NaturalONE

Lifecycle Manager

Version 8.2.7

March 2013

Document ID: LCM-DOC-827-20130320

Table of Contents

Preface ... vii
1 Release Notes ... 1
    What's New in Version 8.2.5 ... 2
    What's New in Version 8.2.6 ... 2
    What's New in Version 8.2.7 ... 3
2 Introduction ... 5
    What is the Lifecycle Manager? ... 6
    Big Picture ... 7
    Sample Lifecycle Model for Natural Projects ... 8
    Sample Lifecycle Model for Compound Projects ... 10
    Supported CentraSite Tools ... 11
    Further Reading ... 11
3 System Requirements ... 13
    Software Requirements ... 14
    Supported Operating System Platforms ... 15
    Browser Prerequisites ... 15
4 Installation and Configuration ... 17
    Issues to Consider before Implementation ... 18
    Required Permissions and Roles in CentraSite ... 18
    Before You Run the Installation Script ... 19
    Running the Installation Script ... 21
    Overview of Imported Lifecycle Manager Objects ... 25
    How to Proceed after the Installation? ... 26
5 First Steps ... 29
    About this Tutorial ... 30
    Creating the Deployment File ... 31
    Viewing the Sample Lifecycle Model for Natural Projects ... 31
    Activating the Sample Lifecycle Model ... 36
    Adding an Asset for a Natural Environment ... 37
    Adding an Asset for a Natural Project ... 40
    Copying the Deployment File to the Build Directory ... 45
    Browsing the Asset Catalog ... 45
    Activating the Policies ... 46
    Deploying the Assets According to the Sample Lifecycle Model for Natural Projects ... 47
    Creating a New Version of a Natural Project ... 51
    Downgrading to a Previous Version of a Natural Project ... 53
    Scheduling the Installation of a Natural Project ... 54
    Simulating the Installation of a Natural Project ... 57
    Uninstalling a Natural Project ... 59
6 Natural for Ajax Applications ... 61
    General Information ... 62
    Using Two Natural Projects ... 63

Using a Single Natural Project 63
Special Deployment Options 63
Adding an Asset for an Application Server 64
Adding an Association for an Application Server to a Natural Project 67
Actions for the User Interface Components 68
7 Asset Types for Natural Applications 71
Natural Environment 72
Natural Project 74
Compound Project 81
Application Server 83
8 Natural Project Settings for Different Scenarios 87
General Information 88
Processing the Sources from a Versioning Repository 88
Processing the Sources Directly in the Natural Environment 89
9 Version Handling 91
General Information 92
Installing the Natural Modules 93
Installing the User Interface Components 96
10 Sample Policies for Natural Applications 97
General Information 98
Sample Policies for Use with an NDV Server 98
Sample Policies for Use when an NDV Server is Not Available in the Target Environment 99
Sample Policies for General Use 100
11 Frequently Asked Questions 103
Is it possible to use the Lifecycle Manager without a versioning repository? 104
Is it possible to deploy Natural projects without a deployment file? 104
What is the structure of the build directory? 104
How can I find out what will be installed? 105
What happens if two installation actions try to install into the same target environment? 105
How can I back out the last promoted objects? 106
What is the purpose of the "SAG Demo Model Load Natural Project on a Separate Server"? 106
How can I activate tracing? 106
How can I trigger state changes without using CentraSite Control? 108
Where can I find additional information for a failed policy? 109
Can I generate reports for the Lifecycle Manager objects? 110
Can I use Predict Application Control for deployment? 110
12 Error Messages 111
Version of Natural project is not an integer 113
Error when accessing CentraSite to update an asset 113
Predecessor version of Natural project could not be determined 113
Association is in an invalid state 114
Could not copy a file 114

LCM property file could not be found 114
Deployment file was not found 115
Version directory could not be created 115
SYNIN file was not written 115
No object code file was written during the unload 116
Policy parameters are invalid 116
Error when accessing CentraSite 116
Natural project does not contain KeyToHistory 117
Error while processing the history file 117
Only higher version can be added to history file (current: %1$s; new version %2$s) 117
No "Next Installation Environment" defined and scheduled in Natural project 118
Project version %1$s has not been created. Current created version of the project is: %2$s 118
No environment defined for uninstallation 119
Project version %1$s cannot be installed because a version has never been created for this project 119
Problem occurred - check log files for information 119
Could not execute Ant - check your Ant installation 120
Could not execute SVN - check your SVN installation 120
Build directory could not be found 120
Version could not be determined 121
%1$s deployment file specified in the master deployment file 121
Project directory "%1$s" was not found 122
Natural for Ajax application file "%1$s" was not found 122
Update of the local SVN working copy failed 122
Checkout of a project failed 123
Failed to commit a local working copy to the SVN repository 123
Invalid repository URL specified 123
Specified value could not be found in the deployment file 124
Neither delete nor transfer directives are set 124

Preface

This documentation explains how to deploy Natural applications to different environments using the Lifecycle Manager. It is organized under the following headings:

Release Notes	Information on new features and enhancements.
Introduction	What is the Lifecycle Manager? About the delivered sample lifecycle models. Further reading.
System Requirements	Software requirements. Supported operating system platforms and browser prerequisites.
Installation and Configuration	Issues to be considered before implementing the Lifecycle Manager. How to run the installation script on the CentraSite machine, and how to proceed after the installation.
First Steps	How to get started with the Lifecycle Manager. You will learn how to deploy a traditional Natural application.
Natural for Ajax Applications	How to deploy a rich internet application which has been created using Ajax Developer.
Asset Types for Natural Applications	Information on the asset types that are delivered with the Lifecycle Manager.
Natural Project Settings for Different Scenarios	Required settings when the Lifecycle Manager takes the sources from a versioning repository, or when it takes the sources directly from a Natural environment.
Version Handling	What happens when you upgrade or downgrade to a full version, incremental version or hot fix?
Sample Policies for Natural Applications	An overview of the sample policies that delivered with the Lifecycle Manager.
Frequently Asked Questions	Answers to frequently asked questions.
Error Messages	An overview of the error messages that may be issued by the Lifecycle Manager.

1 Release Notes

- What's New in Version 8.2.5 2
- What's New in Version 8.2.6 2
- What's New in Version 8.2.7 3

What's New in Version 8.2.5

Installation

The *user.properties* file now contains the following new parameters:

`natural.batloc`

`natural.extlogloc`

Consequently, the required steps for the scheduling of Natural projects are now done by the installation script. You need no longer edit the file *LCM-Natural.prop* in order to adapt the values for the parameters `BATLOC`, `EXTLOGLOC` and `ATJOBS`.

For more information on the new parameters, see ***Before You Run the Installation Script***.

Forcing the Installation of a Hot Fix

It is now possible to force the installation of a hot fix without cleaning the environment by setting the `Force_Hotfix` property on the **Object-specific Properties** tab of your Natural project.

Simulating the Installation of a Natural Project

It is now possible to simulate the installation of a Natural project and list the affected objects in a report. The new **sample policy** "SAG Simulate Installation to Specific Environment" is available for this purpose. You can execute this new policy using the **Run Policy** action. For further information, see ***Simulating the Installation of a Natural Project***.

What's New in Version 8.2.6

Triggering State Changes Without Using CentraSite Control

The documentation now explains how to change the lifecycle state of a Natural Project asset using a script or a command. See ***How can I trigger state changes without using CentraSite Control?*** in *Frequently Asked Questions*.

What's New in Version 8.2.7

Lock Mechanism for Installations

When installing into a target environment, the CentraSite lock mechanism is now used to make sure that no other installation occurs in the same target environment at the same time. For further information, see ***What happens if two installation actions try to install into the same target environment?***.

2 Introduction

- What is the Lifecycle Manager? 6
- Big Picture 7
- Sample Lifecycle Model for Natural Projects 8
- Sample Lifecycle Model for Compound Projects 10
- Supported CentraSite Tools 11
- Further Reading 11

What is the Lifecycle Manager?

The Lifecycle Manager is used to deploy Natural applications (or specific versions of Natural applications) to different environments and to keep track of the installed applications in each environment. It is based on CentraSite and uses Natural projects. For this purpose, the Lifecycle Manager adds its own asset types to CentraSite. These asset types are:

- **Natural Project**
  Defines the Natural project, the location where the Natural project is to be built and additional attributes such as the installation type (full, incremental or hot fix). For each change in your source code (which may be either an error correction or new functionality), you create a new version of your Natural project.
- **Compound Project**
  Defines several Natural Project assets (and any other assets) that are to be installed together. Changes to an included Natural project may make it necessary to create a new version of the compound project.
- **Natural Environment**
  Defines the environment in which the Natural source objects (such as programs and subprograms) of a Natural project are to be installed.

In addition to the above asset types, the Lifecycle Manager also uses the following standard CentraSite asset type:

- **Application Server**
  Defines the application server environment in which the user interface components of a Natural project are to be installed. This asset type is only required if your Natural project refers to a Natural for Ajax application. It is not required for a traditional Natural application.

If you create your Natural projects with NaturalONE, you have the possibility to store your projects in a versioning repository such as Subversion, and you can deploy them using the deployment wizards of NaturalONE. A deployment wizard creates a deployment file. The Lifecycle Manager uses this deployment file, which is an Ant script, to start the deployment.

However, it is also possible to deploy your Natural projects without using a versioning repository. In this case, the Natural source objects are taken from a Natural system file. For details, see ***Is it possible to use the Lifecycle Manager without a versioning repository?*** in *Frequently Asked Questions*.

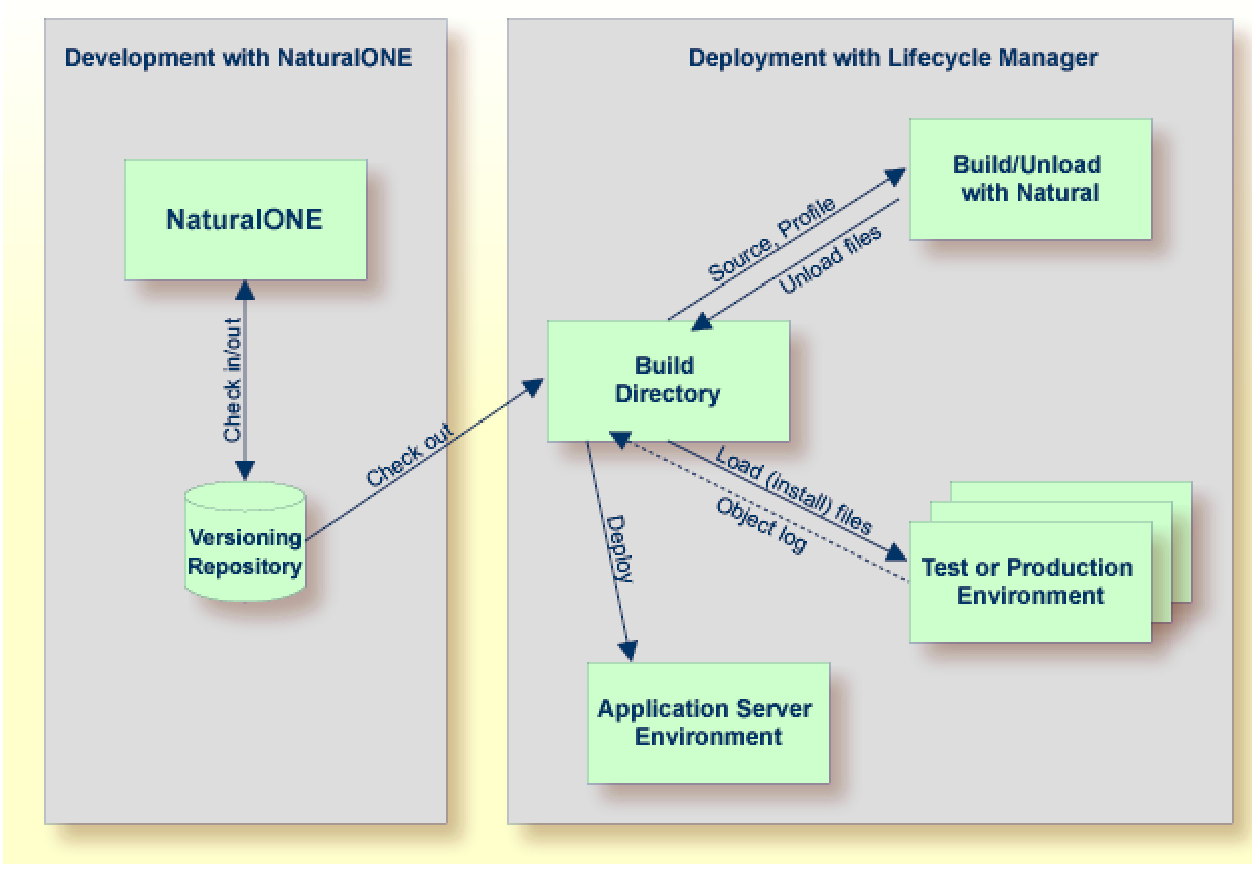
The Lifecycle Manager enables you to describe the distinct steps through which a Natural project passes on its way from conception to retirement. You model the lifecycle process associated with the above mentioned asset types and specify the events that are to be triggered when an asset makes a transition from one lifecycle state to another.

When using the Lifecycle Manager to deploy your Natural projects, you will

- define organizations, users and groups and assign roles to them,
- define Natural projects,
- define different state transitions for projects that are maintained by different development groups,
- define different versions of Natural projects,
- secure the above definitions,
- define which users/groups should be authorized to initiate a transition into a next state,
- trigger and control the necessary actions to deploy a Natural project,
- track and report which version of a Natural project exists in which state.

Big Picture

The following graphic shows the different components that can be used together with the Lifecycle Manager and how they interact.


Development with NaturalONE
NaturalONE
Check in/out
Versioning Repository
Check out
Deployment with Lifecycle Manager
Build/Unload with Natural
Source, Profile
Unload files
Build Directory
Load (install) files
Object log
Test or Production Environment
Deploy
Application Server Environment

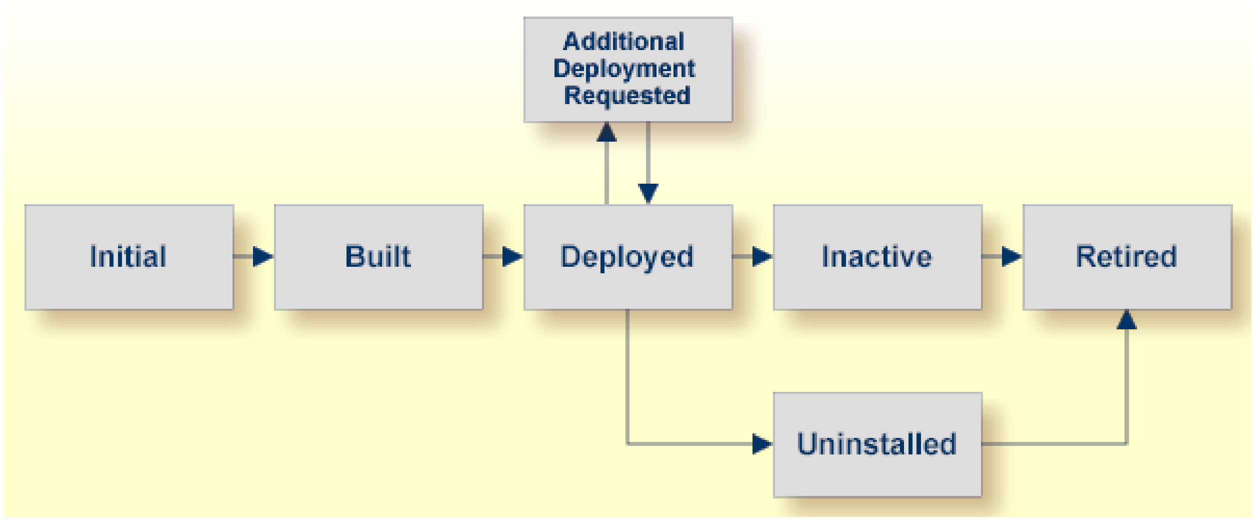
To develop a Natural project and to install it in a target environment, you proceed as follows:

1. You develop and test your sources using NaturalONE.
2. You commit your sources to your versioning repository.
3. Based on the changed sources you now specify which objects are to be promoted. To do so, you create a deployment file using NaturalONE and copy it to a build directory. The deployment file defines, for example, the versioning repository from which the sources are to be taken and the Natural server environment in which the project is to be compiled.
4. You change the lifecycle state of the Natural Project asset to "Built" in order to check out the sources from the versioning repository, catalog them in the Natural server environment, create a transfer file (unload), and put this transfer file into the build directory. If Natural for Ajax is involved, the user interface components are saved in a *.war* archive.
5. You define the Natural Project and Natural Environment assets, and an Application Server asset if Natural for Ajax is involved, in the Lifecycle Manager and specify the required information (such as the build directory and the target environments in which the project is to be installed).
6. You change the lifecycle state of the Natural Project asset to "Deployed" in order to install the generated programs in the target environments (that is, to load the transfer file and/or deploy the *.war* archive to the application server). This also creates a number of object log files in the build directory; these files contain detailed information on the installation process.

Note: If a versioning repository is not used, the source code is read from the Natural system file. However, when deploying the user interface components of a Natural for Ajax application, a versioning repository is a prerequisite.

Sample Lifecycle Model for Natural Projects

The Lifecycle Manager delivers a sample lifecycle model for Natural projects. You can use it to get started with the Lifecycle Manager and adapt it to your needs. Or you can create your own lifecycle model.



This sample lifecycle model supports the states described below.

The following states pertain to a specific version of a Natural project:

- **Initial**
  Initial state in the lifecycle of an asset of type "Natural Project".
- **Built**
  The outcome depends on whether your project pertains to a traditional Natural application (Natural modules only) or to a Natural for Ajax application (Natural modules and user interface components). For the Natural modules, this state means that the project has been built in the compile environment and that it has been unloaded into a transfer file. For the user interface components, this state means that the project has been built in the build environment and that either (depending on the settings in the WAR deployment file) the result or the deployment file has been copied to the *Version_<n>* directory.
- **Deployed**
  The Natural project has been deployed (that is, the contents of transfer file has been loaded or the user interface components have been copied) to at least one test or production environment.
- **Additional Deployment Requested**
  The Natural project needs to be deployed to at least one additional test or production environment.
- **Inactive**
  The Natural project is no longer active in any environment. This state is typically used if the version of the project is superseded by a newer version in all environments.
- **Retired**
  The Natural project is no longer used. All version-related files (transfer file, deployment file, *.war* archive) have been removed from the *Version_<n>* directory.

The following is a special state which pertains to the entire Natural project (not only to a specific version):

- **Uninstalled**
  The Natural project and all predecessor versions have been deleted from all environments.

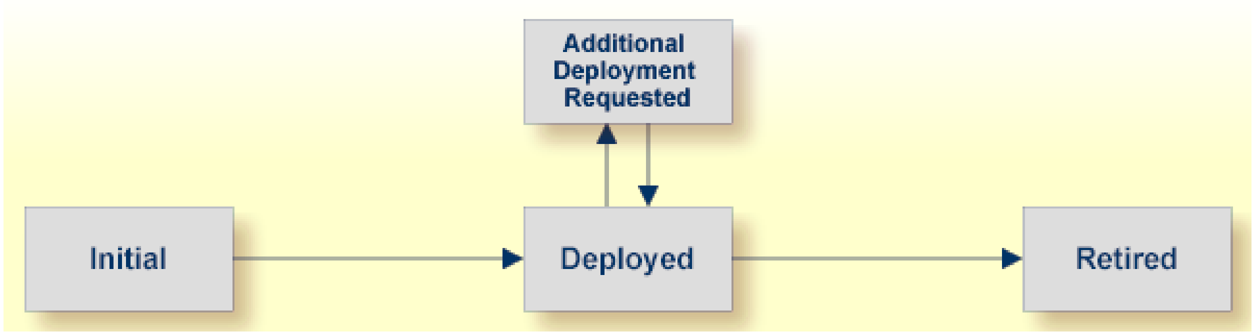
For background information on topics such as deployment strategy, permissions, lifecycle models or policies, see *Implementation Concepts* which is part of the CentraSite documentation.

