command should be added to. If this toolbar path does not yet exist, it will be created automatically. This specification can be prefixed with `$`; in that case, a localized string will be looked up in the string table.
InsertionPosition	[1] Use that element to give information about where within the toolbar specified in the

Element	Description
	InsertionPath element the command should be located. It can be at the first or the last position, or before or after an already existing toolbar item.
BeginGroup	[0..1] An element with boolean value (`true` or `false`) which determines whether a separator item should be added before the new toolbar item. If this element is omitted, `false` is assumed.

2.2.2.49 Command Element

This element contains the type GUID of a command.

```
<Command>…</Command>
```

Parent element: AddMenuCommand, AddToolbarCommand, InsertionPosition, AssignShortcut

Attributes: none

Child elements: none

2.2.2.50 InsertionPath Element

This element contains information about the location in the menu or toolbar structure where the command should be added. It is a string which is a concatenation of the popup menu item names; the parts of that "path" are divided by double colons (`:`☺, e.g. `Edit::Advanced`. In the case of toolbar configuration it is simply the name of the toolbar. If this popup menu path or toolbar does not yet exist, it will be created automatically. This specification can be prefixed with `$`; in that case, a localized string will be looked up in the string table.

It is recommended to use the InsertionPath2 element instead because that element allows an exact specification *where* the elements of the popup path should be created. Using InsertionPath, they will always be created at the end of each level, which is not desired in most cases.

```
<InsertionPath>…</InsertionPath>
```

Parent element: AddMenuCommand, AddToolbarCommand

Attributes: none

Child elements: none

2.2.2.51 InsertionPath2 Element

This element contains information about the location in the menu structure where the command should be added. If this popup path does not yet exist, it will be created automatically.

Requires at least package manager version 3.4.1.0.

```
<InsertionPath2>
   <Popup>…</Popup>
   …
</InsertionPath2>
```

Parent element: AddMenuCommand

Attributes: none

Child elements:

Element	Description
Popup	[1..*] Specifies a part of the popup menu path where a new menu item should be added. If that popup menu does not yet exist, it will be created automatically.

2.2.2.52 Popup Element

Specifies a part of the popup menu path where a new menu item should be added. If that popup menu does not yet exist, it will be created automatically.

Requires at least package manager version 3.4.1.0.

```
<Popup>
   <Name>…</Name>
   <InsertionPosition>…</InsertionPosition>
</Popup>
```

Parent element: Popup

Attributes: none

Child elements:

Element	Description
Name	[1] This element contains the name of the popup menu item to be located or created. This specification can be prefixed with `$`; in that case, a localized string will be looked up in the string table.
InsertionPosition	[1] Use that element to give information about where the popup should be located. It can be at the first or the last position, or before or after an already existing command or popup item.

2.2.2.53 InsertionPosition Element (Below AddMenuCommand or AddToolbarCommand Element)

Use that element to give information about where within the popup menu or toolbar specified in the InsertionPath element the command should be located. It can be at the first or the last position, or before or after an already existing menu or toolbar item.

```
<InsertionPosition>
   <Where>…</Where>
   <Command>…</Command>
</InsertionPosition>
```

Parent element: AddMenuCommand, AddToolbarCommand

Attributes: none

Child elements:

Element	Description
Where	[1] One of the following values: • `First` The item is added at the first position within the popup menu or toolbar. • `Last` The item is added at the last position within the popup menu or toolbar. • `Before` The item is added before the item specified by the Command element. If that item does not exist, it will be added at the first position within the popup menu or toolbar. • `After` The item is added after the item specified by the Command element. If that item does not exist, it will be added at the last position within the popup menu or toolbar.

Element	Description
Command	[0..1] This element contains the type GUID of a command. This element must be present if the Where element is set to one of the values `Before` or `After`. It must not be present if the Where element is set to any other value.

2.2.2.54 InsertionPosition Element (Below Popup Element)

Use that element to give information about where the popup should be located. It can be at the first or the last position, or before or after an already existing command or popup item.

Requires at least package manager version 3.4.1.0.

```
<InsertionPosition>
   <Where>…</Where>
   <Command>…</Command>
   <PopupName>…</PopupName>
</InsertionPosition>
```

Parent element: Popup

Attributes: none

Child elements:

Element	Description
Where	[1] One of the following values: • `First` The item is added at the first position within the popup menu or toolbar. • `Last` The item is added at the last position within the popup menu or toolbar. • `Before` The item is added before the item specified by the Command or PopupName element. If that item does not exist, it will be added at the first position within the popup menu or toolbar. • `After` The item is added after the item specified by the Command or PopupName element. If that item does not exist, it will be added at the last position within the popup menu or toolbar.
Command	[0..1] This element contains the type GUID of a command. Exactly one the elements Command or PopupName must be present if the Where element is set to one of the values `Before` or `After`. Neither of them may be present if the Where element is set to any other value.
PopupName	[0..1] This element contains the name of a popup menu item. This specification can be prefixed with `$`; in that case, a localized string will be looked up in the string table. Exactly one the elements Command or PopupName must be present if the Where element is set to one of the values `Before` or `After`. Neither of them may be present if the Where element is set to any other value.

2.2.2.55 Where Element

One of the following values:

- `First`
  The item is added at the first position within the popup menu or toolbar.
- `Last`
  The item is added at the last position within the popup menu or toolbar.
- `Before`
  The item is added before the item specified by the Command element. If that item does not exist, it will be added at the first position within the popup menu or toolbar.
- `After`
  The item is added after the item specified by the Command element. If that item does not exist, it will be added at the last position within the popup menu or toolbar.

```
<Where>…</Where>
```

Parent element: InsertionPosition

Attributes: none

Child elements: none

2.2.2.56 PopupName Element

This element contains the name of a popup menu item. This specification can be prefixed with `$`; in that case, a localized string will be looked up in the string table.

Requires at least package manager version 3.4.1.0.

```
<PopupName>…</PopupName>
```

Parent element: InsertionPosition (Below Popup Element)

Attributes: none

Child elements: none

2.2.2.57 BeginGroup Element

An element with oolean value (`true` or `false`) which determines whether a separator item should be added before the new toolbar item. If this element is omitted, `false` is assumed.

```
<BeginGroup>…</BeginGroup>
```

Parent element: AddMenuCommand, AddToolbarCommand

Attributes: none

Child elements: none

2.2.2.58 AssignShortcut Element

This element causes the specified command to be assigned with a shortcut. Because of the fact that a shortcut configuration is maintained for each startup profile, the user must specify which profiles should be affected. Note that this profile will be available at the next start of the application, so it will not be effective immediately.

```
<AssignShortcut>
   <Command>…</Command>
   <Shortcut>…</Shortcut>
</AssignShortcut>
```

Parent element: Items

Attributes: none

Child elements:

Element	Description
Command	[1] This element contains the type GUID of the command that should be added to the