Sample Lifecycle Model for Compound Projects

A compound project logically connects different Natural projects that are to be installed together. Other than a Natural project, a compound project initiates neither a build nor the distribution of the projects. It just tracks the actions performed by the Natural projects.

Compound projects allow you to maintain different projects from your versioning repository as a single project. This is helpful, for example, if you have steplibs which are defined in different Natural projects.

The Lifecycle Manager delivers a sample lifecycle model for compound projects. You can adapt it to your needs. Or you can create your own lifecycle model.



This sample lifecycle model supports the following states:

- **Initial**
  Initial state in the lifecycle of an asset of type "Compound Project".
- **Deployed**
  All Natural projects that are defined in the compound project have been deployed to at least one test or production environment.

- **Additional Deployment Requested**
  All Natural projects that are defined in the compound project need to be deployed to at least one additional test or production environment.
- **Retired**
  The compound project is no longer used.

Supported CentraSite Tools

The Lifecycle Manager is fully integrated into CentraSite. You can use the following CentraSite tools:

- **CentraSite Control**
  Provides access to the Lifecycle Manager's full range of features and capabilities.
- **CentraSite Plug-ins for Eclipse**
  These plug-in are helpful for developers who want to use NaturalONE and the Lifecycle Manager from within the same application, the so-called Software AG Designer. However, the plug-ins provide only limited administrative functionality.

Further Reading

How to get started with CentraSite or NaturalONE is not in the scope of this documentation. It is recommended that you read the corresponding product documentation prior to reading this Lifecycle Manager documentation. Basic information (such as concepts or use of the CentraSite editors) is not repeated in this documentation.

Helpful topics in the CentraSite documentation concerning lifecycle management are:

- *Implementation Concepts*
- *Customizing Lifecycle Management*

The most up-to-date CentraSite documentation is always available at ***http://documentation.softwareag.com/webmethods/centrasite.htm***.

If you are a registered user, you can find the complete volume of Software AG's product documentation in Empower, Software AG's global extranet, at ***https://empower.softwareag.com/***.

3 System Requirements

- Software Requirements 14
- Supported Operating System Platforms 15
- Browser Prerequisites 15

Software Requirements

The Lifecycle Manager requires the following:

- CentraSite ActiveSOA Version 8.2 on Windows or Linux. This must be installed on the server.
- NaturalONE Version 8.2. This can be installed anywhere you like.
- Natural Development Server and Natural. For information on the recommended versions, see *Using an Existing Natural Development Server Environment* in the *Installation* documentation of NaturalONE.
- Apache Ant 1.7.1 or above and Ant Contrib 1.0b3 or above. Ant Contrib is required by the installation script only. Both must be installed on the same machine as CentraSite.
- Java Development Kit (JDK) 1.5.0_12 or above. This is required by the installation script. The JDK must be installed on the same machine as CentraSite.

  Alternatively, you can also use the Java Runtime Environment (JRE) 1.5.0_12 or above. In this case, however, the file *tools.jar* must be available in your Java environment. You can use, for example, the *tools.jar* file of the JVM which is installed together with CentraSite.

Optional. If you keep your sources in a versioning repository, the following is also required:

- Versioning repository. The Lifecycle Manager supports the same versioning repositories as NaturalONE: Subversion and CVS.
- Either the Subversion command line tool 1.5.2 or above, or the CVS command line tool 1.11 or above. This must be installed on the same machine as CentraSite.
- Java Runtime Environment (JRE) 1.5.0_12 or above. This must be installed on the same machine as CentraSite.

Optional. If you want to deploy Natural for Ajax applications to a Natural for Ajax environment on an application server or servlet container, the following is also required:

- An SFTP (Secure File Transfer Protocol) server which is capable of private-key authentication with write-access to the web application folder of your application server.

  On Linux, you can use sshd which is a common server capable of SFTP.

  On Windows, you can use Cygwin together with either OpenSSH or CopSSH.
- Optional. If you are using the option **Deploy as file export** in the NaturalONE deployment wizard for web applications, the following must be installed on the machine on which your application server is running and it must be accessible:
  - Apache Ant 1.7.1.
  - Java Runtime Environment (JRE) 1.5.0_12 or above.

- Either the Subversion command line tool 1.5.2 or above, or the CVS command line tool 1.11 or above.

Note: These are the same prerequisites as for deploying Natural applications. See also *Deploying Natural Applications* in *Using NaturalONE*.

Supported Operating System Platforms

Since the Lifecycle Manager is based on CentraSite, it supports the same operating system platforms as CentraSite.

If you want to use the Lifecycle Manager together with NaturalONE (in Eclipse), be aware that NaturalONE supports only a subset of the operating system platforms supported by CentraSite.

Browser Prerequisites

Since the Lifecycle Manager is based on CentraSite, it supports the same browsers as CentraSite.

4 Installation and Configuration

- Issues to Consider before Implementation 18
- Required Permissions and Roles in CentraSite 18
- Before You Run the Installation Script 19
- Running the Installation Script 21
- Overview of Imported Lifecycle Manager Objects 25
- How to Proceed after the Installation? 26

Issues to Consider before Implementation

The administrator has to configure CentraSite for use with the Lifecycle Manager. Several issues have to be considered before you can begin your implementation. For the Lifecycle Manager, some of these issues are:

- Which organizations need to be defined?
- Which users/groups need to be assigned to which organizations?
- Which groups and roles need to be defined?
- Which states are required in a lifecycle model in order to develop, test and deploy Natural projects?
- Which policies and actions are required for the different states of a lifecycle model?
- Who is to be authorized to change the lifecycle state of a Natural project (for example, from "Built" to "Deployed")?
- Which Natural environments need to be defined?
- Which Natural projects need to be deployed into which Natural environments?
- Which compound projects need to be defined?

Before you start configuring CentraSite for use with the Lifecycle Manager, it is recommended that you first read *Implementation Concepts* which is part of the CentraSite documentation.

The topics below just contain brief information of what needs to be done in order to configure the Lifecycle Manager. For detailed information, see the CentraSite documentation.

Required Permissions and Roles in CentraSite

In order to install the Lifecycle Manager, you need the following roles in CentraSite:

- Asset Type Administrator
- Policy Administrator

In addition, you must have administrator rights for the System Management Hub.

Before You Run the Installation Script

The Lifecycle Manager requires that CentraSite ActiveSOA is already installed.

You install the Lifecycle Manager together with NaturalONE using the Software AG Installer. The installer copies several files to your NaturalONE installation, including an installation script which you have to run on the machine on which CentraSite is installed (more detailed information is provided below).

Before you run the installation script for the Lifecycle Manager, which imports all required files into CentraSite, proceed as described below.

In System Management Hub

The topics referenced below can be found in the CentraSite documentation.

- Configure the authentication mechanism to be used (Microsoft Active Directory or LDAP). See *Authentication Topics and LDAP*.
- Make sure that an appropriate email server is defined (to be used, for example, with the "Send Email Notification" policy action). See *Basic Operations* > *Configuring the Email Server*.
- Configure the log settings and allow **Policy Log** with the option **Log Failure and Success**. See *Logging* > *Managing Log Configuration*.
- Set the transaction time to its maximum. See *Basic Operations* > *Repository Configuration Parameters*.
- ⚠ **Important:** It is strongly recommended that you back up the CentraSite repository before you run the installation script for the Lifecycle Manager. See *Basic Operations* > *Backing up the Database*.

On the same machine as CentraSite

When you install the Lifecycle Manager with the Software AG Installer, a directory with the name *lcm* is created in the directory which contains your NaturalONE installation.

1. The installation script for the Lifecycle Manager must run on the same machine on which CentraSite has been installed. Therefore, make sure that the *lcm* directory can be accessed from that machine. If the *lcm* directory is not accessible from the machine on which CentraSite has been installed, copy it to that machine.
2. Adapt the installation properties file *user.properties* to your environment settings. You can find this file in the *LcmInstallerScript* subdirectory of the *lcm* directory. This file has the following content:

```
#Path to Software AG root directory
softwareag.directory=       USER_DEFINED

#Directory containing the files to be installed
installation.directory=     USER_DEFINED

#Name of license file for Natural Runtime
natural.license=            USER_DEFINED

#Name of license information file for Natural Runtime
natural.license.info=       USER_DEFINED

#URL of the CentraSite server
centrasite.repos.url=       USER_DEFINED

#User for which Lifecycle Manager objects are to be imported
centrasite.user.name=       USER_DEFINED

#Password for above user
centrasite.user.password=   USER_DEFINED

######
#The three parameters below are only required for scheduling
#the installation of a Natural project:
######

#Optional. Linux only. Path to "at" jobs directory
natural.atjobs=USER_DEFINED

#Optional. Directory containing the command files for changing the lifecycle
#state with the scheduling tool schtasks (Windows) or at (Linux).
natural.batloc= USER_DEFINED

#Optional. Directory containing the protocol files which result from the command
#files that are defined with natural.batloc.
natural.extlogloc=USER_DEFINED
```

Substitute the string `USER_DEFINED` with the appropriate information as described in the following table:

Parameter	**Description**
`softwareag.directory`	Path to the Software AG root directory in which the CentraSite has been installed, and into which the Lifecycle Manager will also be installed. By default, this is *C:/SoftwareAG/* on Windows and */opt/SoftwareAG/* on Linux. It is important that you also enter a slash at the end of the path.
`installation.directory`	Directory containing the Lifecycle Manager files to be installed. This is the content of the *lcm* directory which has been created during the NaturalONE installation. By default, this is *C:/SoftwareAG/lcm/* on Windows and */opt/SoftwareAG/lcm/* on Linux. It is important that you also enter a slash at the end of the path.

Parameter	Description
`natural.license`	Name of the license file for Natural Runtime. The Lifecycle Manager is delivered with a license file. Its name is *run*`<nn>`*.xml*, where `<nn>` stands for the version number. You can find this file in the *lcm* directory.
`natural.license.info`	Name of the PDF file containing important license information for Natural Runtime. Its name is *run*`<nn>`*.pdf*, where `<nn>` stands for the version number. You can find this file in the *lcm* directory.
`centrasite.repos.url`	URL of the CentraSite server. By default, this is *http://localhost:53305/CentraSite/CentraSite*.
`centrasite.user.name`	Name of the user for which the Lifecycle Manager objects are to be imported.
`centrasite.user.password`	Password for the above user.
`natural.atjobs`	Optional. Linux only. Path to the directory containing the `at` jobs. For example, */var/spool/atjobs*.
`natural.batloc`	Optional. Directory containing the command files for changing the lifecycle state with the scheduling tool `schtasks` (Windows) or `at` (Linux). See also ***Scheduling the Installation of a Natural Project***.
`natural.extlogloc`	Optional. Directory containing the protocol files which result from the command files that are defined with `natural.batloc`.

The parameters `natural.batloc`, `natural.extlogloc` and `natural.atjobs` are only required for scheduling the installation of a Natural project.

The installation script uses the information that you specify in the file *user.properties* to create the file *LCM-Natural.prop* (see below).

Important: When specifying a path for Windows, you have use a slash (/) instead of a backslash (\).

Running the Installation Script

The name of the installation script is *install.xml*. You can find this file in the *LcmInstallerScript* subdirectory of the *lcm* directory.

Notes:

1. In order to run the installation script, make sure that Ant, Ant Contrib and the Java Development Kit (JDK) can be accessed.
2. Make sure that the environment variable `JAVA_HOME` points to your JDK.

If you want to print the help screen of the Ant script, issue the following call:

```
ant -lib path-to-mylib -f install.xml help
```

When all prerequisites are in place, you can start the installation by issuing the following Ant call:

```
ant -lib path-to-mylib -f install.xml install
```

With the above call, the logging information is written to standard output. If logging information is to be written to a file, use a call such as the following:

```
ant -lib path-to-mylib -f install.xml install -logfile mylogfile.txt
```

Note: `path-to-mylib` in the above calls stands for the path to Ant Contrib.

The installation script performs the following steps (you can use these steps as a checklist if an error occurs while running the installation script):

1. Copies the file *CentraSiteRepairsKitPack_NaturalOneFixes.zip* into CentraSite's *trs/xtensions* directory (for example, *C:\SoftwareAG\CentraSite\rts\xtensions*).
2. Copies the *com.softwareag.lcm.plugin.schedule* directory into CentraSite's *webapps/PluggableUI* directory (for example, *C:\SoftwareAG\profiles\CTP\workspace\webapps\PluggableUI*).
3. Creates a properties file with the name *LCM-Natural.prop* which contains your environment settings. To create this file, the installation script uses the information that you specified in the file *user.properties*. The file *LCM-Natural.prop* must have the following content, depending on your operating system:

   Windows:

   ```
   NATROOT=softwareag-directory/LifecycleManager/win-32
   NATUSERDATA=softwareag-directory/LifecycleManager/Pseudo-.naturalone
   NATLICDETAILS=softwareag-directory/LifecycleManager/name-of-license-file.xml
   NATEXEC=softwareag-directory/LifecycleManager/win-32/bin/natrt.exe
   NATANTLIB=lcm-directory
   BATLOC=any-directory
   EXTLOGLOC=any-directory
   INMHOME=softwareag-directory/Centrasite
   ```

   Linux:

   ```
   NATROOT=softwareag-directory/LifecycleManager/Linux_x64
   NATUSERDATA=softwareag-directory/LifecycleManager/Pseudo-.naturalone
   NATLICDETAILS=softwareag-directory/LifecycleManager/name-of-license-file.xml
   NATEXEC=softwareag-directory/LifecycleManager/Linux_x64/bin/natrt
   NATANTLIB=lcm-directory
   BATLOC=any-directory
   EXTLOGLOC=any-directory
   INMHOME=softwareag-directory/Centrasite
   ATJOBS=directory-containing-at-jobs
   ```

where:

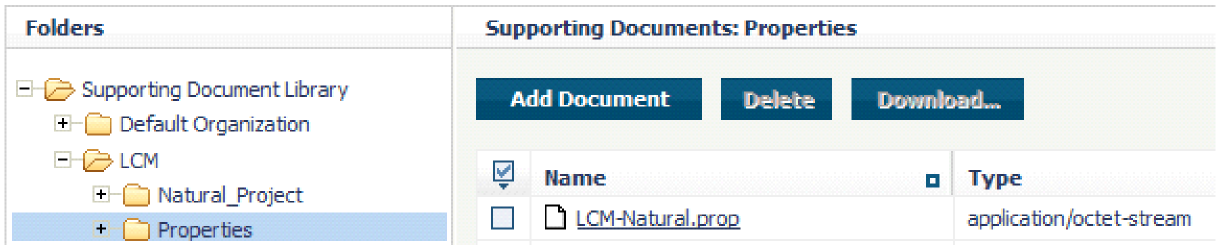
- `softwareag-directory` is the path to the Software AG root directory. By default, this is *C:/SoftwareAG* on Windows and */opt/SoftwareAG* on Linux.
- `lcm-directory` is the path to the directory with the name *lcm* which contains the Lifecycle Manager files to be installed.

Note: If the string `USER_DEFINED` has not been substituted with the appropriate information in the *user.properties* file, the file *LCM-Natural.prop* also contains this string. If required (for example, when nothing has initially been specified for the optional scheduling parameters), you need to adapt your file later manually.

Parameter	Specifies the path to
`NATROOT`	Natural Runtime directory. By default, this is `<softwareag-directory>`*/LifecycleManager/win-32* (Windows) or `<softwareag-directory>`*/LifecycleManager/Linux_x64* (Linux), where the path to the `<softwareag-directory>` has been defined with `softwareag.directory` in the file *user.properties*.
`NATUSERDATA`	`FUSER` location. By default, this is `<softwareag-directory>`*/LifecycleManager/Pseudo-.natural*, where the path to the `<softwareag-directory>` has been defined with `softwareag.directory` in the file *user.properties*.
`NATLICDETAILS`	License file for Natural Runtime. By default, this is the name which has been defined with `natural.license` in the file *user.properties*.
`NATEXEC`	File which starts the Natural Runtime. By default, this is *bin/natrt.exe* (Windows) or *bin/natrt* (Linux) in the Natural Runtime directory that is defined by `NATROOT` (see above).
`NATANTLIB`	Directory containing the JAR files which are required for deploying Natural applications. By default, this is the directory which has been defined with `installation.directory` in the file *user.properties*. See also the prerequisites in *Deploying Natural Applications* which is part of *Using NaturalONE*.
`BATLOC`	Directory containing the command files for changing the lifecycle state with the scheduling tool `schtasks` (Windows) or `at` (Linux). By default, this is the directory which has been defined with `natural.batloc` in the file *user.properties*. See also ***Scheduling the Installation of a Natural Project***.
`EXTLOGLOC`	Directory containing the protocol files which result from the command files that are defined with `BATLOC`. By default, this is the directory which has been defined with `natural.extlogloc` in the file *user.properties*.
`INMHOME`	Directory containing the CentraSite installation. By default, this is *C:/SoftwareAG/Centrasite* on Windows and */opt/SoftwareAG/Centrasite* on Linux.
`ATJOBS`	Linux only. Directory containing the `at` jobs. For example, */var/spool/atjobs*. By default, this is the directory which has been defined with `natural.atjobs` in the file *user.properties*.

4. Starts CentraSite Control, and adds a new folder with the name *LCM* under **Asset Catalog > Supporting Documents**.
5. Adds two subfolders with the following names in the new *LCM* folder: *Natural_Project* and *Properties*.
6. Uploads the newly created *LCM-Natural.prop* file into Centrasite WebDAV.

   When you go to the new *Properties* subfolder, you can see that *LCM-Natural.prop* has been added as a document:



Name	Type
LCM-Natural.prop	application/octet-stream

7. Imports a number of archive files (zip files) from the *lcm/MetaDataExport* directory into CentraSite. These files contain the objects (asset types, sample lifecycle models, sample policies and action templates) that are delivered by the Lifecycle Manager.

   The archive files are imported in the following sequence (problems may occur if they are imported in a different sequence):

   LCM-Association-export.zip
   LCM-Natural-Environment.zip
   LCM-Natural-Project.zip
   LCM-Compound-Project.zip
   LCM-SAG-Demo-Model-for-Compound-Projects.zip
   LCM-SAG-Demo-Model-for-Natural-Projects.zip
   LCM-SAG-Demo-Model-Load-Natural-Project-on-a-Separate-Server.zip
   LCM-Mandatory-Policies.zip
   LCM-SAG-Policies-for-Compound-Projects.zip
   LCM-SAG-Policies-for-Natural-Projects.zip
   LCM-SAG-Policies-for-Natural-Projects-on-a-Separate-Server.zip

   Note: With the installation script, it is not possible to change the organization. The objects are always imported into the "Default Organization". In addition, the installation script uses the import option **Allow replace of registry objects**. This option is important for the Lifecycle Manager. The structures of all imported objects are internally linked, and the structures will only stay consistent when this option is selected.

8. Activates the mandatory policies which are contained in the archive file *LCM-Mandatory-Policies.zip*. These policies ensure the integrity of your lifecycle models. They are executed independently of any lifecycle models that you define.

The mandatory policies are:

SAG Initialize Natural Project
SAG Mark Last Project Version for Deletion
SAG Remove Natural Project
SAG Validate Associations
SAG Validate Deletion of Natural Project
SAG Validate Inactive Natural Project

You can see these policies when you select **Policies > Design/Change-Time**.

Overview of Imported Lifecycle Manager Objects

After you have run the installation script, the objects listed below are available in CentraSite.

Note: You may have to refresh the display to see the new objects.

Asset Types

The following asset types are visible when you select **Administration > Types**:

- Compound Project
- Natural Environment
- Natural Project

Sample Lifecycle Models

The following sample lifecycle models are visible when you select **Administration > Lifecycles > Models**:

- SAG Demo Model for Compound Projects
- SAG Demo Model for Natural Projects
- SAG Demo Model Load Natural Project on a Separate Server

Sample Policies

Sample policies for Natural applications are visible (in addition to the mandatory policies) when you select **Policies > Design/Change-Time**. The names of these policies start with "SAG".

A brief functional description of each sample policy is provided on the **Design/Change-Time Policies** page. A more detailed description can be found when you display the policy. See also *Sample Policies for Natural Applications*.

Action Templates

Action templates are visible when you select **Policies > Action Templates**. To see them, you have to click the arrow which is shown in front of **Natural**. The names of these action templates start with "SAG". The above mentioned sample policies use these action templates.

A brief functional description of each action template is provided on the **Action Templates** page. A more detailed description can be found when you display the action template.

How to Proceed after the Installation?

After you have run the installation script for the Lifecycle Manager, proceed as described below.

In NaturalONE

1. Make sure that your project works as intended. For example, use the **Build Natural Project** command to update your Natural environment and then test the project in the Natural environment.
2. Make sure that all required files of your project have been committed to your versioning repository.
3. Create the deployment file.

For detailed information, see the core documentation of NaturalONE, especially the information in *Using NaturalONE*.

If you are working on Linux

- Make sure that ANT and SVN can be accessed by the Software AG Common Tomcat server. To do so, enter their locations in the `PATH` environment variable of the startup script */etc/init.d/sag*`<n>`*ctp*`<nn>`, where `<n>` stands for the installation number of the Tomcat server and `<nn>` stands for the version number of the suite.

Optional. If you want to set up a schedule for a Natural project so that the project is automatically installed at a predefined date and time (see also ***Scheduling the Installation of a Natural Project***), the following is also required:

- Make sure that the account under which the Tomcat server runs (normally, this is the root account) has the permission to execute the `at` command.
- Make sure that the `at` daemons have been started.

In Natural Security

The following is required if Natural Security is active in your Natural server environment:

- Many Lifecycle Manager actions use the Natural Object Handler which stores reports and intermediate results in a workplan library. By default, the name of this library is `WORKPLAN`. So that the reports can be unloaded, make sure that the workplan library is allowed in Natural Security.

  For detailed information, see *Natural Security* in the Natural documentation for your platform, especially the information on the Object Handler utility profiles in the section *Protecting Utilities*.

In CentraSite

Important: Before you proceed as described below, make sure that all mandatory policies have been activated (see the corresponding list under ***Running the Installation Script***).

1. Define a new lifecycle model (including all states, policies and actions) which is associated with the Natural Project asset type.
2. Activate the lifecycle model and the policies.
3. Add an asset of type "Natural Environment" and specify the environment attributes.
4. Add an asset of type "Natural Project", specify the name of your Natural Environment asset in the attribute **To be installed next**, and specify all required information such as build directory, deployment file and transfer directives.
5. Copy the deployment file that has been created with NaturalONE into the build directory that is defined in your Natural Project asset.

These steps are explained in more detail in ***First Steps***.

For Natural for Ajax applications on an application server

- You need a public/private key pair for authentication. The private key must be located on the CentraSite machine. The public key must be added to the *authorized_keys* file of the SSH server. Private key authentication must be enabled on the SSH server.

  Note: On Windows, you can use PuTTYgen to generate the keys. On Linux, you can use ssh-keygen.

  For further information, see ***Natural for Ajax Applications***.

5 First Steps

- About this Tutorial ... 30
- Creating the Deployment File ... 31
- Viewing the Sample Lifecycle Model for Natural Projects ... 31
- Activating the Sample Lifecycle Model ... 36
- Adding an Asset for a Natural Environment ... 37
- Adding an Asset for a Natural Project ... 40
- Copying the Deployment File to the Build Directory ... 45
- Browsing the Asset Catalog ... 45
- Activating the Policies ... 46
- Deploying the Assets According to the Sample Lifecycle Model for Natural Projects ... 47
- Creating a New Version of a Natural Project ... 51
- Downgrading to a Previous Version of a Natural Project ... 53
- Scheduling the Installation of a Natural Project ... 54
- Simulating the Installation of a Natural Project ... 57
- Uninstalling a Natural Project ... 59

About this Tutorial

This tutorial is intended for administrators who want to get started with the Lifecycle Manager. It covers all basic steps you have to perform in order to work with a project. You will proceed as follows:

1. create a deployment file (this tutorial assumes that a Natural deployment file is available that has been created using NaturalONE),
2. view the sample lifecycle model for Natural projects that is delivered with the Lifecycle Manager (the lifecycle model that is used in this tutorial assumes that a Natural development server is available),
3. activate the lifecycle model,
4. add the required Natural Environment and Natural Project assets,
5. copy the deployment file to the build directory,
6. activate the policies that are to be used when an asset changes its lifecycle state (for example, from "Built" to "Deployed"),
7. change the lifecycle state of the Natural project,
8. create a new version of the Natural project,
9. downgrade to a previous version of the Natural project,
10. schedule an automatic installation of the Natural project,
11. find out which modules will be installed into a specific environment by simulating the installation of a Natural project,
12. uninstall a Natural project from a specific environment.

It is important that you work through the exercises in the same sequence as they appear in this tutorial.

This tutorial assumes that you have a user account on the instance of CentraSite with which you want to work. For detailed information, see the CentraSite documentation.

Important: Before you begin with this tutorial, make sure that you have properly configured the Lifecycle Manager as described in the section ***Installation and Configuration***.

Creating the Deployment File

In this tutorial, you will learn how to deploy a traditional Natural application. Therefore, you will use a deployment file which has been created using NaturalONE's deployment wizard for Natural applications. The deployment file is an Ant script which is used to start the deployment process. This file is always created for a single project. It defines, for example, the versioning repository from which the sources are to be taken and the Natural server environment in which the project is to be compiled.

When you want to deploy your project for the very first time, you enable the full deployment mode in the wizard. For later updates of your deployed project, it is sufficient to enable the incremental deployment mode in the wizard (this is the correct setting for both, incremental versions and hot fixes).

For detailed information on how to create a deployment file, see *Deploying Natural Applications* in *Using NaturalONE*.

Note: The Lifecycle Manager can also use other types of deployment files that have been created using NaturalONE, for example, a WAR deployment file which is used to deploy Natural for Ajax applications. This type of deployment is not covered in this tutorial. It is described in the section *Natural for Ajax Applications*.

Viewing the Sample Lifecycle Model for Natural Projects

For the purpose of this tutorial, you will look at the states, state transitions, policies and actions that have been defined for the sample lifecycle model for projects that is delivered with the Lifecycle Manager.

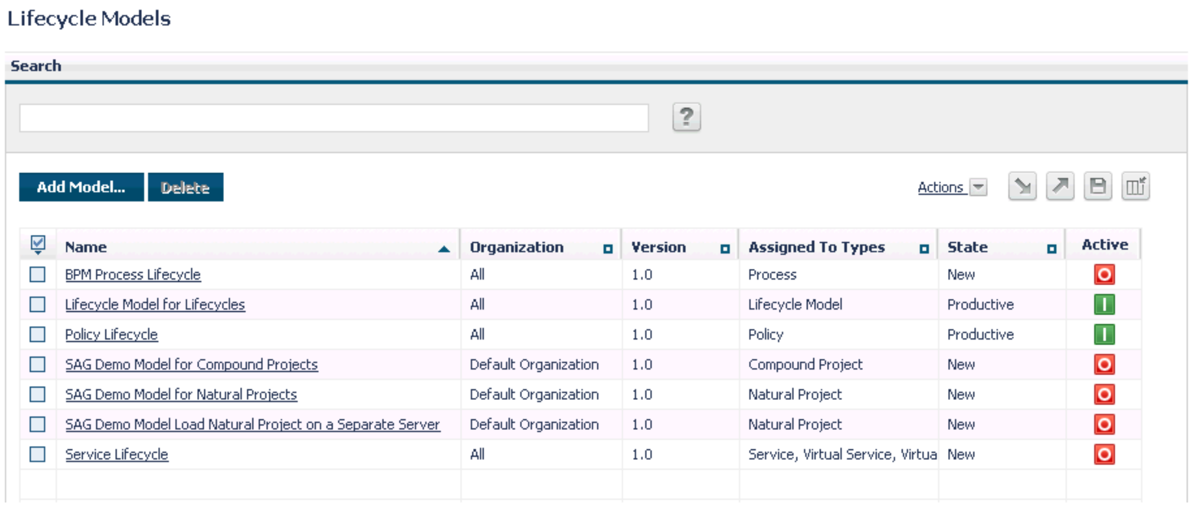
Note: If you want to create your own lifecycle model, see *Customizing Lifecycle Management* in the CentraSite documentation.

To view the sample lifecycle model

1 Start CentraSite Control.

2 Go to **Administration > Lifecycles > Models**.

The names of all available lifecycle models are shown. This also includes the sample lifecycle models that are delivered with the Lifecycle Manager. The names of these sample models start with "SAG Demo Model". This tutorial will use the sample lifecycle model named "SAG Demo Model for Natural Projects".



Lifecycle Models

Search

Add Model... Delete

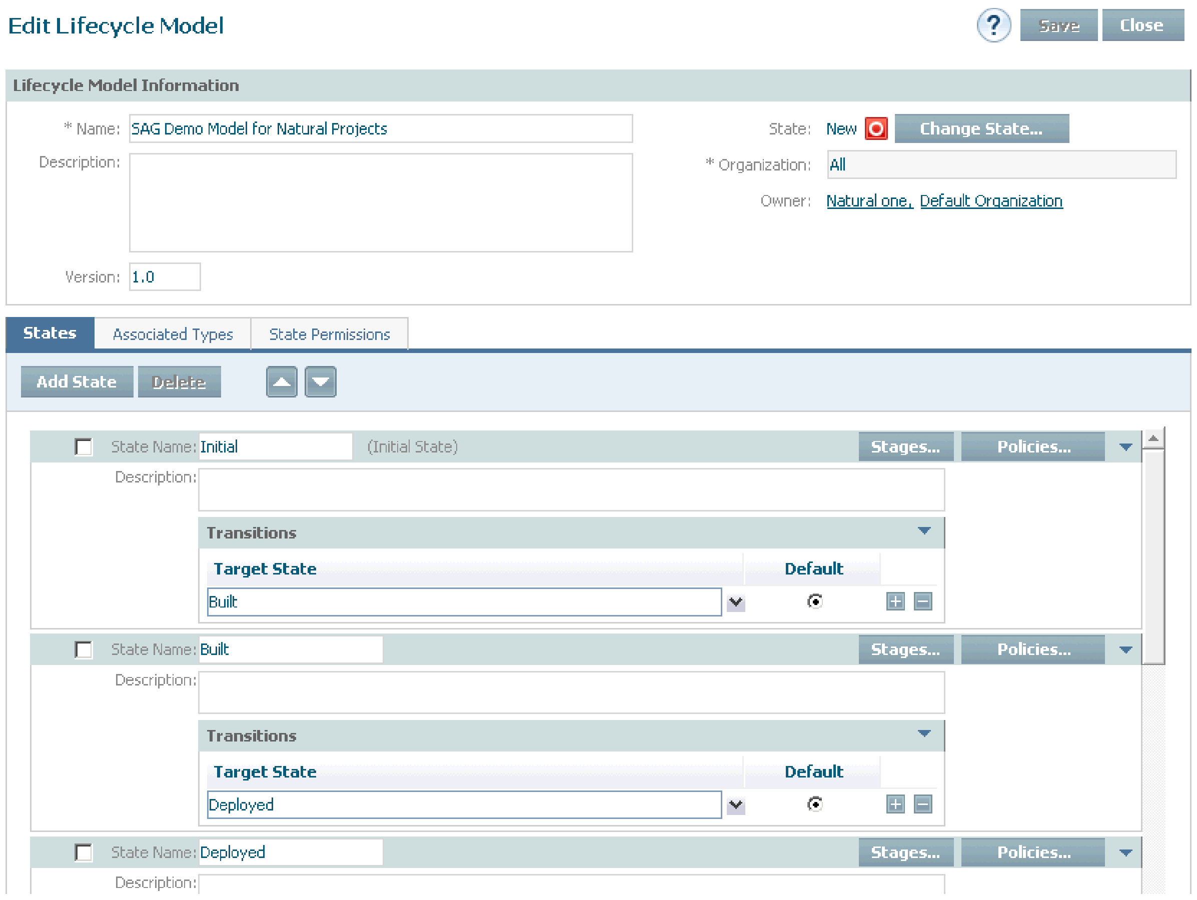
Actions

Name	Organization	Version	Assigned To Types	State	Active
BPM Process Lifecycle	All	1.0	Process	New	🟥
Lifecycle Model for Lifecycles	All	1.0	Lifecycle Model	Productive	🟩
Policy Lifecycle	All	1.0	Policy	Productive	🟩
SAG Demo Model for Compound Projects	Default Organization	1.0	Compound Project	New	🟥
SAG Demo Model for Natural Projects	Default Organization	1.0	Natural Project	New	🟥
SAG Demo Model Load Natural Project on a Separate Server	Default Organization	1.0	Natural Project	New	🟥
Service Lifecycle	All	1.0	Service, Virtual Service, Virtua	New	🟥

By default, the sample lifecycle models that are delivered with the Lifecycle Manager are inactive. This is indicated by a red icon in the **Active** column. As long as a lifecycle model is inactive, it can be modified. You will activate the lifecycle model later in this tutorial.

3 Click the sample lifecycle model named "SAG Demo Model for Natural Projects".

The details of this sample lifecycle model are shown.



* Name: SAG Demo Model for Natural Projects

Description:

Version: 1.0

State: New

* Organization: All

Owner: Natural one, Default Organization

State Name: Initial (Initial State)

Description:

Target State	Default
Built	●

State Name: Built

Description:

Target State	Default
Deployed	●

State Name: Deployed

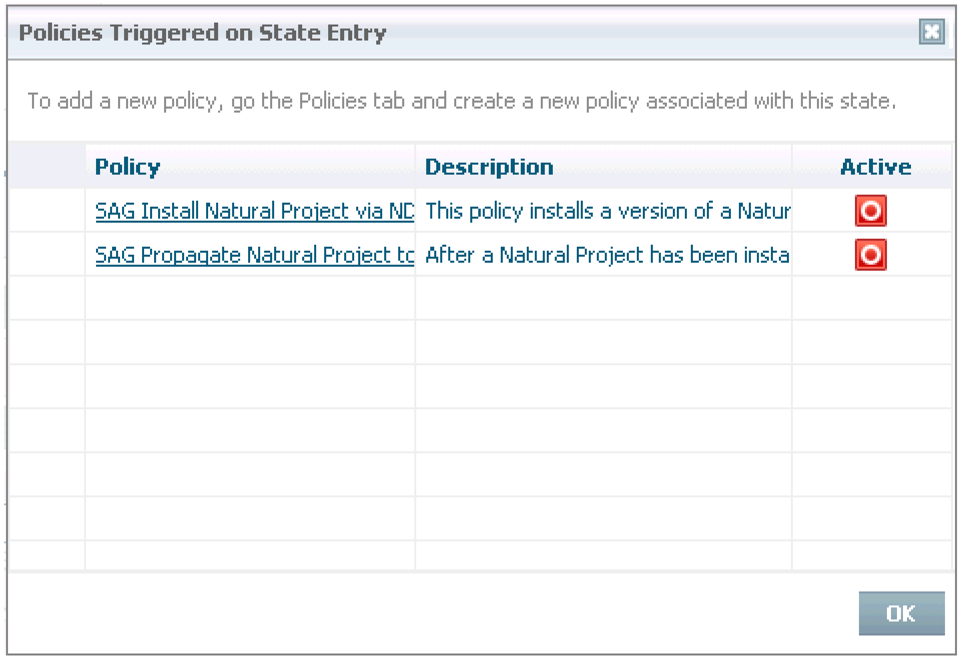
Description:

At the bottom of the page, you can see the different states ("Initial", "Built", etc.) that are defined for this sample lifecycle model. For each state except the "Retired" state, at least one transition to a target state is defined.

When you scroll down in the list of states, you can see that more than one target state is defined for the "Deployed" state. One of these states is marked as the default state. This is the state for which the option button will be selected by default when the user later changes the state in a dialog box.

4 To view the policies that are to be executed when a Natural project enters a specific state, click the **Policies** button which is located next to this state.

For example, the following information is shown when you view the policies for the "Deployed" state.



Policies Triggered on State Entry

To add a new policy, go the Policies tab and create a new policy associated with this state.

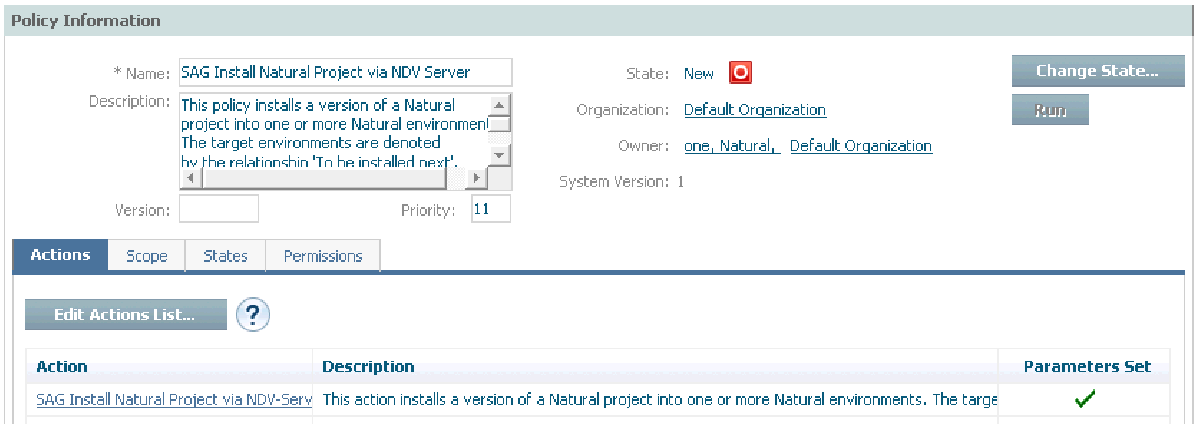
	Policy	Description	Active
	SAG Install Natural Project via ND	This policy installs a version of a Natur	🟥
	SAG Propagate Natural Project tc	After a Natural Project has been insta	🟥

OK

When a yellow warning icon is shown in the first column, this indicates that information is still missing for this policy (for example, the parameters of at least one action that is used in this policy are not set).

5 To view the actions that are to be executed with a policy, click the policy name.

For example, when you click the policy "SAG Install Natural Project via NDV Server", you can see that one action is defined. This action is also named "SAG Install Natural Project via NDV Server".



Policy Information

* Name:	SAG Install Natural Project via NDV Server
Description:	This policy installs a version of a Natural project into one or more Natural environmen The target environments are denoted by the relationship 'To be installed next'.
Version:	
Priority:	11
State:	New 🟥
Organization:	Default Organization
Owner:	one, Natural, Default Organization
System Version:	1

Change State... Run

Actions | Scope | States | Permissions

Edit Actions List... ?