Element	Description
	shortcut configuration.
Shortcut	[1] One of the following values: None Ins Del F1 F2 F3 F4 F5 F6 F7 F8 F9 F10 F11 F12 ShiftIns ShiftDel ShiftF1 ShiftF2 ShiftF3 ShiftF4 ShiftF5 ShiftF6 ShiftF7 ShiftF8 ShiftF9 ShiftF10 ShiftF11 ShiftF12 CtrlIns CtrlDel Crtl0 Crtl1 Crtl2 Crtl3 Crtl4 Crtl5 Crtl6 Crtl7 Crtl8 Crtl9 CrtlA CrtlB CrtlC CrtlD CrtlE CrtlF CrtlG CrtlH CrtlI CrtlJ CrtlK CrtlL CrtlM CrtlN CrtlO CrtlP CrtlQ CrtlR CrtlS CrtlT CrtlU CrtlV CrtlW CrtlX CrtlY CrtlZ CtrlF1 CtrlF2 CtrlF3 CtrlF4 CtrlF5 CtrlF6 CtrlF7 CtrlF8 CtrlF9 CtrlF10 CtrlF11 CtrlF12 CtrlShift0 CtrlShift1 CtrlShift2 CtrlShift3 CtrlShift4 CtrlShift5 CtrlShift6 CtrlShift7 CtrlShift8 CtrlShift9 CtrlShiftA CtrlShiftB CtrlShiftC CtrlShiftD CtrlShiftE CtrlShiftF CtrlShiftG CtrlShiftH CtrlShiftI CtrlShiftJ CtrlShiftK CtrlShiftL CtrlShiftM CtrlShiftN CtrlShiftO CtrlShiftP CtrlShiftQ CtrlShiftR CtrlShiftS CtrlShiftT CtrlShiftU CtrlShiftV CtrlShiftW CtrlShiftX CtrlShiftY CtrlShiftZ CtrlShiftF1 CtrlShiftF2 CtrlShiftF3 CtrlShiftF4 CtrlShiftF5 CtrlShiftF6 CtrlShiftF7 CtrlShiftF8 CtrlShiftF9 CtrlShiftF10 CtrlShiftF11 CtrlShiftF12 AltBksp AltLeftArrow AltUpArrow AltRightArrow AltDownArrow Alt0 Alt1 Alt2 Alt3 Alt4 Alt5 Alt6 Alt7 Alt8 Alt9 AltF1 AltF2 AltF3 AltF4 AltF5 AltF6 AltF7 AltF8 AltF9 AltF10 AltF11 AltF12

2.2.2.59 Shortcut Element

One of the following values:

None Ins Del F1 F2 F3 F4 F5 F6 F7 F8 F9 F10 F11 F12 ShiftIns ShiftDel ShiftF1 ShiftF2 ShiftF3 ShiftF4 ShiftF5 ShiftF6 ShiftF7 ShiftF8 ShiftF9 ShiftF10 ShiftF11 ShiftF12 CtrlIns CtrlDel Crtl0 Crtl1 Crtl2 Crtl3 Crtl4 Crtl5 Crtl6 Crtl7 Crtl8 Crtl9 CrtlA CrtlB CrtlC CrtlD CrtlE CrtlF CrtlG CrtlH CrtlI CrtlJ CrtlK CrtlL CrtlM CrtlN CrtlO CrtlP CrtlQ CrtlR CrtlS CrtlT CrtlU CrtlV CrtlW CrtlX CrtlY CrtlZ CtrlF1 CtrlF2 CtrlF3 CtrlF4 CtrlF5 CtrlF6 CtrlF7 CtrlF8 CtrlF9 CtrlF10 CtrlF11 CtrlF12 CtrlShift0 CtrlShift1 CtrlShift2 CtrlShift3 CtrlShift4 CtrlShift5 CtrlShift6 CtrlShift7 CtrlShift8 CtrlShift9 CtrlShiftA CtrlShiftB CtrlShiftC CtrlShiftD CtrlShiftE CtrlShiftF CtrlShiftG CtrlShiftH CtrlShiftI CtrlShiftJ CtrlShiftK CtrlShiftL CtrlShiftM CtrlShiftN CtrlShiftO CtrlShiftP CtrlShiftQ CtrlShiftR CtrlShiftS CtrlShiftT CtrlShiftU CtrlShiftV CtrlShiftW CtrlShiftX CtrlShiftY CtrlShiftZ CtrlShiftF1 CtrlShiftF2 CtrlShiftF3 CtrlShiftF4 CtrlShiftF5 CtrlShiftF6 CtrlShiftF7 CtrlShiftF8 CtrlShiftF9 CtrlShiftF10 CtrlShiftF11 CtrlShiftF12 AltBksp AltLeftArrow AltUpArrow AltRightArrow AltDownArrow Alt0 Alt1 Alt2 Alt3 Alt4 Alt5 Alt6 Alt7 Alt8 Alt9 AltF1 AltF2 AltF3 AltF4 AltF5 AltF6 AltF7 AltF8 AltF9 AltF10 AltF11 AltF12

```
<Shortcut>…</Shortcut>
```

Parent element: AssignShortcut

Attributes: none

Child elements: none

2.2.2.60 AddView Element

This element causes the specified view to be added to the view configuration. Note that this profile will be available at the next start of the application, so it will not be effective immediately.

```
<AddView>
   <ViewFactory>…</ViewFactory>
</AddView>
```

Parent element: Items

Attributes: none

Child elements:

Element	Description
ViewFactory	[1] This element contains the GUID of the view factory that should be opened the next time the application starts.

2.2.2.61 ViewFactory Element

This element contains the GUID of the view factory that should be opened the next time the application starts.

```
<ViewFactory>…</ViewFactory>
```

Parent element: AddView

Attributes: none

Child elements: none

2.2.2.62 OnlineHelpFile Element

This element causes the specified help file to be added to the global online help system. Note that this copy action does not change the appearance of the help structure unless a corresponding OnlineHelpMerge item is also included within the package.

```
<OnlineHelpFile>
   <Culture>…</Culture>
   <Path>…</Path>
</OnlineHelpFile>
```

Parent element: Items

Attributes: none

Child elements:

Element	Description
Culture	[1] The language of the help file. This must be a valid culture specifier (e.g. `zh-CHS`, `de-DE`, or `en`). This file will be copied to the corresponding localized online help folder in the target installation file system.
Path	[1] This element describes at which location within the package archive the help file is located. This file must be either in the *.chm or the *.htm/*.html format.

2.2.2.63 OnlineHelpMerge Element

This element causes the specified merge file to be added to the global online help system.

```
<OnlineHelpMerge>
   <Path>…</Path>
</OnlineHelpMerge>
```

Parent element: Items

Attributes: none

Child elements:

Element	Description
Path	[1] This element describes at which location within the package archive the merge file is

	located. This file must be in the *.merge format.

2.2.2.64 Path Element

This element can be found at various places within the package.manifest file reference. In all cases, it describes the location of a file within the package archive, relatively to the package.manifest file itself.

```
<Path>…</Path>
```

Parent element: PlugIn, Library, DeviceDescription, VendorDescription, Profile, InformationalProfile, File, OnlineHelpFile, OnlineHelpMerge, LibraryProfile, Folder

Attributes: none

Child elements: none

2.2.2.65 MenuConfiguration Element

This element causes an entire menu configuration file for a specific profile to be installed to the system. Note that this configuration will be available at the next start of the application, so it will not be effective immediately.

Requires at least package manager version 3.4.1.0.

```
<MenuConfiguration>
   <Path>…</Path>
   <Profile>…</Profile>
</MenuConfiguration>
```

Parent element: Items

Attributes: none

Child elements:

Element	Description
Path	[1] This element describes at which location within the package archive the file is located.
Profile	[1] This element contains the name of the profile which the configuration file should apply for.

2.2.2.66 ToolbarConfiguration Element

This element causes an entire toolbar configuration file for a specific profile to be installed to the system. Note that this configuration will be available at the next start of the application, so it will not be effective immediately.