Action	Description	Parameters Set
SAG Install Natural Project via NDV-Serv	This action installs a version of a Natural project into one or more Natural environments. The targe	✓

A green checkmark in the right column indicates that all required input parameters of the action have been set. A red circle in this column would indicate that the action has required input parameters that have not yet been set.

Note: If parameters need to be set, you click the action name. The parameters are then displayed and you can set them.

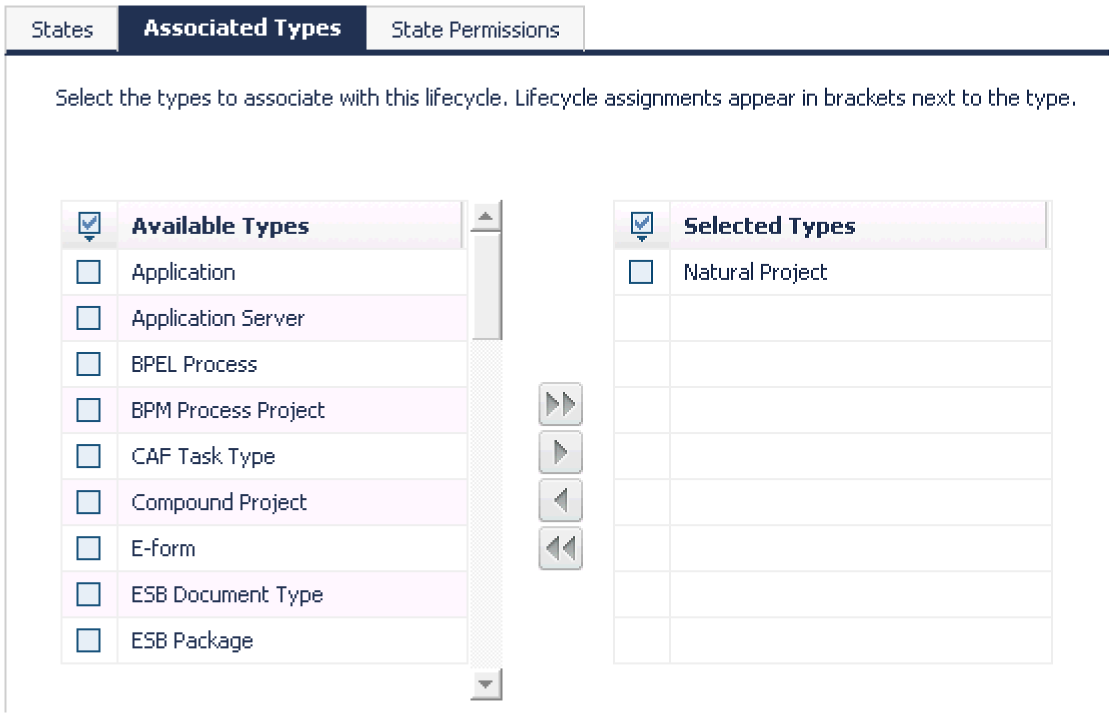
6 Click the **Close** button to return to the **Design/Change-Time Policy Details** page.

7 Click the **Close** button once more to return to the **Edit Lifecycle Model** page.

Note: This sample lifecycle does not use stages, that is, it is not split over two or more registries (for example, a creation registry and a consumption registry). Therefore, the **Stages** button (which is located next to the **Policies** button) is not used. A stage is defined for the end state of a lifecycle model to indicate that the lifecycle continues on another registry.

8 Switch to the **Associated Types** tab.

Here you can see that this lifecycle model has been associated with the asset type "Natural Project". This is the asset type to which this model applies.



Select the types to associate with this lifecycle. Lifecycle assignments appear in brackets next to the type.

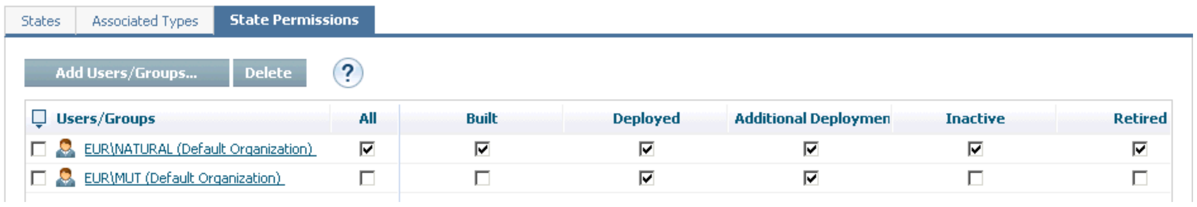
☐	Available Types
☐	Application
☐	Application Server
☐	BPEL Process
☐	BPM Process Project
☐	CAF Task Type
☐	Compound Project
☐	E-form
☐	ESB Document Type
☐	ESB Package

☐	Selected Types
☐	Natural Project

9 Switch to the **State Permissions** tab.

Here you can define the user and group permissions associated with this lifecycle model. Each permission defines whether a user or group can move assets into a particular lifecycle state. If you leave the list of users and groups empty, CentraSite grants state permissions to all users and groups. If the list contains at least one user or group, permissions are denied for all other users and groups who are not in the list.

You use the **Add Users/Groups** button to define the users or groups. Then you select the check boxes for the states into which the user or group is allowed to move the Natural project. Example:



Users/Groups	All	Built	Deployed	Additional Deploymen	Inactive	Retired
EUR\NATURAL (Default Organization)	✓	✓	✓	✓	✓	✓
EUR\MUT (Default Organization)	☐	☐	✓	✓	☐	☐

Note: Keep in mind that changes to the lifecycle model can only be applied as long as the lifecycle model has not yet been activated. When it has already been activated, the **Add Users/Groups** button is disabled.

Activating the Sample Lifecycle Model

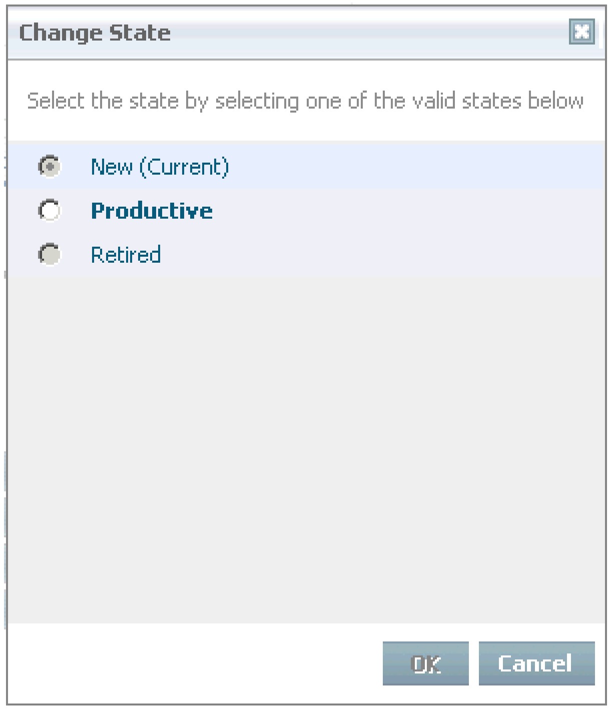
By default, the sample lifecycle models that are delivered with the Lifecycle Manager are inactive (that is, they are in the state "New" and a red icon is shown). To activate a model, you must change its state to "Productive" as described below.

Note: While a lifecycle model is in the "Productive" state, you cannot modify it. If you need to modify a lifecycle model, you have to create a new version of it.

To activate the sample lifecycle model

1 On the **Edit Lifecycle Model** page, click the **Change State** button.

The **Change State** dialog box appears.



2 Select the **Productive** option button.

3 Click the **OK** button.

The lifecycle model is now active (that is, it is in the state "Productive" and a green icon is shown).

Adding an Asset for a Natural Environment

You will now define the target environment in which your Natural project is to be installed. For this purpose, you add an asset of type "Natural Environment".

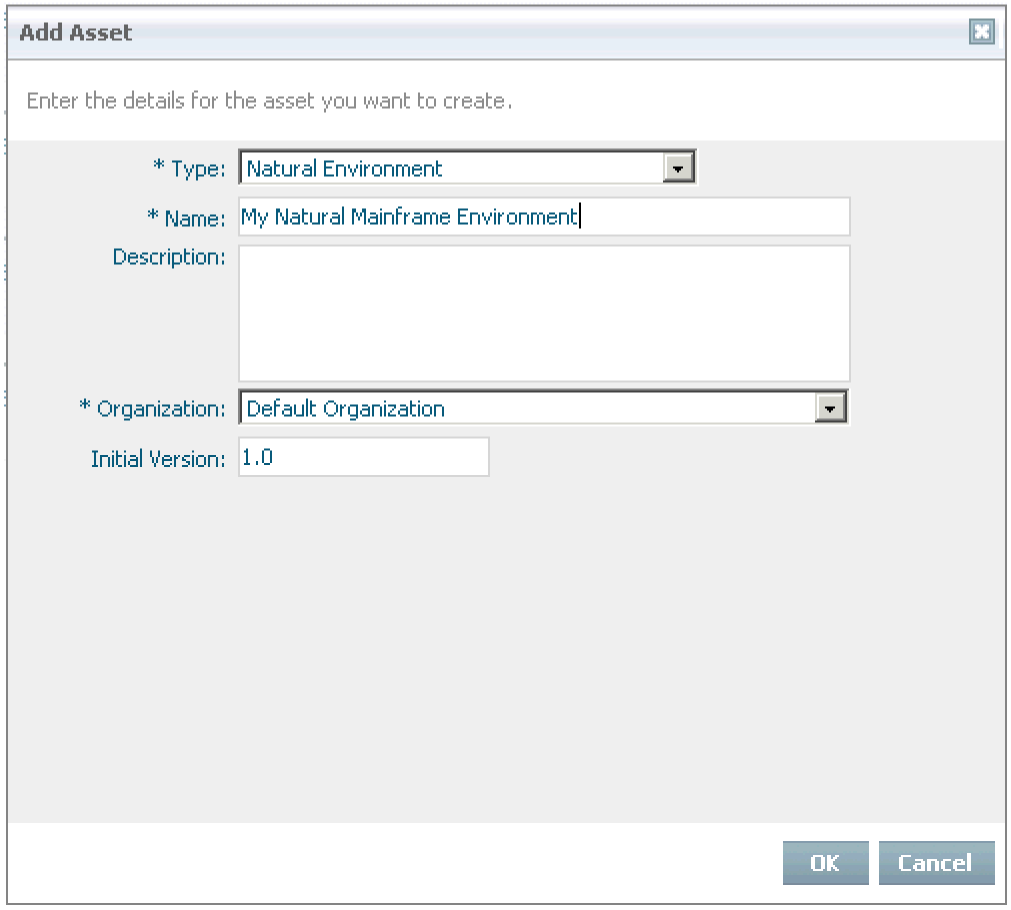
Note: Do not confuse the target environment with the compile environment. The compile environment is defined in the deployment file.

To add an asset for a Natural Environment

1 Go to **Asset Catalog > Browse**.

2 Click the **Add Asset** button.

3 In the resulting dialog box, select the type **Natural Environment** and specify all required information. Make sure to select the appropriate organization. Example:



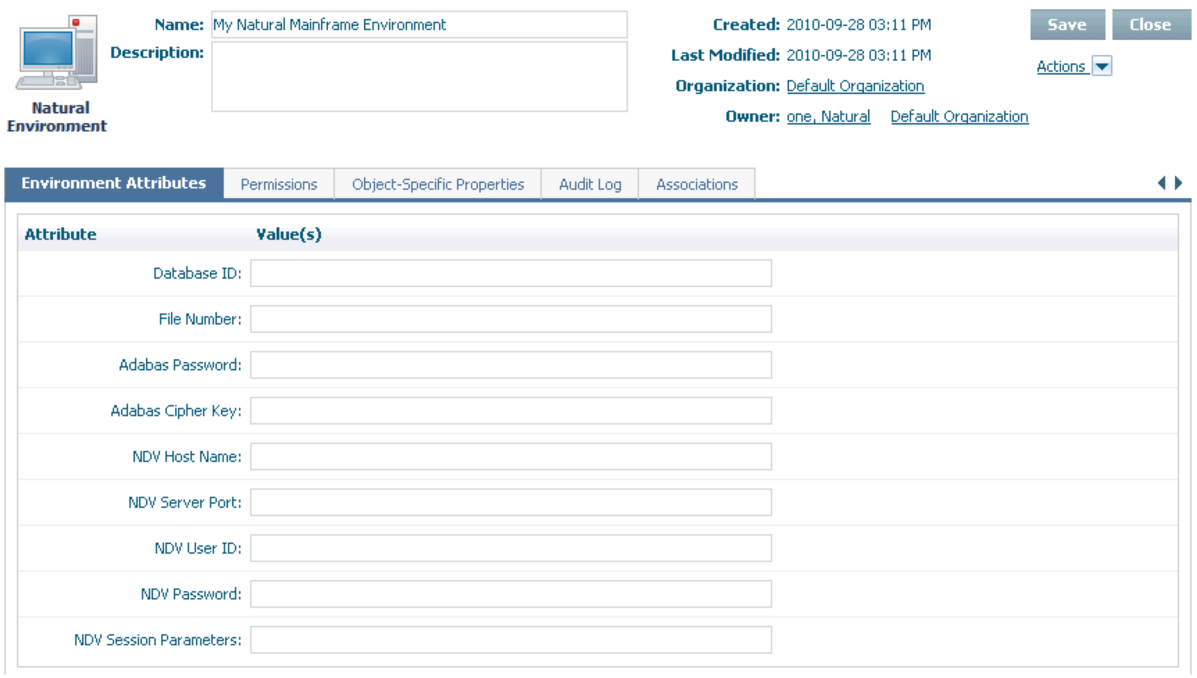
Add Asset

Enter the details for the asset you want to create.

* Type:	Natural Environment
* Name:	My Natural Mainframe Environment
Description:	
* Organization:	Default Organization
Initial Version:	1.0

4 Click the **OK** button.

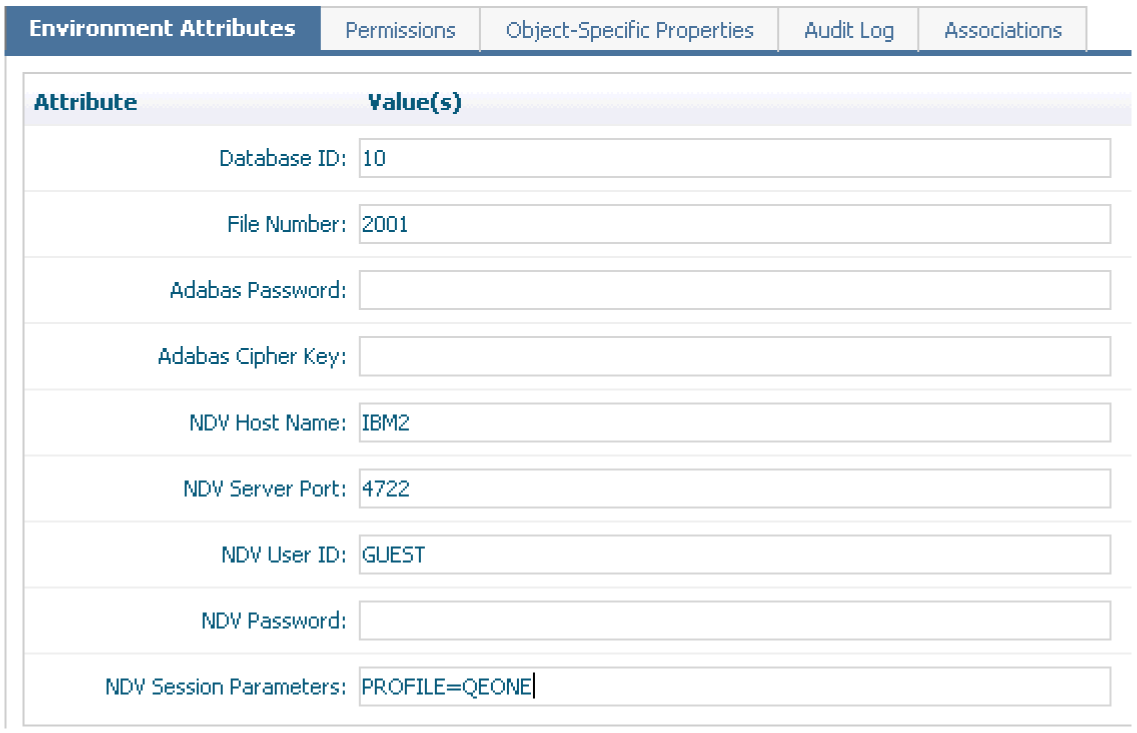
The details page of the newly created asset is now shown.



Name: My Natural Mainframe Environment
Description:
Created: 2010-09-28 03:11 PM
Last Modified: 2010-09-28 03:11 PM
Organization: Default Organization
Owner: one, Natural Default Organization

Attribute	Value(s)
Database ID:	
File Number:	
Adabas Password:	
Adabas Cipher Key:	
NDV Host Name:	
NDV Server Port:	
NDV User ID:	
NDV Password:	
NDV Session Parameters:	

5 On the **Environment Attributes** tab of the new asset, specify all required information for the target environment. Example:



Attribute	Value(s)
Database ID:	10
File Number:	2001
Adabas Password:	
Adabas Cipher Key:	
NDV Host Name:	IBM2
NDV Server Port:	4722
NDV User ID:	GUEST
NDV Password:	
NDV Session Parameters:	PROFILE=QEONE

For detailed information on these attributes, see ***Environment Attributes*** in *Asset Types for Natural Applications*.

6 Click the **Save** button to save all of your changes.

Adding an Asset for a Natural Project

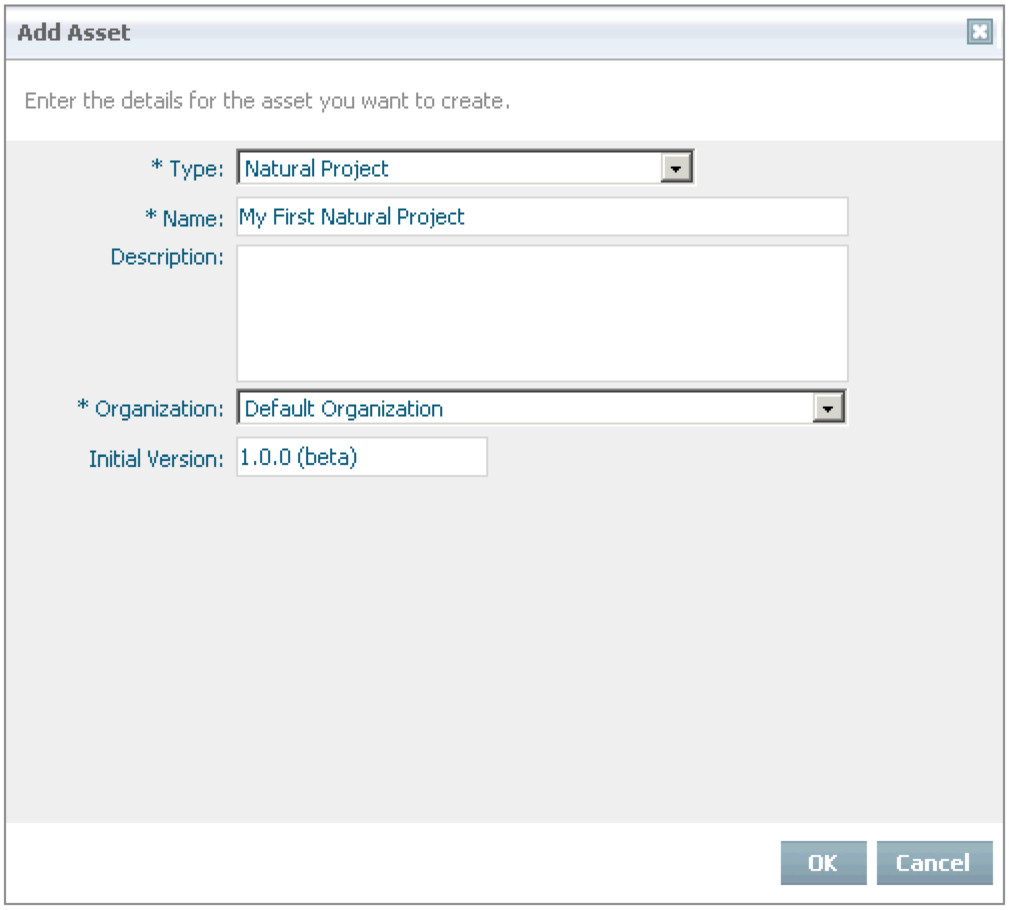
You will now add an asset of type "Natural Project". You will associate the target environment that you have defined with the Natural Environment asset with this Natural Project asset. You will also specify all required build information such as the directory into which the sources are to be checked out and the name of the deployment file.

To add an asset for a Natural Project

1 Go to **Asset Catalog > Browse**.

2 Click the **Add Asset** button.

3 In the resulting dialog box, select the type **Natural Project** and specify all required information. Make sure to select the appropriate organization. Example:



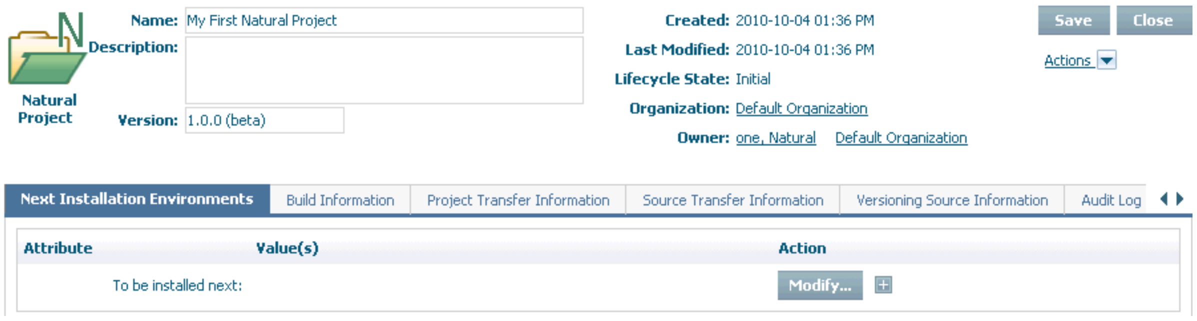
Enter the details for the asset you want to create.

* Type: Natural Project
* Name: My First Natural Project
Description:
* Organization: Default Organization
Initial Version: 1.0.0 (beta)

Note: The initial version that you can specify in this dialog is the user-defined version (as opposed to the internal version number which is automatically generated by CentraSite). You can enter any string in this field, that is, the version identifier does not need to be numeric.

4 Click the **OK** button.

The details page of the newly created asset is now shown.



Name:	My First Natural Project
Description:	
Version:	1.0.0 (beta)
Created:	2010-10-04 01:36 PM
Last Modified:	2010-10-04 01:36 PM
Lifecycle State:	Initial
Organization:	Default Organization
Owner:	one, Natural Default Organization

Natural Project
Save
Close
Actions
Next Installation Environments
Build Information
Project Transfer Information
Source Transfer Information
Versioning Source Information
Audit Log

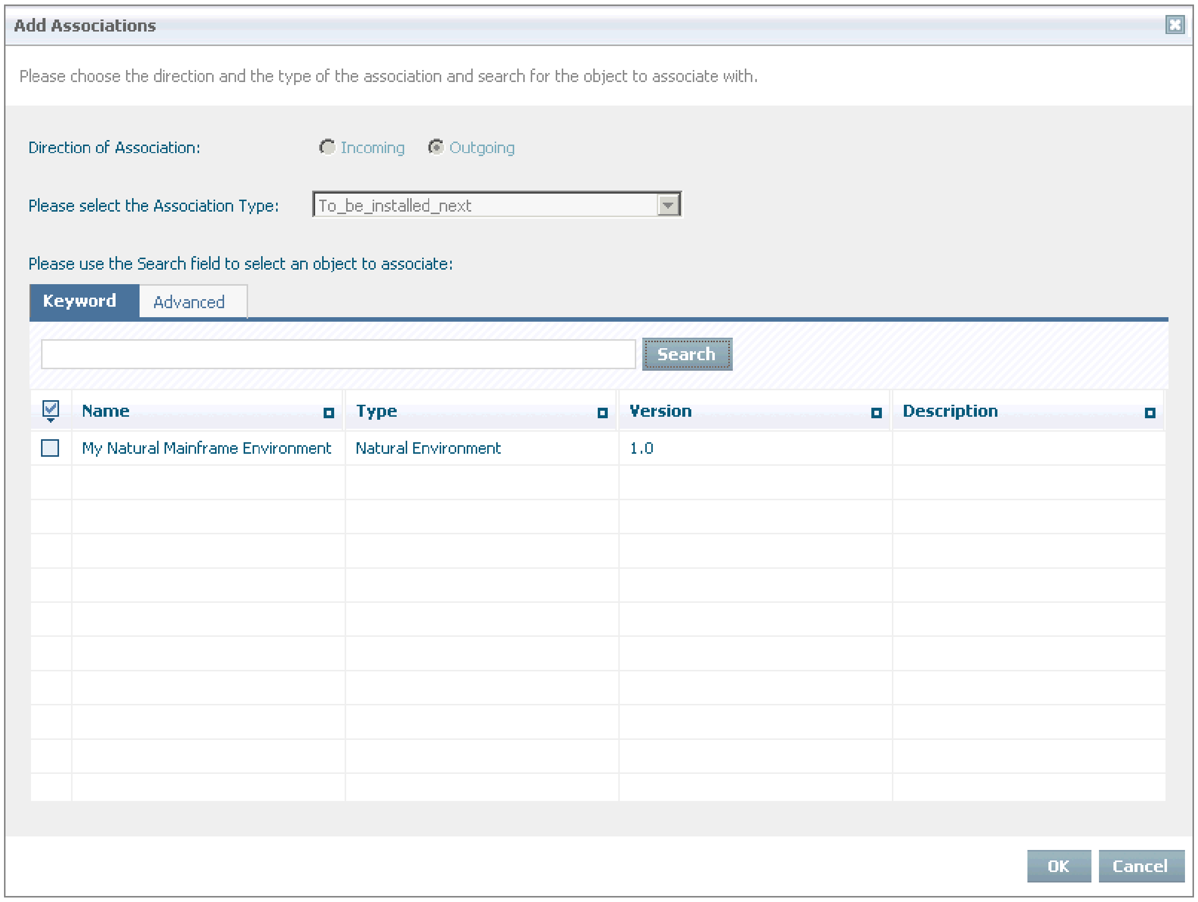
Attribute	Value(s)	Action
To be installed next:		Modify...

5 On the **Next Installation Environments** tab, click the **Modify** button.

The **Add Associations** dialog appears. You will now define your Natural Environment asset as an association.

6 Click the **Search** button (on the **Keyword** tab).

All Natural Environment assets that are currently defined are now shown at the bottom of the dialog.


Add Associations

Please choose the direction and the type of the association and search for the object to associate with.

Direction of Association: Incoming [✓ Outgoing]

Please select the Association Type: To_be_installed_next

Please use the Search field to select an object to associate:

	Name	Type	Version	Description
☐	My Natural Mainframe Environment	Natural Environment	1.0	

7 Enable the check box in front of your Natural Environment asset.

Note: It is possible to install your Natural project into several Natural environments at the same time. If the search function finds more than one Natural Environment asset, you can enable the check boxes for all required environments. However, you must make sure that these target environments run on the same type of operating system. For example, one project can only contain Natural environments on UNIX. If you plan to install into a mainframe environment, you must create a separate Natural project for this purpose.

8 Click the **OK** button.

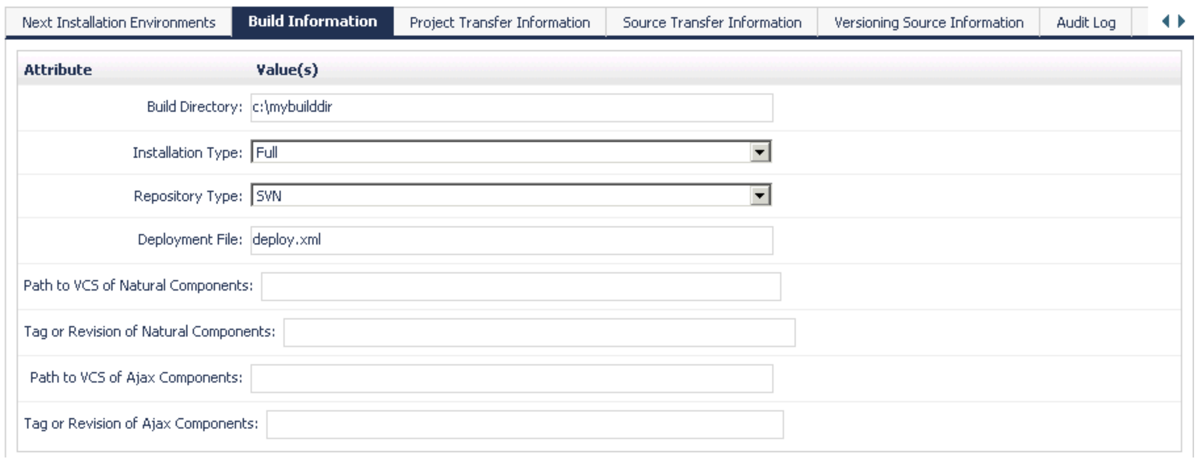
This closes the **Add Associations** dialog.

On the **Next Installation Environments** tab, the name of your Natural Environment asset is now shown next to **To be installed next**.

9 Go to the **Build Information** tab and specify the following information:

Attribute	Description
Build Directory	The directory into which all sources are to be checked out from your versioning repository. This directory also contains the deployment file (see below) and all other files that can be specified on the different tabs of this Natural project. The directory which is defined in the deployment file (that is, the directory which has been specified in the wizard as the root directory) and the build directory of the Lifecycle Manager must be the same. **Important:** Do not delete the build directory and do not change the content that is generated into this directory. If you plan to access the Lifecycle Manager from different locations, make sure that the build directory is located on a shared device.
Repository Type	The versioning repository to be used: CVS or SVN. **Note:** Do not use the **None** option for this tutorial.
Deployment File	The name of the deployment file which has been created with NaturalONE. This file must exist in (or must later be copied to) the above build directory.

You need not fill all fields on this tab. Some information will be filled automatically when you specify a deployment file. Therefore, you only have to specify information in the above mentioned fields. Example:


Next Installation Environments
Build Information
Project Transfer Information
Source Transfer Information
Versioning Source Information
Audit Log

Attribute	Value(s)
Build Directory:	c:\mybuilddir
Installation Type:	Full
Repository Type:	SVN
Deployment File:	deploy.xml
Path to VCS of Natural Components:	
Tag or Revision of Natural Components:	
Path to VCS of Ajax Components:	
Tag or Revision of Ajax Components:	

For detailed information on all attributes that can be specified on this tab, see ***Build Information*** in *Asset Types for Natural Applications*.

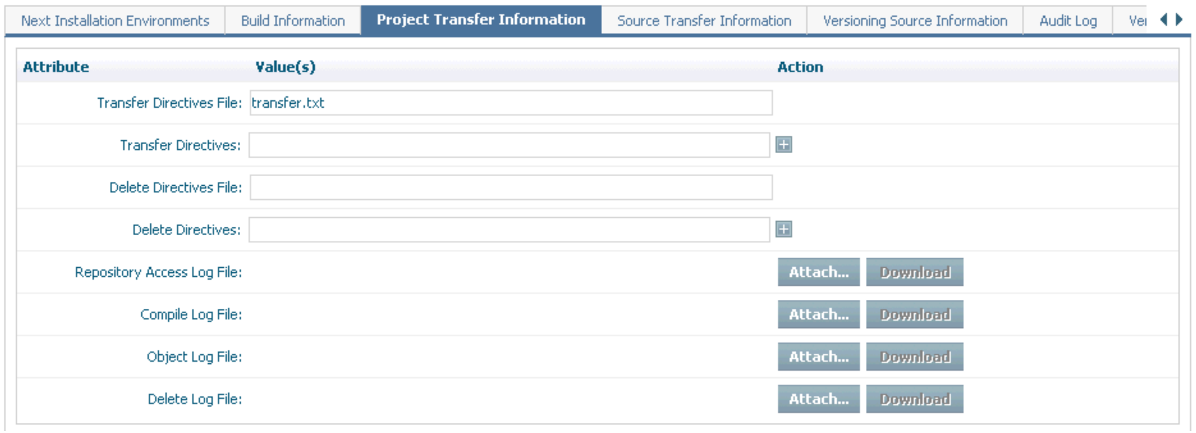
10 Go to the **Project Transfer Information** tab and specify the following information:

Attribute	Description

Transfer Directives File	The name of the file which describes the objects that are to be transferred. This file must exist in (or must later be copied to) the build directory. You need not specify a path to this file.
	
	This information is used by the policy which performs the unload. The syntax is the same as for the `select-clause` and `parameter-clause` of the Natural Object Handler. For detailed information, see *Utilities* in the Natural documentation for the appropriate platform.
	
	Each transfer directive must be specified in a single line. Example:
	
	`* LIB BONNLX OBJT E NATT * FMNUM 1 TONUM 100`
	`LXP06* LIB BONNLX OBJTYPE N SCKIND A NATTYPE PNS DATE TODAY WITH ↩`
	`NEWLIBRARY BERLINLX`
	`LXDDM001* LIB * OBJT D`

Transfer Directives	Instead of using a transfer directives file, you can enter the directives directly on this tab. If both is specified, only the directives on this tab are used. If you need an additional line, click the button showing the plus (+) sign.

Example:


Next Installation Environments
Build Information
Project Transfer Information
Source Transfer Information
Versioning Source Information
Audit Log
Ve
Attribute
Value(s)
Action
Attach...
Download

Transfer Directives File: transfer.txt
Transfer Directives:
Delete Directives File:
Delete Directives:
Repository Access Log File:
Compile Log File:
Object Log File:
Delete Log File:

Note: For this tutorial, it is assumed that you want to install into an empty target environment (that is, into an environment that does not yet contain any version of your Natural project). It is therefore not required to define delete directives. Delete directives are normally used for upgrade installations.

For detailed information on all attributes that can be specified on this tab, see *Project Transfer Information* in *Asset Types for Natural Applications*.

11 Click the **Save** button.

Copying the Deployment File to the Build Directory

Before you continue with the steps below, copy the deployment file (which has been created using NaturalONE) to the build directory that you have just defined for your Natural Project asset.

Browsing the Asset Catalog

You will now browse the asset catalog and select the asset types to be included in the view for which you have already defined assets.

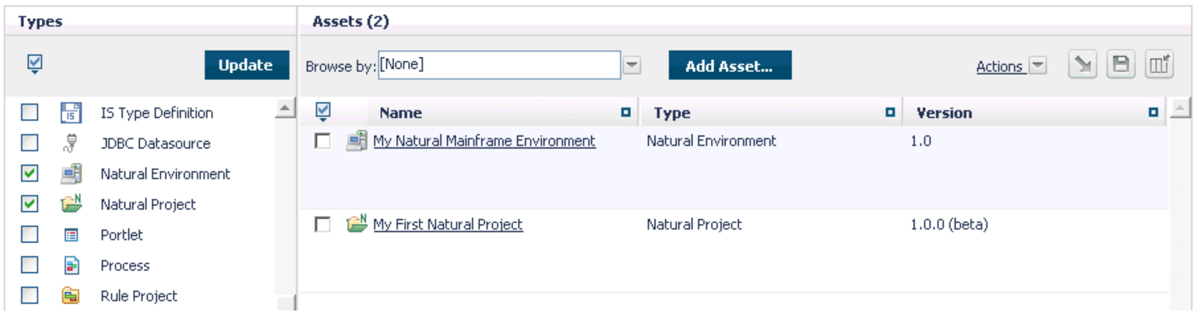
To browse the asset catalog

1 Go to **Asset Catalog > Browse**.

2 On the left side of the asset catalog, enable the check boxes in front of **Natural Environment** and **Natural Project**. Disable the check boxes for any other asset types that are currently selected.

3 Click the **Update** button.

The right side of the page now only shows the assets for the asset types you have enabled. These are the assets that you have just defined for the Natural environment and the Natural project.



Types
☐ IS Type Definition
☐ JDBC Datasource
✓ Natural Environment
✓ Natural Project
☐ Portlet
☐ Process
☐ Rule Project

Name	Type	Version
My Natural Mainframe Environment	Natural Environment	1.0
My First Natural Project	Natural Project	1.0.0 (beta)

Activating the Policies

By default, the sample policies that are delivered with the Lifecycle Manager are inactive. As long as a policy is inactive, it can be modified. To activate a policy, you have to place the policy in "Productive" state. In this state, a policy cannot be modified.

To activate the policies

1 Go to **Policies > Design/Change-Time**.

A list of all available policies is shown. The names of the sample policies delivered with the Lifecycle Manager start with "SAG". For further information, see *Sample Policies for Natural Applications*.

The icon in the **Active** column (at the very right of the page) indicates whether a policy is active (green icon) or inactive (red icon).

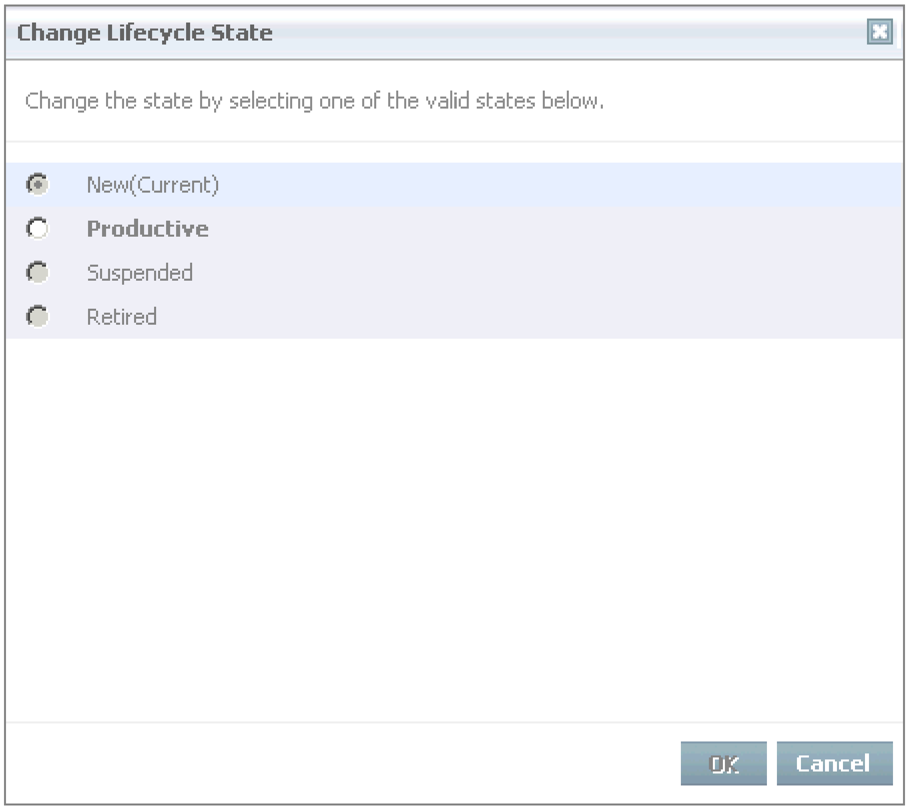
A policy can only be activated when all of its parameters have been set. Therefore, you will first check the parameters.