Requires at least package manager version 3.4.1.0.

```
<ToolbarConfiguration>
   <Path>…</Path>
   <Profile>…</Profile>
</ToolbarConfiguration>
```

Parent element: Items

Attributes: none

Child elements:

Element	Description
Path	[1] This element describes at which location within the package archive the file is located.

Profile	[1] This element contains the name of the profile which the configuration file should apply for.

2.2.2.67 KeyboardConfiguration Element

This element causes an entire keyboard configuration file for a specific profile to be installed to the system. Note that this configuration will be available at the next start of the application, so it will not be effective immediately.

Requires at least package manager version 3.4.1.0.

```
<MenuConfiguration>
   <Path>…</Path>
   <Profile>…</Profile>
</MenuConfiguration>
```

Parent element: Items

Attributes: none

Child elements:

Element	Description
Path	[1] This element describes at which location within the package archive the file is located.
Profile	[1] This element contains the name of the profile which the configuration file should apply for.

2.2.2.68 Profile Element (Below MenuConfiguration, ToolbarConfiguration, or KeyboardConfiguration Element)

This element contains the name of the profile which the configuration file should apply for.

Requires at least package manager version 3.4.1.0.

```
<Profile>…</Profile>
```

Parent element: MenuConfiguration, ToolbarConfiguration, KeyboardConfiguration

Attributes: none

Child elements: none

2.2.2.69 Option Element

This element causes an option to be added to the option storage.

Requires at least package manager version 3.4.3.0.

```
<Option>
   <Key>…</Key>
   <Value>…</Value>
</Option>
```

Parent element: Items

Attributes: none

Child elements:

Element	Description
Key	[1] This element contains the full key path of the option that should be added to the option storage. The single keys must be separated by "\". First path element must be the option root (e. G. "User" for current user account, "Machine" for local machine, "AllUsers" for all

Element	Description
	users).
Value	[1] This element contains the value of the key to be added. The content must conform to how the value is usually written to the CODESYS.opt file.

2.2.2.70 Key Element

This element contains the full key path of the option that should be added to the option storage. The single keys must be separated by "\". First path element must be the option root (e. G. "User" for current user account, "Machine" for local machine, "AllUsers" for all users).

```
<Key>…</Key>
```

Parent element: Option

Attributes: none

Child elements: none

2.2.2.71 Value Element

This element contains the value of the key to be added. The content must conform to how the value is usually written to the CODESYS.opt file.

```
<Value>…</Value>
```

Parent element: Option

Attributes: none

Child elements: none

2.2.2.72 LibraryProfile Element

This element causes a library profile file to be installed to the system.

Requires at least package manager version 3.4.3.0.

```
<LibraryProfile>
   <Path>…</Path>
</LibraryProfile>
```

Parent element: Items

Attributes: none

Child elements:

Element	Description
Path	[1] This element describes at which location within the package archive the file is located.

2.2.2.73 VisualizationStyle Element

This element causes the specified visualization style to be installed.
If the style references images, then these references must be relative paths and the according files must be contained within the package file using the referencing image path.

Requires at least package manager version 3.4.4.0.

```
<VisualizationStyle>
   <Path>…</Path>
</VisualizationStyle>
```

Parent element: Items

Attributes: none

Child elements:

Element	Description
Path	[1] This element describes at which location within the package archive the library file is located.

2.2.2.74 VisualizationExtensions Element

This element causes the specified visualization extension to be installed.
Requires at least package manager version 3.4.4.0.

```
<VisualizationExtensions>
    <Repository>…</Repository>
    <Company>…</Company>
    <Version>…</Version>
    <Name>…</Name>
    <Path>…</Path>
</VisualizationExtensions>
```

Parent element: Items

Attributes: none

Child elements:

Element	Description
Repository	[1] The visualization repository in which the extension should be installed. F.e. System
Company	[1] The company name of the extension.
Version	[1] The version of this package. This string must be in one of the following formats: • major.minor • major.minor.build • major.minor.build.revision where major, minor, build, and revision are numbers between 0 and 255 each.
Name	[1] The human-readable name of the extension. This string can be prefixed with `$`; in that case, a localized string will be looked up in the string table.
Path	[1] This element describes at which location within the package archive the extension file is located.

2.2.2.75 ExternalCall element

This element instructs the Package Manager to execute an external program.

Requires at least Package Manager version 3.5.

```
<ExternalCall>
   <Installation>…</Installation>
   <Uninstallation>…</Uninstallation>
</ExternalCall>
```

Parent element: Items

Attributes: none

Child elements:

Element	Description
Installation	[0..1] This element describes which external program to call during installation of the package.
Uninstallation	[0..1] This element describes which external program to call during uninstallation of the package.

2.2.2.76 Installation element

This element describes which external program to call during installation of the package.

Requires at least Package Manager version 3.5.

```
<Installation>
   <FileName>…</FileName>
   <Arguments>…</Arguments>
   <CreateNoWindow>…</CreateNoWindow>
   <EnvironmentVariable>…</EnvironmentVariable>
   <Exit>…</Exit>
</Installation>
```

Parent element: ExternalCall

Attributes: none

Child elements:

Element	Description
FileName	[1] A string denoting the path to the executable file. Environment variables will be expanded automatically.
Arguments	[0..1] This element specifies the command line arguments to be passed to the process.
CreateNoWindow	[0..1] True to start the process without creating a new window to contain it; otherwise, false. The default value is true.
EnvironmentVariable	[0..*] Defines a specific environment variable to be passed to the process.
Exit	[0..*] Defines one or more reactions to exit conditions of the process.

2.2.2.77 Uninstallation element

This element describes which external program to call during uninstallation of the package.

Requires at least Package Manager version 3.5.

```
<Installation>
   <FileName>…</FileName>
   <Arguments>…</Arguments>
   <CreateNoWindow>…</CreateNoWindow>
   <EnvironmentVariable>…</EnvironmentVariable>
   <Exit>…</Exit>
</Installation>
```

Parent element: ExternalCall

Attributes: none

Child elements:

Element	Description
FileName	[1] A string denoting the path to the executable file. Environment variables will be expanded automatically.
Arguments	[0..1] This element specifies the command line arguments to be passed to the process.
CreateNoWindow	[0..1] True to start the process without creating a new window to contain it; otherwise, false. The default value is true.
EnvironmentVariable	[0..*] Defines a specific environment variable to be passed to the process.
Exit	[0..*] Defines one or more reactions to exit conditions of the process.

2.2.2.78 FileName element

A string denoting the path to the executable file. Environment variables will be expanded automatically.

Requires at least Package Manager version 3.5.

```
<FileName>…</FileName>
```

Parent element: Installation, Uninstallation

Attributes: none

Child elements: none

2.2.2.79 Arguments element

This element specifies the command line arguments to be passed to the process.

Requires at least Package Manager version 3.5.

```
<Arguments>…</Arguments>
```

Parent element: Installation, Uninstallation

Attributes: none

Child elements: none

2.2.2.80 CreateNoWindow element

True to start the process without creating a new window to contain it; otherwise, false. The default value is true.

Requires at least Package Manager version 3.5.

```
<CreateNoWindow>…</CreateNoWindow>
```

Parent element: Installation, Uninstallation

Attributes: none

Child elements: none

2.2.2.81 EnvironmentVariable element

Defines a specific environment variable to be passed to the process.

Requires at least Package Manager version 3.5.

```
<EnvironmentVariable Name="…" Value="…"/>
```

Parent element: Installation, Uninstallation

Attributes:

Attribute	Description
Name	Required. The name of the environment variable.
Value	Required. The value of the environment variable.