2 Locate the policy that you want to activate (for example, the "SAG Install Natural Project via NDV Server" policy) and click its name.

3 On the resulting details page, examine the **Actions** tab and verify that all of the actions on this tab display a green checkmark in the **Parameters Set** column. If an action displays a red circle in this column, set its parameters before you continue.

4 When all parameters have been set, click the **Change State** button which is located at the top right of the details page.

The **Change Lifecycle State** dialog box appears.



5 Select the **Productive** option button.

6 Click the **OK** button.

The **State** field on the details page now shows the new state.

7 Repeat the above steps for all policies that are to be activated.

Deploying the Assets According to the Sample Lifecycle Model for Natural Projects

If an asset has an associated lifecycle model, you can switch the asset's lifecycle state. In our case, the sample lifecycle model has been associated with the Natural Project asset type.

You will now change the lifecycle state for the Natural Project asset you have defined. You will first change the state from "Initial" to "Built", and then from "Built" to "Deployed".

Notes:

1. Users who are not administrators can also change the lifecycle state of an asset if the corresponding permission has been granted.

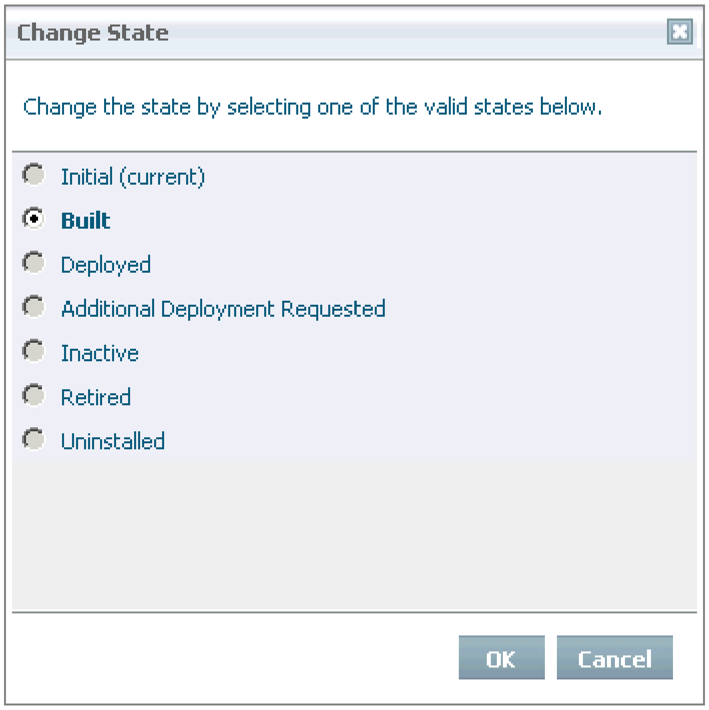
2. It is also possible to change the lifecycle state of an asset using the CentraSite plug-in for Eclipse. This tutorial, however, only explains how to do this using CentraSite Control.

To change the lifecycle state of an asset

1 Go to **Asset Catalog > Browse**.

2 On the right side of the page, invoke the context menu for your Natural Project asset and click **Change Lifecycle State**.

The **Change State** dialog box appears.



Change State

Change the state by selecting one of the valid states below.

Initial (current) [✓ Built] Deployed Additional Deployment Requested Inactive Retired Uninstalled

The next possible state is automatically selected. In this case, this is the "Built" state.

3 Click the **OK** button.

The policies defined for the state transition from "Initial" to "Built" are now executed. These policies perform the following steps:

- Check out the Natural project from the versioning repository into the build directory.
- Upload the Natural sources to the Natural development server. This is the compile environment. The setting in the deployment file controls whether all sources are to be uploaded, or only the changed sources.
- Catalog (build) the Natural objects on the Natural development server.
- Unload the cataloged Natural objects into a transfer file according to the transfer directives defined for the Natural project.

- Create a *Version_<n>* directory in the build directory (where <*n*> represents the current internal version number of your Natural project).
- Put the transfer file into the *Version_<n>* directory. This is the file with the extension *.sag*.
- Create several protocol files (with the extension `.txt`) in the *Version_<n>* directory. The names of these files start with "unload". They contain information from the Natural Object Handler.

This may take a while. Wait until the hour glass is no longer shown.

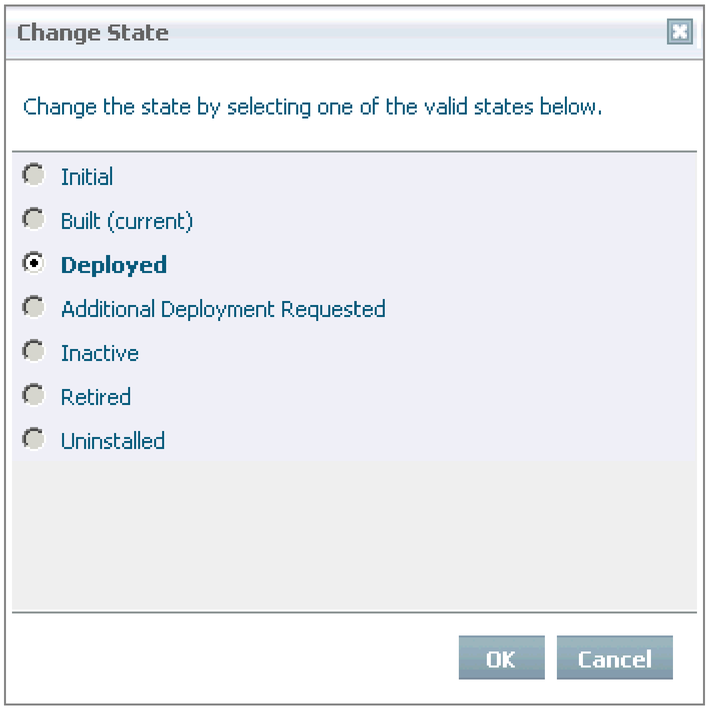
4 Click the name of your Natural Project asset.

The details page for this asset is now shown. You can see that the new state "Built" is shown next to **Lifecycle State**.

Next, you will change the lifecycle state of the asset from "Built" to "Deployed". This time, you will change the state directly on the page which shows the details for the asset.

5 Click the **Actions** button and then click **Change Lifecycle State**.

The **Change State** dialog box appears.


Change State

Change the state by selecting one of the valid states below.

- ○ Initial
- ○ Built (current)
- ● **Deployed**
- ○ Additional Deployment Requested
- ○ Inactive
- ○ Retired
- ○ Uninstalled

This time, the next possible state "Deployed" is automatically selected.

6 Click the **OK** button.

The policy defined for the state transition from "Built" to "Deployed" is now executed. It performs the following steps for each target environment that is defined on the **Next Installation Environments** tab:

- If objects are to be deleted in the target environment: delete the objects according to the delete directives defined for the Natural project.
- Load the transfer file into the target environment.
- Create several protocol files (with the extension `.txt`) in the *Version_<n>* directory. The names of these files start with "load". They contain information from the Natural Object Handler.

This may take a while. Wait until the hour glass is no longer shown.

When processing has completed, you can see that the new state "Deployed" is shown next to **Lifecycle State**. This indicates that the project has been installed.

Note: It may happen that "PENDING (*state-name*)" is shown next to **Lifecycle State**. This means that a condition has been encountered which caused the asset to become stuck in the pending state. To resolve the situation where an asset becomes stuck in the pending state, a user that belongs to the CentraSite Administrator role can use the **Revert Pending State** command from the **Actions** menu to return the asset to its prior state. After the asset is reverted and the issue that caused the asset to become stuck is corrected, an authorized user can switch the asset to its next lifecycle state again.

After a Natural project has been installed, the **To be installed next** entry on the **Next Installation Environments** tab is empty. The **Associations** tab, however, now contains an entry with the association type **Installed_in** which indicates in which environment the project has been installed.

Important: Do not change the associations on the **Associations** tab because this will make the Lifecycle Manager inconsistent.

If you decide later that your Natural project needs to be deployed to one or more additional environments, you specify them on the **Next Installation Environments** tab. Two changes in the lifecycle state are then required. First, you change the lifecycle state to "Additional Deployment Requested", and then you change it to "Deployed".

7 Go to the **Project Transfer Information** tab of the Natural project.

Links to the different types of log files are now provided. These files contain information on the objects that have been processed, including the corresponding dates and times. You will find information such as the following:

- A list of the objects that have been compiled.
- A list of the generated programs which have been installed in the target environments.
- A list of the files which have been deleted in the target environments.

Creating a New Version of a Natural Project

When you create a new version of an asset, the new version enters the initial lifecycle state and will then pass through the entire lifecycle.

We will now assume that you have changed some sources in your Natural project, committed them your source control system and created a new deployment file. We will also assume that you now want to deploy these changes according to the sample lifecycle model. Therefore, you will now create a new version of your Natural Project asset.

The new version will be deployed to the target environments which are defined for this version. For example, your first version was deployed to only one environment, but now you may decide that you want to deploy the new version into additional environments.

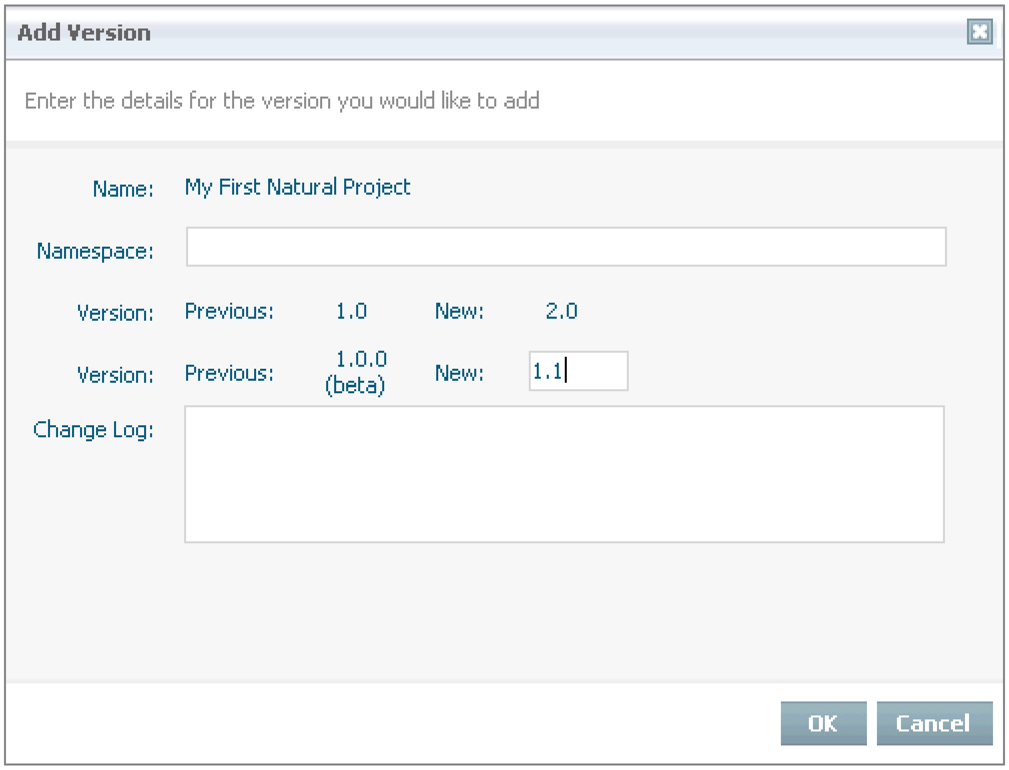
Before you proceed with the steps below, make sure to copy the new deployment file, which contains the latest changes, to your build directory.

If the list of files that is to be transferred has changed and if objects need to be deleted in the target environment, make sure to copy also the transfer directive file and the delete directives file to your build directory (or make sure to specify the directives directly on the appropriate tab of your Natural Project asset).

To create a new version of a project

1. Go to **Asset Catalog > Browse**.
2. Invoke the context menu for the Natural Project asset you have previously defined and click **Add New Version**.

   The **Add Version** dialog box appears.
3. In the **New** text box, specify an identifier for the new version, for example, "1.1". You can use any versioning scheme. The version identifier does not need to be numeric. Example:


Add Version

Enter the details for the version you would like to add

Name:	My First Natural Project			
Namespace:				
Version:	Previous:	1.0	New:	2.0
Version:	Previous:	1.0.0 (beta)	New:	1.1
Change Log:				

The version numbers 1.0 and 2.0 which are shown in the first **Version** row of the dialog box are the internal system version numbers which are automatically generated by CentraSite.

Optional. Specify a comment or other descriptive information about the new version in the **Change Log** text box.

Note: Namespaces are not supported by the Lifecycle Manager. Do not enter any information in the corresponding text box.

4 Click the **OK** button.

The details page of the new version of the Natural Project asset is now shown. It shows the new version number that you have specified. The information from the previous version (build information and project transfer information) is also available in the new version. The installation environment, however, is not copied. You must always specify it for each new Natural project version that you create.

5 On the **Next Installation Environments** tab, click the **Modify** button.

In the resulting **Add Associations** dialog, define the Natural Environment assets which pertain to the target environments in which you want to install the new version (how to do this has already been described in the section *Adding an Asset for a Natural Project*).

6 Go to the **Build Information** tab.

7 Since you want to update a previous version of your project, make sure to use the same build directory as for the previous version.

8 Make sure to specify the name of the deployment file which contains the latest changes.

9 From the **Installation Type** drop-down list box, select either **Incremental** or **Hot Fix**, depending on the changes you have made in your project.

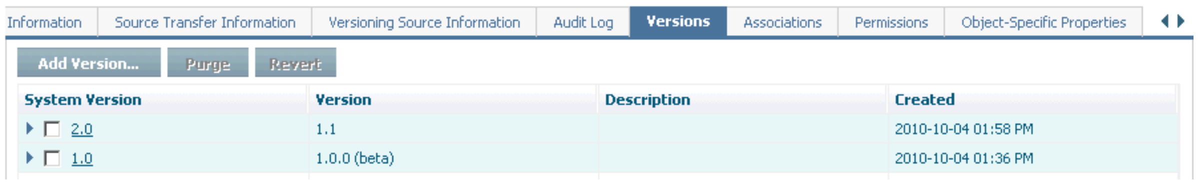
10 Go to the **Project Transfer Information** tab.

11 If you have created new files containing transfer and delete directives, specify their names. Or enter the directives directly on this tab.

12 Click the **Save** button.

13 Go to the **Versions** tab and view its content.

All versions of the current Natural project, including the version you have just created, are listed on this tab.


Information
Source Transfer Information
Versioning Source Information
Audit Log
Versions
Associations
Permissions
Object-Specific Properties
Add Version...
Purge
Revert

System Version	Version	Description	Created
2.0	1.1		2010-10-04 01:58 PM
1.0	1.0.0 (beta)		2010-10-04 01:36 PM

Note: It is also possible to add new versions from this tab.

Downgrading to a Previous Version of a Natural Project

If you want to downgrade to a previous version of your Natural project, you display the details page for the previous version and specify the target environment for the downgrade installation. For example, if version 2.0 is currently installed in the target environment and you want to downgrade to version 1.0, you have to display version 1.0. See also *Version Handling*.

We will now assume that version 2.0 is installed in the target environment and you want to return to version 1.0 (these are the internal version numbers which are independent from the version numbers you specify).

The information in your build directory will be used to install the previous version. In the case of a downgrade, you need not provide any deployment file, transfer directives file or delete directives file. However, the downgrade will only work when the structure of your build directory is still intact. Therefore, it is important that you do not delete the build directory or change its generated contents.

To downgrade to a previous version

1. Go to the **Versions** tab of your Natural project.
2. Click the version of the Natural project to which you want to downgrade.

   The details page for this version is shown. Since this version has already been installed, the state "Deployed" should be shown next to **Lifecycle State** and the **To be installed next** entry on the **Next Installation Environments** tab should be empty.
3. On the **Next Installation Environments** tab, click the **Modify** button.

   In the resulting **Add Associations** dialog, define the Natural Environment asset which pertains to the target environment in which you want to perform the downgrade (how to do this has already been described in the section *Adding an Asset for a Natural Project*).

   When the target environment has been defined, two changes (as described below) are required in the lifecycle state.
4. Click the **Actions** button and then click **Change Lifecycle State**.
5. In the resulting **Change State** dialog box, make sure that the lifecycle state "Additional Deployment Requested" is selected and click the **OK** button.
6. Invoke the **Change State** dialog box once more, make sure that the lifecycle state "Deployed" is selected and click the **OK** button.

   The version that is currently installed in the target environment is now downgraded.

   Note: Revision processing is not supported by the Lifecycle Manager. By default, revision processing is switched off in CentraSite. Therefore, the **Revert** button which is shown on the **Versions** tab is disabled.

Scheduling the Installation of a Natural Project

In the section *Deploying the Assets According to the Sample Lifecycle Model for Natural Projects* you have learned how to install a Natural project in a defined target environment by changing the lifecycle state manually. In addition to this manual state change, you can also set up a schedule for a Natural project so that the project is automatically installed at a predefined date and time. This also enables you to automatically install the project into different target environments at different times. For example, you can define that the project is to be installed tonight into environment A and that it is to be installed two weeks later into environment B.

The prerequisite for an automatic installation is that the Natural Project asset is in a lifecycle state which has the "Deployed" state as a target state (in other words, a transfer file must already exist in your build directory). With the sample lifecycle model named "SAG Demo Model for Natural

Projects", the project must therefore be either in the "Built" state or in the "Additional Deployment Requested" state.

When the scheduled date and time is reached, the transition to the "Deployed" state is initialized and all policies which are defined for this state transition are executed.

After an automatic installation has successfully completed, the project is in the "Deployed" state. If you have scheduled additional automatic installations into other target environments which are to occur in the future, you have to make sure that the project is moved to the "Additional Deployment Requested" state. The policy "SAG Propagate Natural Project to Next State", which is defined for the "Deployed" state of the sample lifecycle model, takes care of this.

If the Natural Project asset is in a lifecycle state which does not have the "Deployed" state as a target state (for example, if it is still in the "Initial" state), an automatic installation cannot be performed. When the scheduled date and time is reached in this case, nothing is installed and a corresponding message is written to a log file. The location of this log file is defined in the properties file *LCM-Natural.prop*.

We will now assume that the current version of your project has not yet been installed, that it is in the "Built" state, and that the **To be installed next** entry on the **Next Installation Environments** tab defines the target environment(s) in which you want to install your project (how to do this has already been described in the section ***Adding an Asset for a Natural Project***).

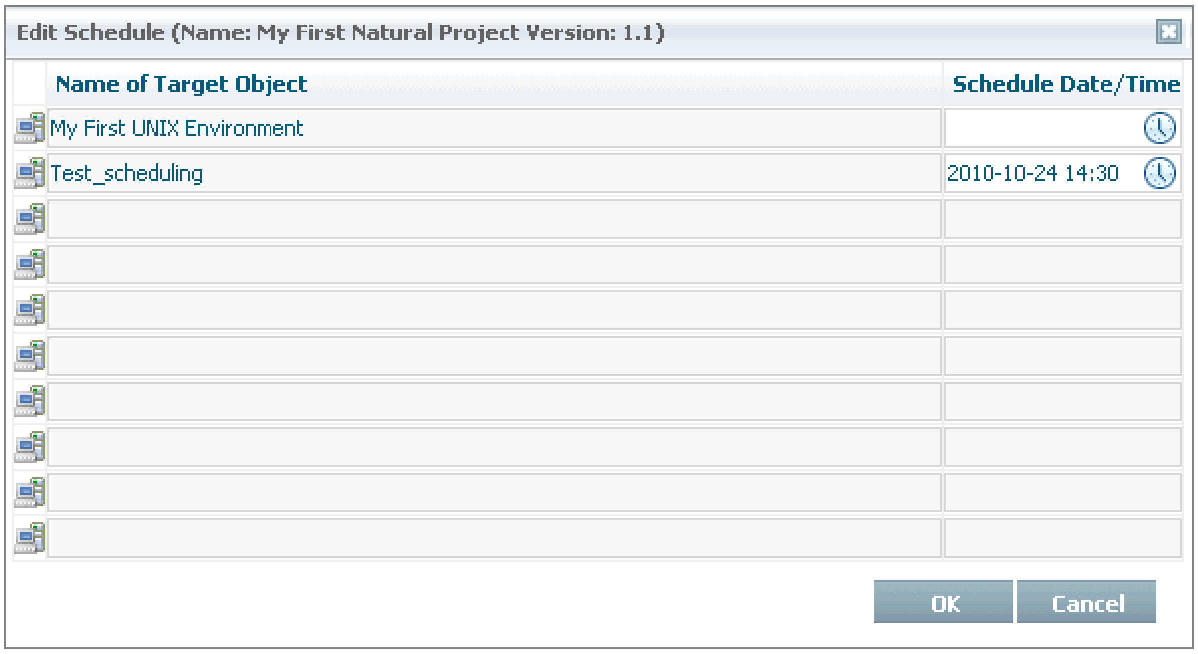
To schedule a Natural project

1 When the project details are currently shown, click the **Actions** button and then click **Edit Schedule** (it does not matter which tab of your Natural project is currently shown).

  Or:

  When the Natural project is currently shown in a list (for example, on the **Asset Catalog > Browse** page), invoke the context menu for the Natural project and then click **Edit Schedule**.

  The **Edit Schedule** dialog box appears, listing all environments that are currently defined on the **Next Installation Environments** tab. When an installation into a target environment has already been scheduled, the corresponding date and time are shown in the dialog box. Example:



Name of Target Object	Schedule Date/Time
My First UNIX Environment	
Test_scheduling	2010-10-24 14:30

Notes:

1. After a scheduled installation in a target environment has completed, the corresponding entry is no longer shown in the dialog box (and the environment is no longer shown as “To be installed next” on the **Next Installation Environments** tab).
2. If you manually install into a target environment by changing the lifecycle state before the scheduled date is reached, the corresponding entry is no longer shown in the dialog box.

2 To schedule a date and time for a target environment, click the button which is shown next to the corresponding environment.

Or:

Double-click the **Schedule Date/Time** text box for the corresponding target environment.

A **Date Input** dialog box appears.

3 Specify the date (that is, select month, day and year) on which the installation is to be started.

4 Specify the time (that is, enter the hour and minute in the corresponding text boxes) at which the installation is to be started.

Important: You have to specify the time using the 24-hour format (for example, "14:30" instead of "2:30 PM").

5 Click the **OK** button to close the **Date Input** dialog box.

The specified date and time is now shown in the **Edit Schedule** dialog box.

Notes:

1. If you want to modify a scheduled date and time, you can simply change the numbers in the **Schedule Date/Time** text box.
2. If you want to delete a scheduled date and time, you simply delete the contents of the **Schedule Date/Time** text box.

6 Click the **OK** button to save your schedule.

Your schedule is now passed to the scheduling tool in your operation system. This is `schtasks` for Windows and `at` for Linux. Any existing Lifecycle Manager entries in the scheduling tool are automatically adjusted or deleted according to your specifications in the **Edit Schedule** dialog box. When a scheduled date and time is reached, the scheduling tool starts the installation in the corresponding target environment.

Simulating the Installation of a Natural Project

If you want to find out which Natural modules will be installed when a Natural project is installed into a specific environment, you use the policy "SAG Simulate Installation to Specific Environment" as described below. This policy creates a report in the build directory which lists all Natural modules which a regular installation would load into the target environment. For example:

```
Software AG - NaturalONE - Lifecycle Manager
Load simulated 2012-01-17 10:58:06
Natural Environment: su_10_32 / uddi:534964cd-0d17-11e0-8566-bf9ea9ec04ef
Version to be installed: su_unload_2 v2 / uddi:244964ba-0d17-11e0-8566-bf8ea2ec04ef HOTFIX
Installed in Version:    su_unload_3 v3 / uddi:7ff854b0-132a-11e0-9f95-fae4daa4d999

Items to be loaded:
Library Object Name Type    S/C Version
------- ----------- ------- --- ---------------------------------------------------------
BONNLX  LXP003      Program S/C su_unload_2 v2 / uddi:244964ba-0d17-11e0-8566-bf8ea2ec04ef


Items to be deleted:
Library Object Name Type         S/C
------- ----------- ------------ ---
BONNLX  LXN001      Subprogram   S/C
BONNLX  LXN002      Subprogram   S/C
BONNLX  LXN003      Subprogram   S/C
BONNLX  LXP001      Program      S/C
BONNLX  LXP002      Program      S/C
BONNLX  LXP004      Program      S/C
SYSTEM  1-3008      System Error S/L
SYSTEM  1-3009      System Error S/L
```

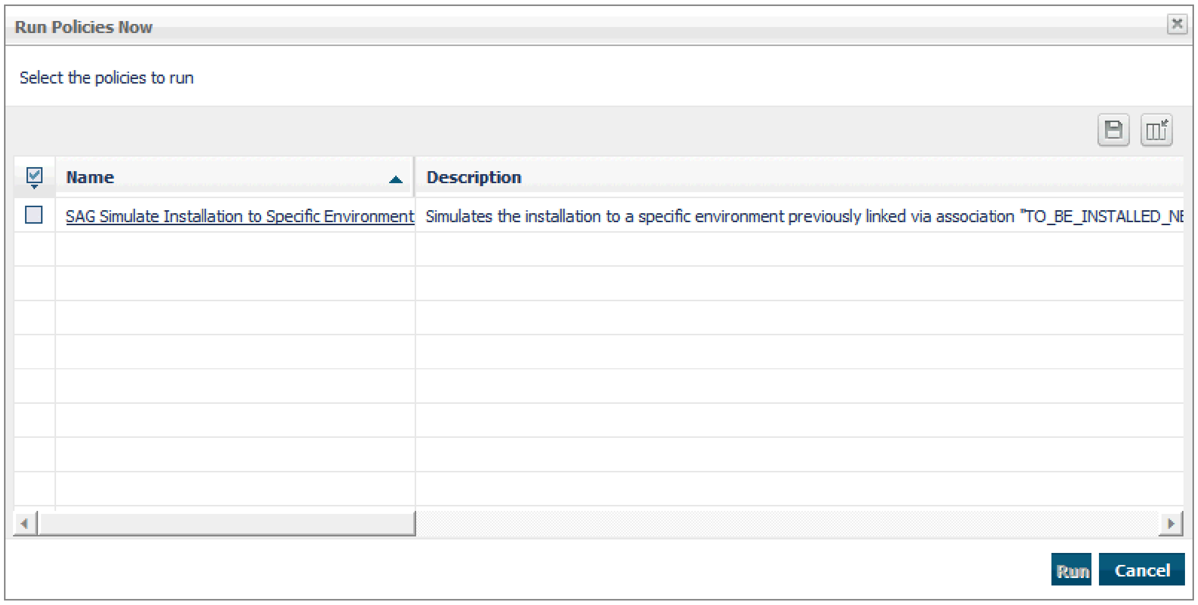
Note: The policy "SAG Simulate Installation to Specific Environment" simulates the installation steps that are normally performed by the policy "SAG Install Natural Project via NDV

Server". It is intended to be used when a connection to an Natural Development Server (NDV) is available.

To simulate the installation of a Natural project

1. Make sure that the policy "SAG Simulate Installation to Specific Environment" has been placed in the "Productive" state (see also *Activating the Policies*).
2. Go to the **Next Installation Environments** tab which pertains to the version of the Natural project for which you want to simulate an installation.
3. Click the **Modify** button.
4. In the resulting **Add Associations** dialog, click the **Search** button and select the assets which pertain to the target environments for which you want to simulate the installation (how to do this has already been described in the section *Adding an Asset for a Natural Project*).
5. Click the **Actions** button and then click **Run Policy**.

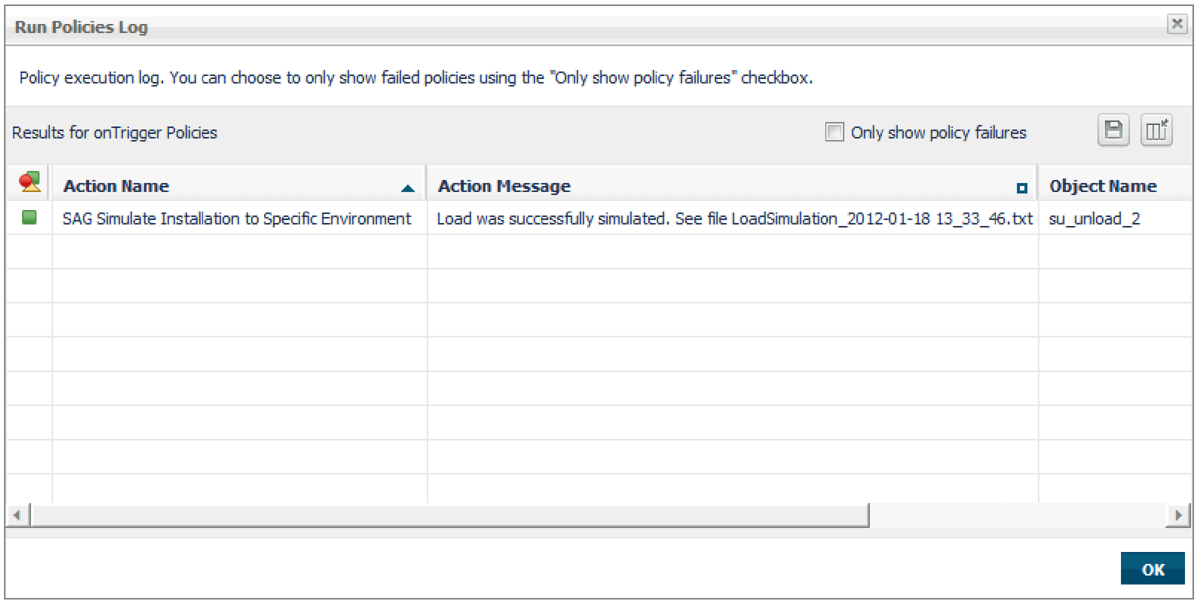
   The **Run Policies Now** dialog box appears.



	Name	Description
☐	SAG Simulate Installation to Specific Environment	Simulates the installation to a specific environment previously linked via association "TO_BE_INSTALLED_NE

6. Enable the check box in front of the "SAG Simulate Installation to Specific Environment" policy.
7. Click the **Run** button.

   Wait until information such as the following is shown:


Run Policies Log

Policy execution log. You can choose to only show failed policies using the "Only show policy failures" checkbox.

Results for onTrigger Policies

☐ Only show policy failures

	Action Name	Action Message	Object Name
🟩	SAG Simulate Installation to Specific Environment	Load was successfully simulated. See file LoadSimulation_2012-01-18 13_33_46.txt	su_unload_2

The report is written to the build directory of the Natural project. It is written to a text file with the name *LoadSimulation_<time-stamp>.txt* which is located in the root of the build directory.

Uninstalling a Natural Project

If you want to uninstall a Natural project from a specific environment, you use the policy "SAG Uninstall Natural Project via NDV Server" as described below.

Notes:

1. The sample lifecycle model for projects that is delivered with the Lifecycle Manager includes the lifecycle state "Uninstalled". Keep in mind that it is only possible to set a project to this state when the Natural project has been deleted from all environments and when the Natural project is not scheduled to be installed.
2. If you want to find out in which environments the Natural project is currently installed, go to the **Associations** tab of your Natural project and check the entries for the association type "Installed_in".

To uninstall a Natural project from one or more environments

1 Make sure that the policy "SAG Uninstall Natural Project via NDV Server" has been placed in the "Productive" state (see also *Activating the Policies*).

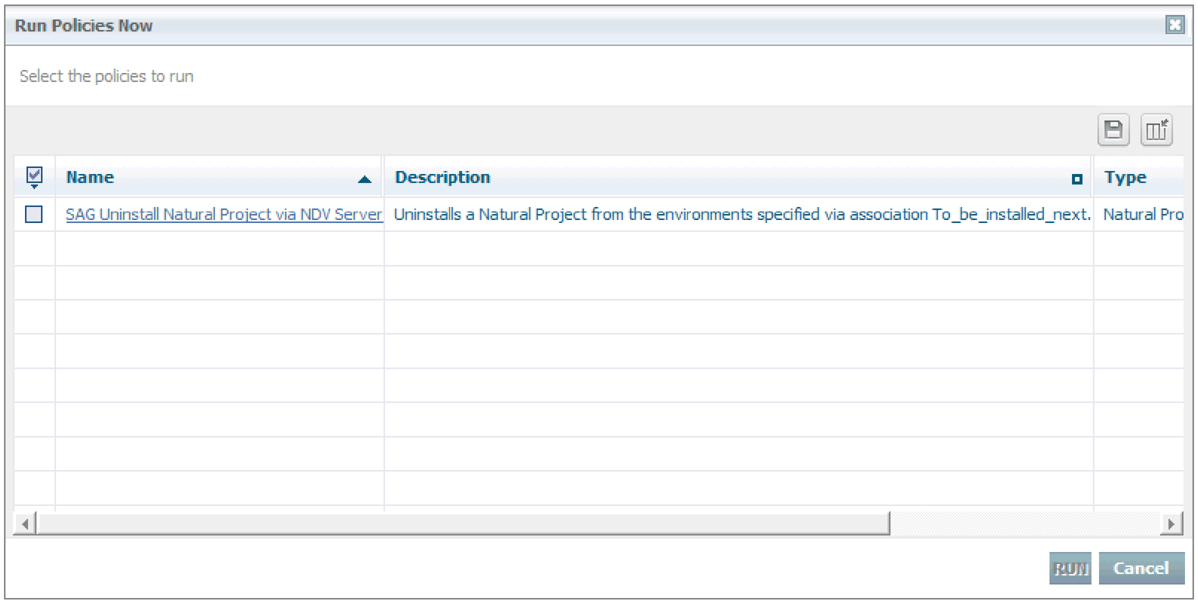
2 Go to the **Next Installation Environments** tab which pertains to the version of the Natural project that you want to uninstall.

3 Click the **Modify** button.

4 In the resulting **Add Associations** dialog, click the **Search** button and select the assets which pertain to the target environments in which you want to uninstall the project (how to do this has already been described in the section *Adding an Asset for a Natural Project*).

5 Click the **Actions** button and then click **Run Policy**.

The **Run Policies Now** dialog box appears.



	Name	Description	Type
☐	SAG Uninstall Natural Project via NDV Server	Uninstalls a Natural Project from the environments specified via association To_be_installed_next.	Natural Pro

6 Enable the check box in front of the "SAG Uninstall Natural Project via NDV Server" policy.

7 Click the **Run** button.

The Natural project is now uninstalled in the specified target environments.

6 Natural for Ajax Applications

- General Information 62
- Using Two Natural Projects 63
- Using a Single Natural Project 63
- Special Deployment Options 63
- Adding an Asset for an Application Server 64
- Adding an Association for an Application Server to a Natural Project 67
- Actions for the User Interface Components 68

General Information

A Natural for Ajax application is a rich internet application which uses the Ajax (Asynchronous JavaScript and XML) technology. When you want to deploy a Natural for Ajax application with the Lifecycle Manager, you use a Natural for Ajax application that has been created using NaturalONE's Ajax Developer.

A Natural for Ajax application consists of two parts that are usually installed on two different machines:

- The Natural modules (adapters, programs, subprograms and other Natural objects) are installed on a Natural server. For this purpose, you use a Natural deployment file which has been created using NaturalONE's deployment wizard for Natural applications. For detailed information, see *Deploying Natural Applications* in *Using NaturalONE*.
- The user interface components (page layouts of rich GUI pages and other user interface related files) are installed in a Natural for Ajax environment on an application server or servlet container. For this purpose, you use a WAR deployment file which has been created using NaturalONE's deployment wizard for web applications. For detailed information, see *Deploying the Application* in *Natural for Ajax*.

Using the Lifecycle Manager, you can deploy a Natural for Ajax application in two different ways:

- You create two assets of type "Natural Project", one to administer the Natural modules on the Natural server and another to administer the user interface components on the application server. In this case, you are able to define two different lifecycle models. If you need to synchronize the deployment of the two projects, you have to create a **compound project**.
- You create a single asset of type "Natural Project" to administer both, the Natural modules on the Natural server and the user interface components on the application server. In this case, you are able to install the changes on both types of server in synchronism. As a prerequisite for this type of deployment, you need a master deployment file which includes exactly one Natural deployment file and one WAR deployment file. For detailed information, see *Using a Master Deployment* in *Using NaturalONE*.

In addition to the Natural Project asset(s), you also have to add an asset of type “Application Server”. This asset type, for which an association has to be defined in the appropriate Natural Project asset, defines the environment in which the user interface components are to be installed.

Notes:

1. The machine which runs the application server onto which you want to deploy your Natural for Ajax applications must be an SFTP server. See also ***Software Requirements***.
2. You need a public/private key pair for authentication. See also ***How to Proceed after the Installation?***.

Using Two Natural Projects

You create the Natural project which is used to administer the Natural modules in the same way as described in the section ***First Steps***. You add the association for the Natural Environment asset on the **Next Installation Environments** tab of the Natural project.

When you create the Natural project which is used to administer the user interface components, you create it in the same way as described for the above project. In this case, however, you add an association for an Application Server asset on the **Next Installation Environments** tab of the Natural project (instead of an association for a Natural Environment asset).

Since two different projects are used, you should use two different build directories.

You have to copy the Natural deployment file to build directory which is defined in the Natural project that is used to administer the Natural modules.

You have to copy the WAR deployment file to the build directory which is defined in the Natural project that is used to administer the user interface components.

Using a Single Natural Project

You create the Natural project in the same way as described in the section ***First Steps***. You add both associations, one for the Natural Environment asset and another for the Application Server asset, on the **Next Installation Environments** tab of the Natural project.

Keep in mind that you have to use a master deployment file in this case. This means that you have to specify the name of the master deployment file on the **Build Information** tab of the Natural project.

You have to copy all three deployment files (master, Natural and WAR) to the build directory which is defined in the Natural project.

Special Deployment Options

Additional steps may be required, depending on the settings that have been defined in the NaturalONE deployment wizards. These are described in the topics below:

- Deployment Mode

- Zipped Deployment Package

Deployment Mode

Depending on the development mode that has been defined for the WAR deployment file, the deployment wizard creates either a single *.war* archive or exports the selected components to a folder.

When a *.war* archive has been created, the WAR template file *njx<nn>.war* (where *<nn>* stands for the version number) must be accessible. The `natural.ant.ajax.njx.application` property in the WAR deployment file defines the location of this template file. If the template file is not accessible, you have to copy it to the build directory, and, if required, you have to adapt the `natural.ant.ajax.njx.application` property so that it defines this location.