Child elements: none

2.2.2.82 Exit element

Defines one or more reactions to exit conditions of the process.

Requires at least Package Manager version 3.5.

```
<Exit>
   <Code>…</Code>
   <IsError>…</IsError>
   <Message>…</Message>
</Exit>
```

Parent element: Installation, Uninstallation

Attributes: none

Child elements:

Element	Description
Code	[0..1] The exit code of the process. If the process exits with this specific code, the sibling IsError and Message elements are evaluated. If this element is missing, all error codes that are not explicitly specified elsewhere are handled with this block.
IsError	[0..1] If true, the corresponding exit code of the process is considered as an error. The user will get notified about this error. If false, installation or uninstallation will continue silently. The default value is true.
Message	[0..1] The message to be displayed to the user for this specific error code of the process. If not specified, a default error text is displayed. This element is meaningless if the IsError element is false.

2.2.2.83 Code element

The exit code of the process. If the process exits with this specific code, the sibling IsError and Message elements are evaluated. If this element is missing, all error codes that are not explicitly specified elsewhere are handled with this block.

Requires at least Package Manager version 3.5.

```
<Code>…</Code>
```

Parent element: Exit

Attributes: none

Child elements: none

2.2.2.84 IsError element

If true, the corresponding exit code of the process is considered as an error. The user will get notified about this error. If false, installation or uninstallation will continue silently. The default value is true.

Requires at least Package Manager version 3.5.

```
<IsError>…</IsError>
```

Parent element: Exit

Attributes: none

Child elements: none

2.2.2.85 Message element

The message to be displayed to the user for this specific error code of the process. If not specified, a default error text is displayed. This element is meaningless if the IsError element is false.

Requires at least Package Manager version 3.5.

```
<Message>…</Message>
```

Parent element: Exit

Attributes: none

Child elements: none

2.2.2.86 Folder element

This element causes the specified folder to copied somewhere into the target file system. The target path is either hard-coded or selectable by the user.

Requires at least package manager version 3.5.1.0.

```
<Folder>
   <TargetFolder>…</TargetFolder>
   <Path>…</Path>
    <IgnoreArchiveFolder>…<IgnoreArchiveFolder>
</Folder>
```

Parent element: Items

Attributes: none

Child elements:

Element	Description
TargetFolder	[1] This element contains the path to the target folder where the folder should be copied to. If this is a string, that path will be directly used. If this is a `$` followed by an integer value, it is a reference into the target directory table (see section 2.2.2.17, TargetDirectoryDefinitions Element, p. 14) and the user will be able to select the desired target path interactively.

Element	Description
	You can also use environment variables enclosed in percent signs (%). Beyond the variables provided by the operating system, there are some additional local environment variables: • AP_COMMON The location of the Common folder of the Automation Platform installation. • AP_PLUGINS: The location of the PlugIns folder of the Automation Platform installation. • AP_PROFILES: The location of the Profiles folder of the Automation Platform installation. • AP_ROOT: The location of the root folder of the Automation Platform installation. • DESKTOP: The logical Desktop rather than the physical file system location
Path	[1] This element describes at which location within the package archive the folder is located.
IgnoreArchiveFolder	[0..1] An element with boolean value (`true` or `false`) which determines whether the archive path is ignored and the installation routine behaves as if the folder would be top-level within the package archive.. If this element is omitted, `false` is assumed.

2.2.2.87 ProfileSelectionList element

Some items may require to select the profiles which should be affected. This element can be used to configure the dialog of profile change selection. It can also be defined if the dialog of profile change selection should be shown during installation or not. If the dialog of profile change selection is hidden and no profile is defined (no ActiveProfileOnly attribute or Profile element defined), all installed profiles and profiles of the package are used.

Requires at least package manager version 3.5.1.0.

```
<ProfileSelectionList>
   <Profiles>…</Profiles>
</ProfileSelectionList>
```

Parent element: Component

Attributes:

Attribute	Description
Hide	Optional attribute. If `true` the dialog of profile change is hidden during installation process. If this attribute is omitted, `false` is assumed. If one component has set `hide = true` and another component `hide = false`, the `false` covers the `true` and the dialog is shown.
ActiveProfileOnly	Optional attribute. This attribute specifies that only the active profile should be affected. If this attribute is `false` the Profile elements are ignored.

Attribute	Description
	If this attribute is omitted, `false` is assumed.

Child elements:

Element	Description
Profile	[0..*] This element defines a profile which should be part of the profile selection list.

2.2.2.88 Profile element (below Profiles element)

This element defines a profile which should be part of the profile selection list.

Requires at least package manager version 3.5.1.0.

```
<Profile>…</Profile>
```

Parent element: Profiles

Attributes: none

Child elements: none

Element	Description
Profile	[0..*] The name of the profile which should be part of the profile selection list.

2.2.2.89 ReadMe element

Path to a TXT or RTF file that contains the read me for the package. The path must point to a file within the package archive and must be specified relatively to the package.manifest file. It can be prefixed with `$`; in that case, a localized path will be looked up in the string table, thus supporting multiple license agreements in different languages. The read me file will be displayed after the installation has been executed successfully.

```
<ReadMe>…</ReadMe>
```

Parent element: General (below Package)

Attributes: none

Child elements: none

2.2.2.90 Overwrite element

This element describes if the file should be overwritten or not if it already exists in the target folder where the file should be copied to.

```
<Overwrite>
   <Option>…</Option>
   <Version>…</Version>
</Overwrite>
```

Parent element: File

Attributes: none

Child elements:

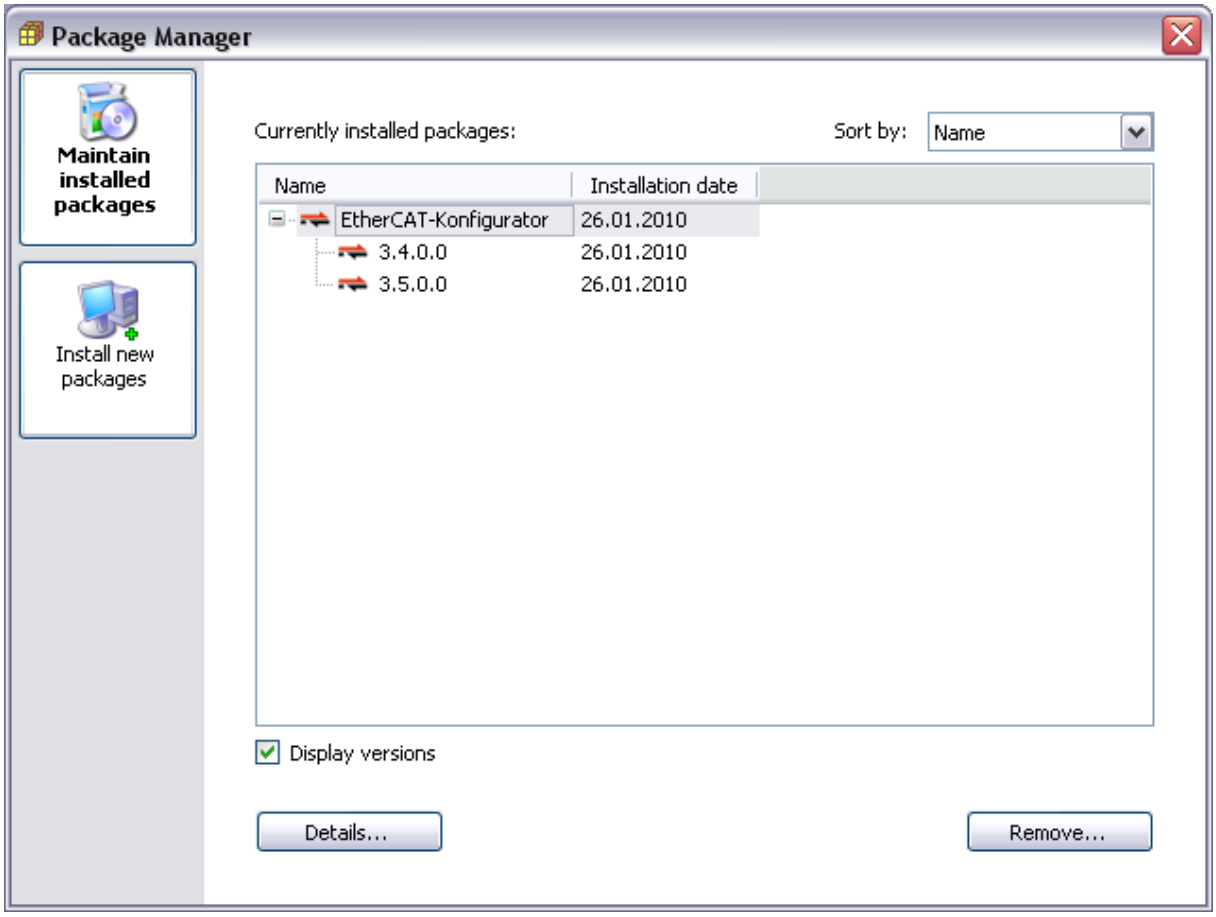
Element	Description
Option	[1] The overwrite option. This string must have one of the following values: • Yes

Element	Description
	• No • TargetVersionSmaller • TargetVersionEqual • TargetVersionGreater • TargetVersionSmallerOrEqual • TargetVersionGreaterOrEqual Yes: Overwrite No: Do not overwrite TargetVersionSmaller, TargetVersionEqual, TargetVersionGreater, TargetVersionSmallerOrEqual or TargetVersionGreaterOrEqual: Overwrite dependent on version given in Version element or if Version element is not present the package file version
Version	[0..1] The version to compare with if Option is TargetVersionSmaller, TargetVersionEqual, TargetVersionGreater, TargetVersionSmallerOrEqual or TargetVersionGreaterOrEqual. This string must be in one of the following formats: • major.minor • major.minor.build • major.minor.build.revision where major, minor, build, and revision are numbers between 0 and 255 each.

3 User Interface

3.1 The Package Manager

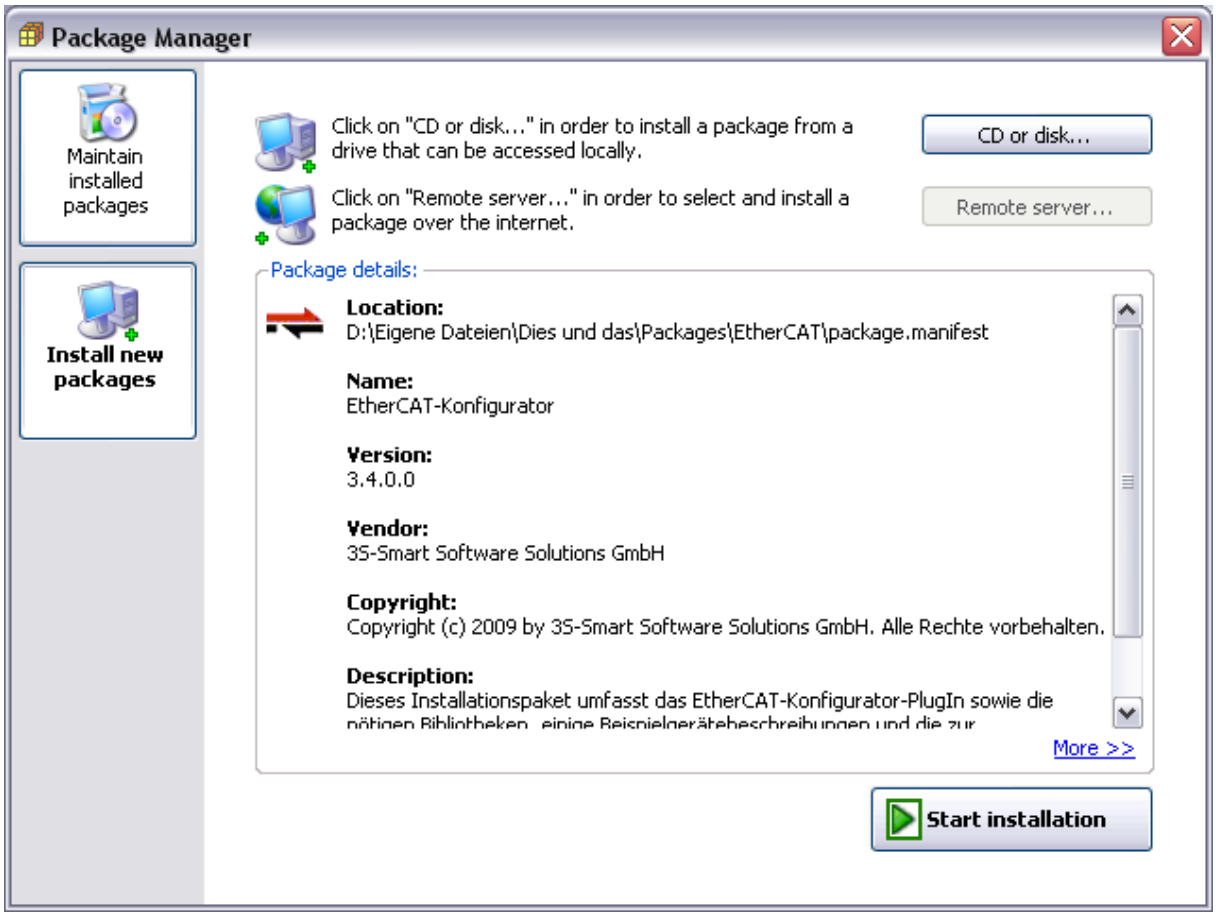
The main Package Manager dialog consists of two pages: The **Maintain installed packages** page where already installed packages are listed and from where they can be uninstalled, and the **Install new packages** page where the user can browse to a package for installation.



As you can see in the screenshot, it is possible to install different versions of the same package side by side. If you click on the **Remove…** button, the behaviour depends on the currently selected package:

- If **Display versions** is not checked, then all versions of the selected package are uninstalled.
- If **Display versions** is checked, and a top-level package node is selected, then all versions of the selected package are uninstalled.
- If **Display versions** is checked, and a single package version node is selected, then exactly this version is uninstalled.

The **Details…** button displays information that is associated with the package.



On this page, initially only the two button rows on the top are displayed. After the user has selected a package to install, the lower area is filled with information that is part of the package file. The “More >>” link leads to the HTML page that is defined in the package (see p. 14).

Note: *The “Remote server…” functionality is currently not implemented.*

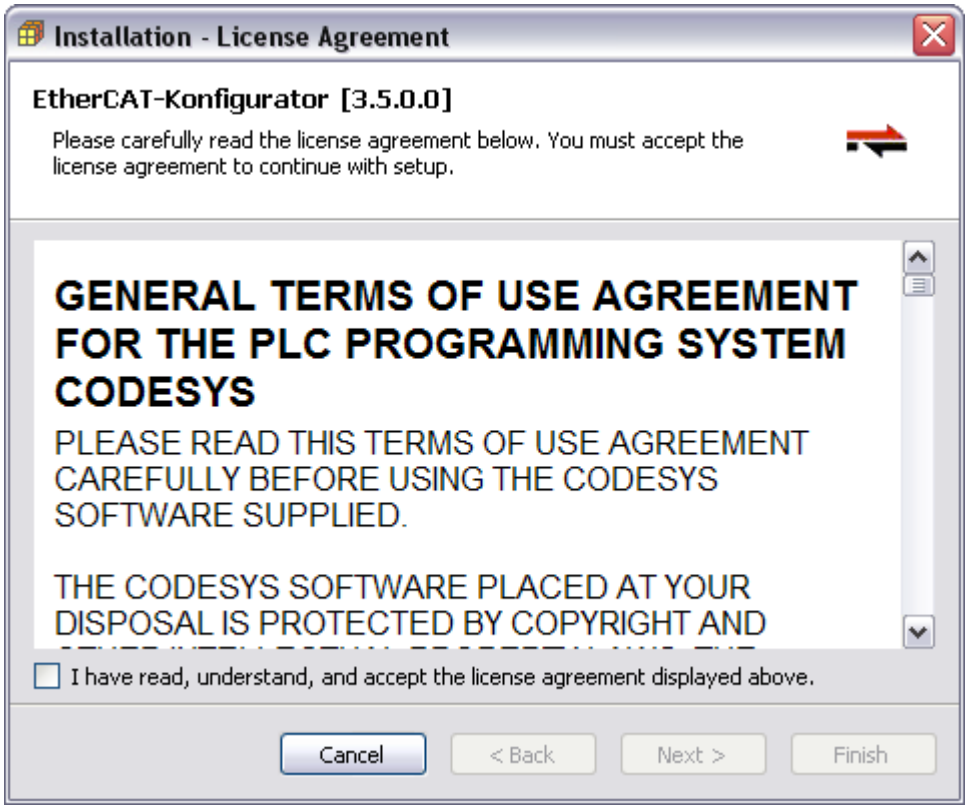
Note: It is not possible to install the same package twice. However, it is possible to install different versions of the same profile.

3.2 Installing a Package

After selecting a package file for install, a wizard will appear that guides the user through the installation process.

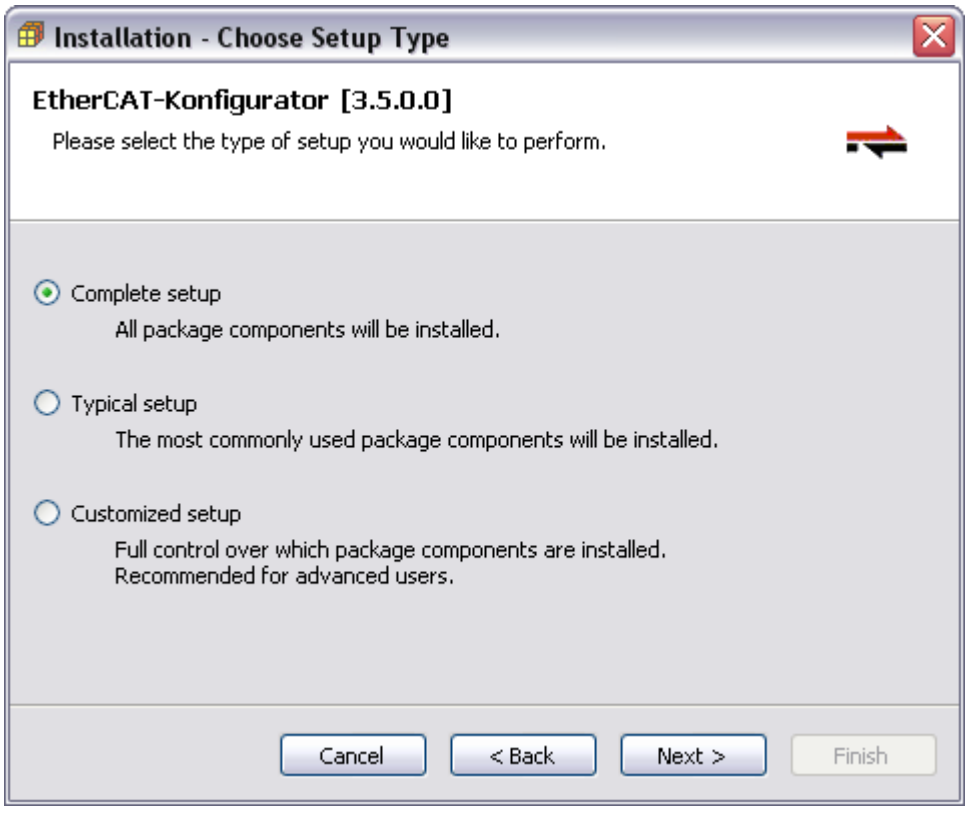
This wizard contains the following steps:

3.2.1 License Agreement



This page appears if the package file contains a license agreement (see section 2.2.2.16, p. 14). The user must confirm this agreement by clicking the checkbox, otherwise the installation process cannot be continued. If the package does not contain a license agreement, this page will not appear.

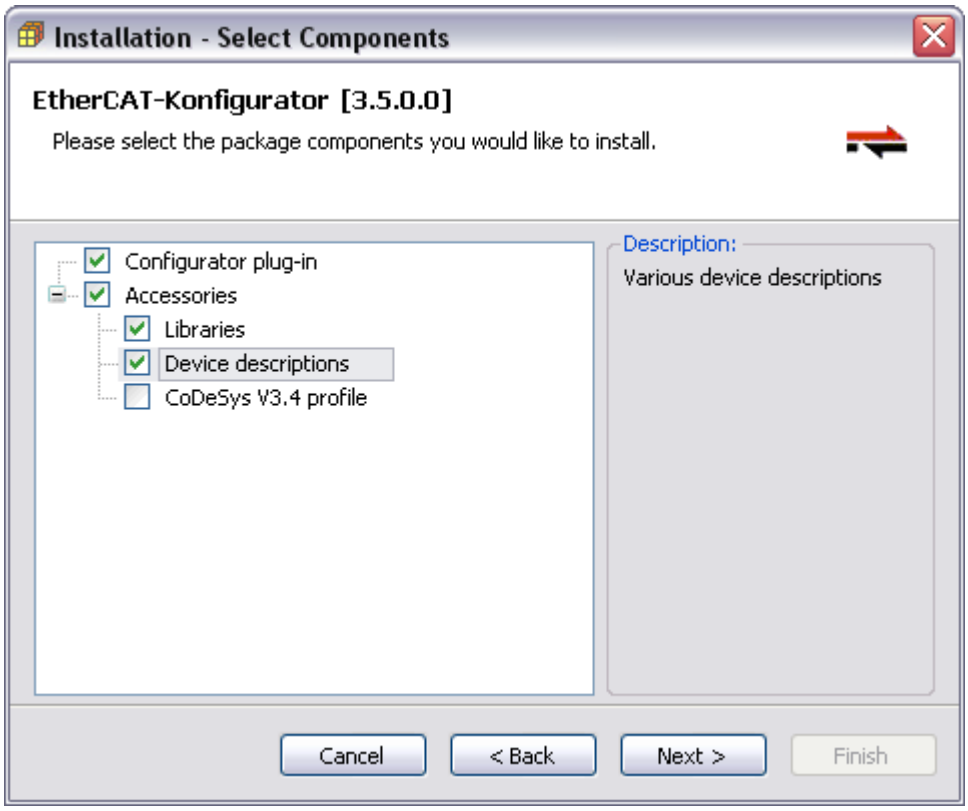
3.2.2 Choose Setup Type



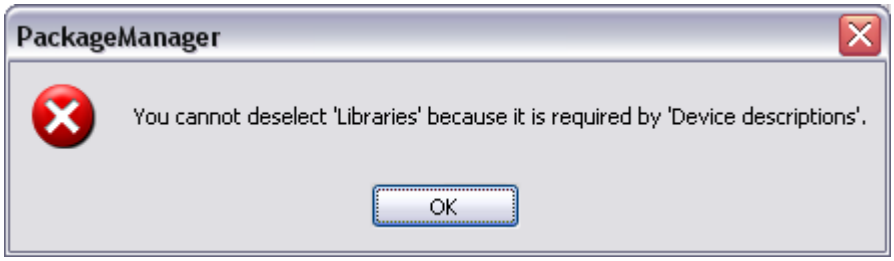
The user can select from three typical setup scenarios:

- **Complete setup:** All components in the package will be installed.
- **Typcial setup:** All components in the package which do not contain a SelectedByDefault element, or which contain a SelectedByDefault element with value true will be installed. (See section 2.2.2.26, p. 18)
- **Customized setup:** An additional wizard page will appear where the user can select which components should be installed.

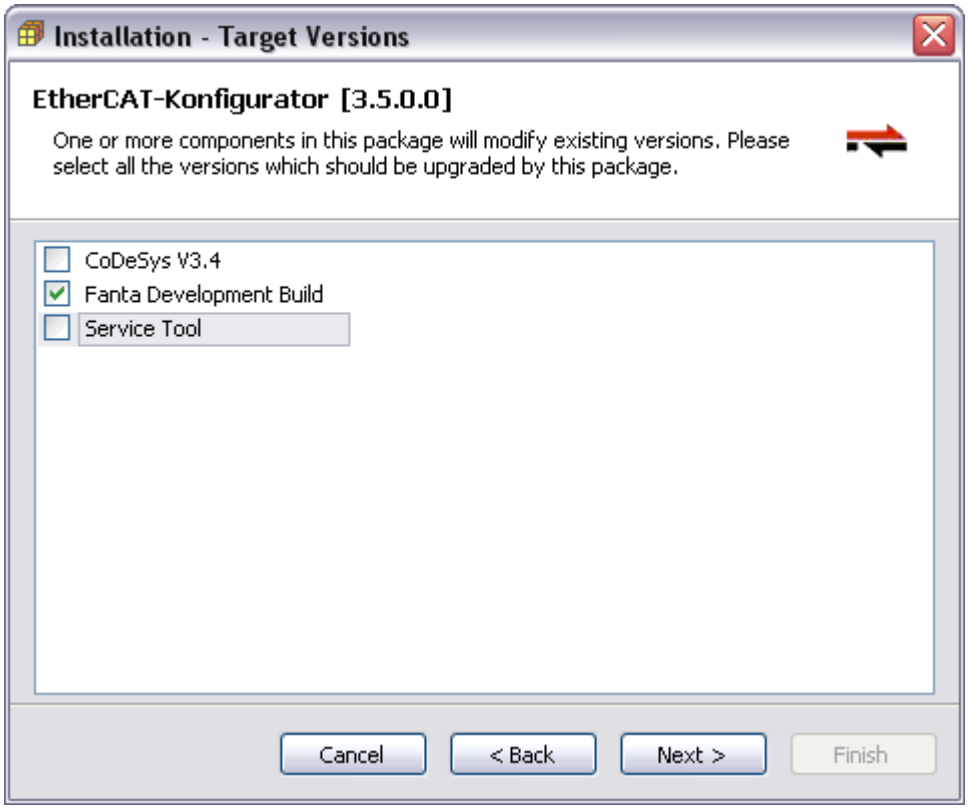
3.2.3 Select Components



This page only appears if the user selected “Customized setup” in the **Choose Setup Type** wizard page. Here it is possible to select exactly those components that should be installed. This dialog takes care that all dependencies and requirements are evaluated correctly: if a component requires another component, it is selected automatically, and if a required component is deselected, an error message appears, like so:



3.2.4 Target Versions

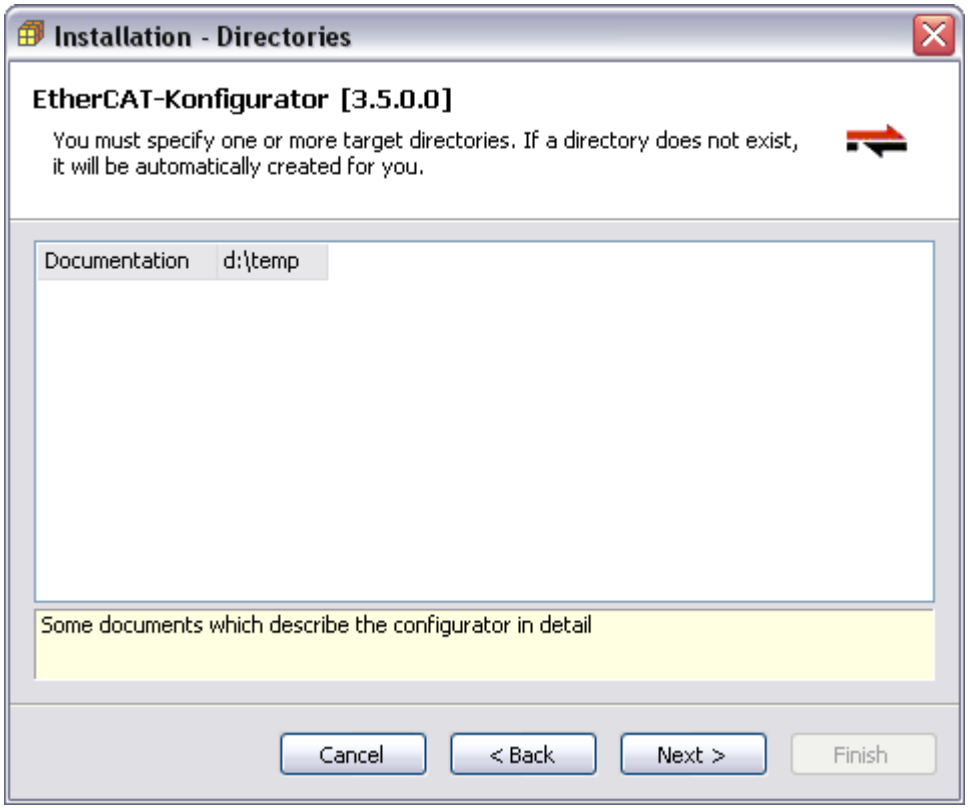


In this wizard page, the user can select which already existing target version profiles should be affected by the package installation. At least one version profile must be chosen here. This page only appears, if at least one item of the following types is in the set of selected components:

- ProfileChange (p. 27)
- AddMenuCommand (p. 29)
- AddToolbarCommand (p. 30)
- AssignShortcutCommand (p.34)
- OnlineHelpMerge (p.36)

The list of profiles which is shown in this dialog can and if the dialog should be shown or not can be configured by the ProfileSelectionList element.

3.2.5 Directories



In this wizard page, the user can select target directories for files. At least one version profile must be chosen here. This page only appears, if at least one File item (p. 28) which contains a reference to a TargetDirectoryDefinition (p. 15) is in the set of selected components.

3.2.6 Progress, Setup Completed

The last two wizard pages indicate the current progress of the installation, and the confirmation that the package has been successfully installed.

3.3 Uninstalling a Package

The uninstallation wizard simply contains three pages. No data has to be entered by the user.

- Confirmation page
- Progress page
- Finish page

3.4 Invocation

Package Management can be displayed to the user by two different ways:

- As a plug-in in the programming system
- As a standalone application

3.4.1 Programming System Plug-in

There is a menu command **Package Manager…** that is typically part of the **Tools** menu. The type GUID of the command implementation is `{7D20F8E2-A557-47ff-ABAB-70D417887AC3}`. After invocation, the Package Manager dialog is displayed.

3.4.2 Standalone Application

There is a standalone application called `PackageManager.exe`. This program must be executed within an Automation Platform infrastructure. The command line of this tool is as follows:

```
PackageManager.exe
   --profile="…"
   [--culture="…"]
   [--install="…" | --uninstall="…"]
```

Option	Description
--profile	Required. This option contains the name of the version profile that should be used for the Package Manager.
--culture	Optional. Specifying this option sets the language of the Package Manager. The value given here must be a valid culture identifier (e.g. `zh-CHS`). If the application is not localized for the specified culture, then the default language will be displayed.
--install	Optional. If this option is present, then the Package Manager immediately displays the Installation wizard for the specified package. After the installation, the application exits. The value of this option must be the file path to a valid package file. If neither –install nor –uninstall are specified, the Package Manager starts with the Package Manager dialog. If –install is specified, --uninstall must not be specified.
--uninstall	Optional. If this option is present, then the Package manager immediately display the Uninstallation wizard for the specified package. After the uninstallation, the application exits. The value of this option be in the format *`Id,Version`*, where Id is the package GUID (see p. 12), and Version is the version of the package (see p. 12). If neither –install nor –uninstall are specified, the Package Manager starts with the Package Manager dialog. If –uninstall is specified, --install must not be specified.

4 Design Considerations

The Package Manager maintains a local database which contains information about installed packages. This information is necessary to provide an “Uninstallation” feature to the user. In order to have a stable method of uninstallation, the implementation has been realized with the following basic principles in mind:

- All items are reference counted. That means, if two packages install the same item, then after uninstalling the first package the item will still be present, and after uninstalling the second package the item will be removed as well.
- If an item is already existing and has not been installed by any package, then it is never deleted by a package uninstallation.
- If an item has been installed by a package, then modified by some means, then it will not be removed by a package uninstallation any more. Only items that remain unchanged between installation and uninstallation will be removed.
- Package uninstallation will never restore an item to an older version, i.e. the version history of an item is not maintained.

To clarify things a bit, the pseudo code for the install and uninstall algorithms is presented here.

4.1 Install Algorithm Pseudo Code

For each item to be installed, the following algorithm is executed:

```
if (TargetItemExisting)
{
   if (!SourceAndTargetItemAreEqual)
   {
      OverwriteTargetItem;
      RememberSourceItemChecksum;
   }

   if (ReferenceCounterIsExisting)
      IncrementReferenceCounter;
   else
   {
      // The item is already existing, but was not created by a package.
      // Initializing the reference counter to a value of 2 causes that
      // package management will never delete this item because it
      // cannot be decrement to a value less than 1.
      InitializeReferenceCounter(2);
   }
}
else
{
   CreateTargetItem;
   RememberSourceItemChecksum;
   InitializeReferenceCounter(1);
}
```

4.2 Uninstall Algorithm Pseudo Code

For each item to be uninstalled, the following algorithm is executed:

```
if (TargetItemExisting)
{
   if (ReferenceCounter == 1)
   {
      if (SourceAndTargetItemAreEqual)
         DeleteTargetItem;

      DeleteReferenceCounter;
      ForgetTargetItemChecksum;
   }
   else
      DecrementReferenceCounter;
}
```

Change History

Version	Description	Editor	Date
0.1	Initial version	KeK	2010-01-26
1.0	Release after formal review	MN	2010-04-30
1.1	New features for V3.4 SP1: • 2.2.2.65, MenuConfiguration Element • 2.2.2.66, ToolbarConfiguration Element • 2.2.2.67, KeyboardConfiguration Element • 2.2.2.68, Profile Element (Below MenuConfiguration, ToolbarConfiguration, or KeyboardConfiguration Element)	KeK	2010-05-02
1.2	New features for V3.4 SP1: • 2.2.2.51, InsertionPath2 Element • 2.2.2.52, Popup Element • 2.2.2.54, InsertionPosition Element (Below Popup Element) • 2.2.2.56, PopupName Element • Corresponding adaptions in other sections	KeK	2010-05-04
1.3	New features for V3.4 SP3 (CDS-19539, CDS-19987) • 2.2.2.5167, Option Element • 2.2.2.68, Key Element • 2.2.2.69, Value Element1 • 2.2.2.5170, LibraryProfile Element	AM	2010-12-16
2.0	Release after formal review	MN	2011-01-28
2.1	New features for V3.4 SP4 (CDS-17579) • 2.2.2.73, VisualizationStyle Element	MP	2011-04-21
2.2	New feature for V3.4 SP4 (CDS-14898) • 2.2.2.74, VisualizationExtensions Element	StS	2011-07-06
3.0	Release after formal review	MaH	2011-07-12
3.1	New feature for V3.4 SP2 (CDS-17306) • 2.2.2.45, IgnoreArchiveFolder Element	MaH, (KeK)	2011-10-17
3.2	New feature for V3.4 SP3 (CDS-21917) • 2.2.2.45, TargetFolder Element	MaH, (KeK)	2011-10-18
3.3	New feature for V3.5 (CDS-16094) • 0, ExternalCall element • 2.2.2.76, Installation element • 2.2.2.77, Uninstallation element • 2.2.2.78, FileName element • 2.2.2.79, Arguments element	KeK	2011-10-25

Version	Description	Editor	Date
	• 2.2.2.80, CreateNoWindow element • 2.2.2.81, EnvironmentVariable element • 2.2.2.82, Exit element • 2.2.2.83, Code element • 2.2.2.84, IsError element • 2.2.2.85, Message element		
4.0	Release after formal review	MN	2011-11-30
4.1	New feature for V3.5 SP1 (CDS-19260, CDS-20473, CDS-20988, CDS-16096) • 2.2.2.86, Folder element • 2.2.2.87, ProfileSelectionList element • 2.2.2.88, Profile element • 2.2.2.89, ReadMe element	AM	2012-04-11
4.2	New feature for V3.5 SP1 (CDS-20986) • 2.2.2.86, Overwrite element	AM	2012-05-11
5.0	Release after formal review	MN	2012-06-11
5.1	Added descriptions of new environment variables: %DESKTOP%, %DESKTOP_DIRECTORY%, %APP_DATA_CODESYS%, and %COMMON_APP_DATA_CODESYS%.	KeK	2012-09-26
5.2	"CODESYS" instead of "CoDeSys" (CDS-29303)	MN	2012-10-02
6.0	Release after formal review	MN	2012-12-05
7.0	New feature for V3.5 SP3 (CDS-17199) • 2.2.2.40, CreateStartMenuLink element • 2.2.2.41, CreateDesktopLink element Reviewed and released	KeK	2013-01-30