Zipped Deployment Package

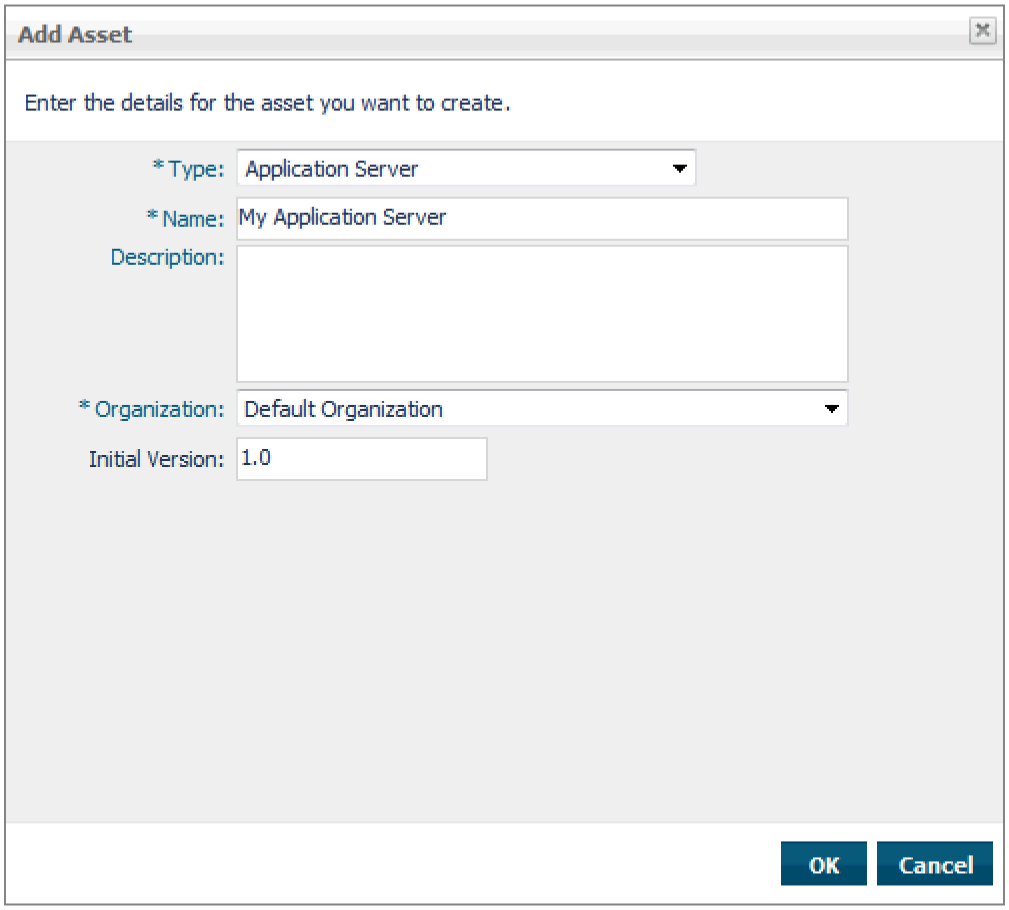
When creating the master deployment file, you can define to create a zip archive instead of single files. When a zip archive has been created, you have to unzip it before you copy it to the build directory.

Adding an Asset for an Application Server

The target environment in which the user interface components are to be installed is defined by an asset of type "Application Server".

To add an asset for an application server

1. Go to **Asset Catalog > Browse**.
2. Click the **Add Asset** button.
3. In the resulting dialog box, select the type **Application Server** and specify all required information. Make sure to select the appropriate organization. Example:

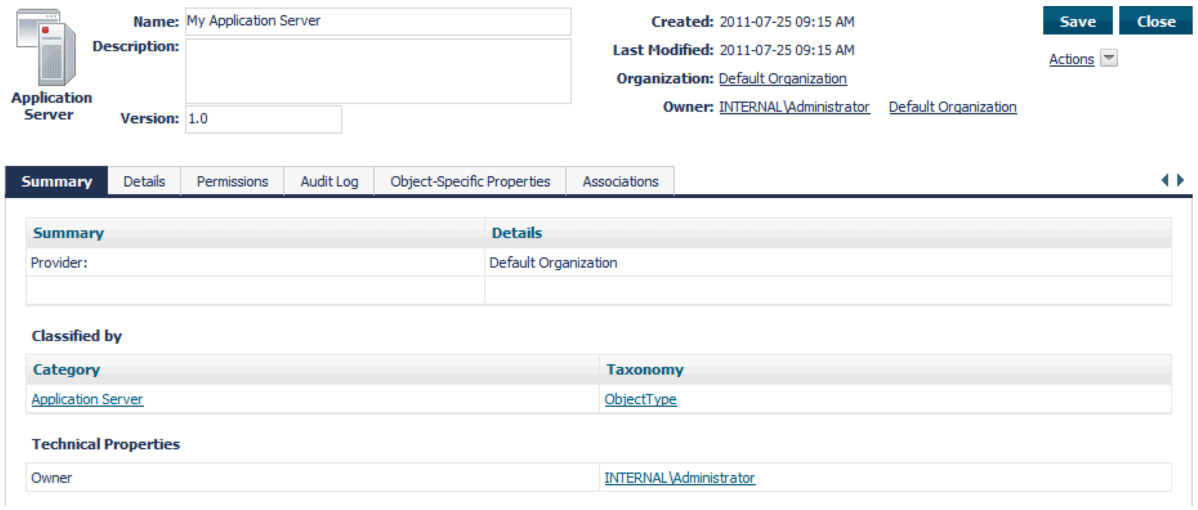

Add Asset

Enter the details for the asset you want to create.

* Type:	Application Server
* Name:	My Application Server
Description:	
* Organization:	Default Organization
Initial Version:	1.0

4 Click the **OK** button.

The newly created asset is now shown.



Name:	My Application Server
Description:	
Version:	1.0
Created:	2011-07-25 09:15 AM
Last Modified:	2011-07-25 09:15 AM
Organization:	Default Organization
Owner:	INTERNAL\Administrator Default Organization

Summary	Details
Provider:	Default Organization

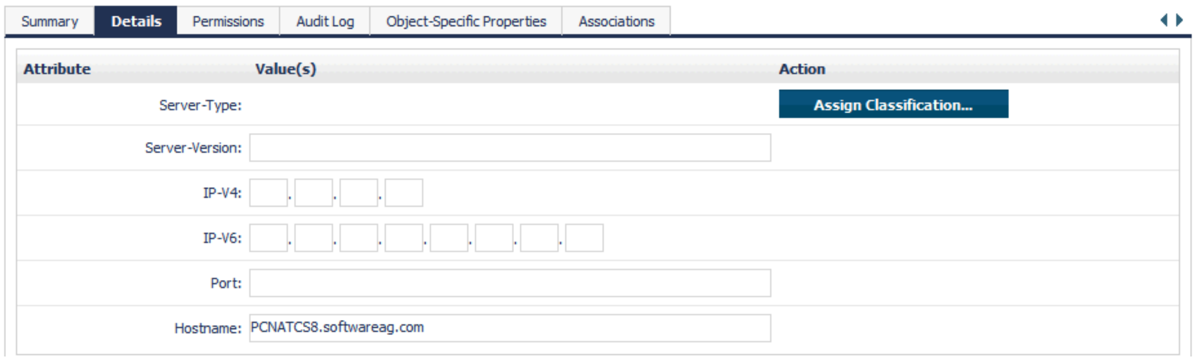
Classified by

Category	Taxonomy
Application Server	ObjectType

Technical Properties

Owner	INTERNAL\Administrator

5 Go to the **Details** tab of the new asset and in the **Hostname** text box, specify the name of the host on which your application server is running. You need not specify any other information on this page. Example:

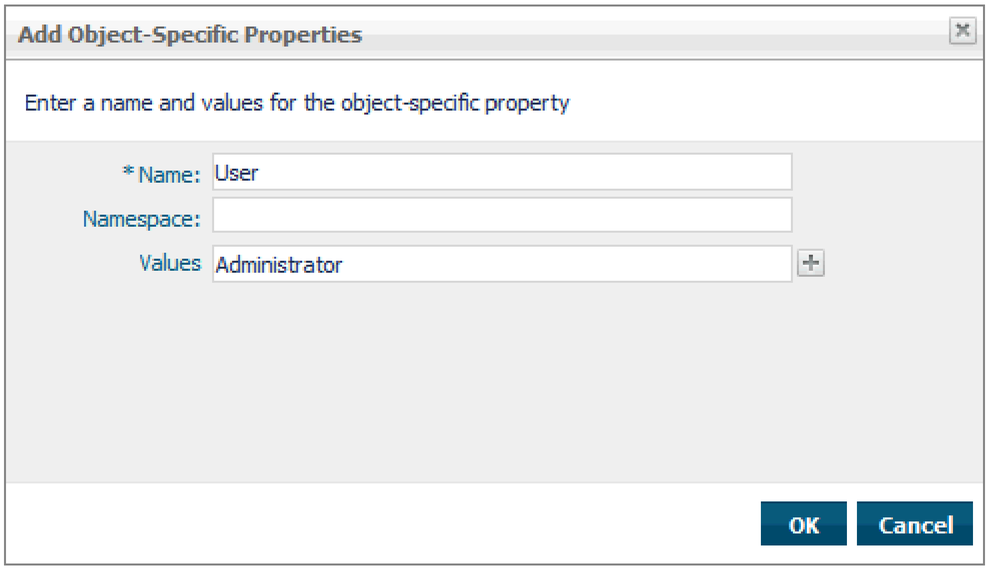


Attribute	Value(s)	Action
Server-Type:		Assign Classification...
Server-Version:		
IP-V4:		
IP-V6:		
Port:		
Hostname:	PCNATCS8.softwareag.com	

Note: If a port is specified, this information is not used by the Lifecycle Manager since the host is accessed via SFTP.

6 Go to the **Object-Specific Properties** tab and add all properties that are required by the Lifecycle Manager. For detailed information on the required properties, see ***Object-Specific Properties*** under *Application Server* in the section *Asset Types for Natural Applications*.

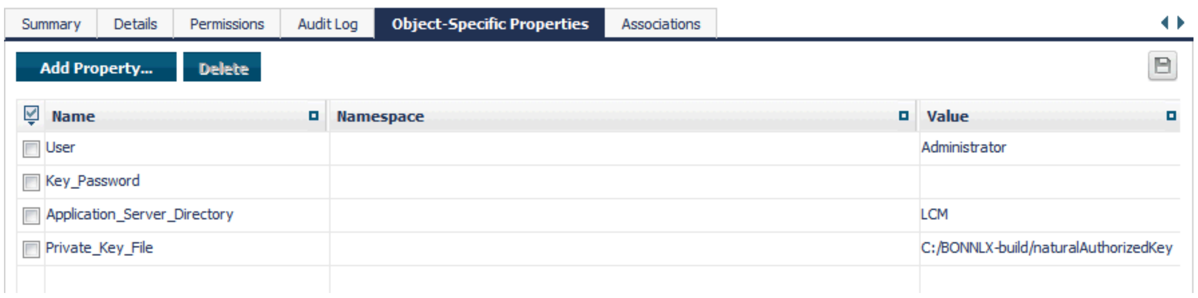
For each property to be added, click the **Add Property** button, specify a name and a value, and then click the **OK** button. Example:



Enter a name and values for the object-specific property

* Name:	User
Namespace:	
Values	Administrator

When all properties have been added, the **Object-Specific Properties** tab should look similar as in the following example:



Name	Namespace	Value
User		Administrator
Key_Password		
Application_Server_Directory		LCM
Private_Key_File		C:/BONNLX-build/naturalAuthorizedKey

Adding an Association for an Application Server to a Natural Project

After you have added the asset for your application server, you have to add an association for this application server to your Natural project.

To add an association for an application server

1 Go to the **Next Installation Environments** tab of your Natural project.

2 Click the **Modify** button.

The **Add Associations** dialog appears.

3 Click the **Search** button (on the Keyword tab).

All Application Server assets that are currently defined are now shown at the bottom of the dialog (the defined Natural Environment assets are also shown).

4 Enable the check box in front of your Application Server asset.

5 Click the **OK** button.

This closes the **Add Associations** dialog.

On the **Next Installation Environments** tab, the name of your Application Server asset is now shown next to **To be installed next**.

Note: When you check the **Summary** page of your Application Server asset, you can see that the name of your Natural project and the defined association type are now shown there.

Actions for the User Interface Components

You build and deploy a Natural project which is used to administer the user interface components on the application server in the same way as you build and deploy a traditional Natural project: you change its lifecycle state. See also ***Deploying the Assets According to the Sample Lifecycle Model for Natural Projects***.

Information on the steps that an action performs for the user interface components is given in the topics below:

- Checkout and Compile
- Install

Checkout and Compile

It is important that the sources of the user interface components are stored in a versioning repository from where they are checked out into a build directory. Other than for traditional Natural applications, the Lifecycle Manager does not provide any policy that copies the sources of the user interface components from an Eclipse workspace to a build directory or which commits these sources to a versioning repository.

For checking out the user interface components from the versioning repository and compiling them, the same action is used as for the traditional Natural sources: "SAG Checkout and Compile".

How the user interface components are handled is determined by the deployment mode that is defined in the WAR deployment file. NaturalONE's deployment wizard for web applications offers the following options for this purpose:

Option	Description
Deploy as war archive	When this option has been selected, the "SAG Checkout and Compile" action executes Ant with the given deployment file. The resulting *.war* archive and the deployment file are copied to the *Version_<n>* subdirectory of the build directory.
Deploy as file export	When this option has been selected, the "SAG Checkout and Compile" action just creates a *Version_<n>* subdirectory (if it does not yet exist) and copies the deployment file to this subdirectory.

Install

For installing the user interface components on the application server, the same action is used as for the traditional Natural sources: "SAG Install Natural Project via NDV Server".

How the user interface components are handled is determined by the deployment mode that is defined in the WAR deployment file. NaturalONE's deployment wizard for web applications offers the following options for this purpose:

Option	Description
Deploy as war archive	When this option has been selected, the "SAG Install Natural Project via NDV Server" action copies the *.war* archive from the *Version_*`<n>` subdirectory of the build directory to the specified directory on the machine running the application server.
Deploy as file export	When this option has been selected, the "SAG Install Natural Project via NDV Server" action copies the deployment file from the *Version_*`<n>` subdirectory of the build directory to the machine running the application server and then starts Ant on this server in order to package the web application. The deployment is started with the following Ant call: `ant -lib . -f deployment-file-name.xml checkout build` **Important:** Make sure that Ant and SVN have been installed on the application server.

7 Asset Types for Natural Applications

- Natural Environment 72
- Natural Project 74
- Compound Project 81
- Application Server 83

Natural Environment

A Natural Environment asset defines the target environment in which a Natural project is to be installed.

The following tabs are available for this asset type:

- Environment Attributes
- Permissions
- Object-Specific Properties
- Audit Log
- Associations

Environment Attributes

The **Environment Attributes** tab is a Lifecycle Manager-specific tab. The following attributes are available:

Attribute	Description
Database ID	The database ID (DBID) of the Natural system file.
File Number	The file number (FNR) of the Natural system file.
Adabas Password	The password is only required if the Natural system file has been password-protected using the Adabas security feature.
Adabas Cipher Key	The cipher key is only required if the Natural system file has been ciphered using the Adabas security feature.
NDV Host Name	The name of the Natural server (that is, the name of the machine on which Natural Development Server has been installed).
NDV Server Port	The TCP/IP port number of the Natural server.
NDV User ID	The user ID that is to be used for accessing the Natural server environment.
NDV Password	The password is only required if Natural Security is active on the Natural server.
NDV Session Parameters	Optional. Dynamic parameters that are required in the Natural server environment.

Permissions

By default, everyone in your organization is permitted to view the assets that you publish. However, only you (as the owner of the asset) and users who belong to a role with the "Manage Assets" permission for your organization are allowed to view, edit and delete these assets. To enable other users to view, edit and/or delete an asset that you have published, you must modify the asset's permission settings on the **Permissions** tab. This is a standard CentraSite tab. For further information, see the CentraSite documentation.

Object-Specific Properties

The **Object-Specific Properties** tab is a standard CentraSite tab. It is not used by the Lifecycle Manager.

Audit Log

The **Audit Log** tab is a standard CentraSite tab. It shows when and by whom a Natural project asset of a specific version has been installed in this Natural environment.

Associations

The **Associations** tab is a standard CentraSite tab. It shows which versions of Natural projects are installed or are meant to be installed in this Natural environment. This is only intended to facilitate the retrieval.

Caution: Do not change the information on this tab manually because this will make the Lifecycle Manager inconsistent.

For each version of your Natural project, one of the following association types is shown:

Association Type	Meaning
Installed_in	The Natural project has already been installed in the target environment.
To_be_installed_next	The Natural project has not yet been installed in the target environment.
Supersedes	This version of the Natural project has replaced the previously installed version in the target environment.

Tip: Use the (Select Columns) button at the top right of the **Associations** tab to display the **Source Object** and **Target Object** columns. You can then see the different version numbers of your assets.

Natural Project

One Natural project always applies to target environments which use the same type of operating system. For example, one and the same project can only be used for UNIX environments; it cannot be used for mainframe environments. If you have different operating systems, you have to define a dedicated Natural project for each operating system.

The following tabs are available for this asset type:

- Next Installation Environments
- Build Information
- Project Transfer Information
- Source Transfer Information
- Versioning Source Information
- Audit Log
- Versions
- Associations
- Permissions
- Object-Specific Properties

Next Installation Environments

The **Next Installation Environments** tab is a Lifecycle Manager-specific tab. The following attribute is available:

Attribute	Description
To be installed next	The list of environments in which the version of the project will be installed by executing the action which performs the installation. The environments are defined using the **Modify** button which is shown next to this attribute.

See also ***Scheduling the Installation of a Natural Project***.

Build Information

The **Build Information** tab is a Lifecycle Manager-specific tab. Information on this tab is always required (see also ***Natural Project Settings for Different Scenarios***). The following attributes are available:

Attribute	Description
Build Directory	The directory into which all sources are to be checked out from your versioning repository. This directory also contains the deployment file (see below) and all other files that can be specified on the different tabs of this Natural project. The directory which is defined in the deployment file (that is, the directory which has been specified in the wizard as the root directory) and the build directory of the Lifecycle Manager must be the same. **Important:** Do not delete the build directory and do not change the content that is generated into this directory. If you plan to access the Lifecycle Manager from different locations, make sure that the build directory is located on a shared device. See also ***What is the structure of the build directory?*** in *Frequently Asked Questions*.
Installation Type	The type of installation: Full, Incremental or Hot Fix. This describes the content of the transfer file that is to be loaded into the target environment: whether it contains a complete project or only a subset. **Note:** This information is not used by the user interface components of a Natural for Ajax application. See also ***Version Handling***.
Repository Type	The versioning repository to be used: CVS or SVN. If you do not want to use a versioning repository, you can select **None**.
Deployment File	The name of the deployment file which has been created with NaturalONE. This file must exist in (or must later be copied to) the above build directory.
Path to VCS of Natural Components *	The path to your Natural sources in the versioning repository. Relevant only when a Natural deployment file or master deployment file is used.
Tag or Revision of Natural Components *	The tag or revision number of your Natural sources in the versioning repository. Relevant only when a Natural deployment file or master deployment file is used.
Path to VCS of Ajax Components *	The path to your Natural for Ajax sources in the versioning repository. Relevant only when a WAR deployment file or master deployment file is used.
Tag or Revision of Ajax Components *	The tag or revision number of your Natural for Ajax sources in the versioning repository. Relevant only when a WAR deployment file or master deployment file is used.

* These attributes are automatically filled as soon as the appropriate policy is executed with the specified deployment file (see above).

Caution: You must not manually change the attributes which have been filled automatically because this will make the Lifecycle Manager inconsistent.

Note: The abbreviation "VCS" which is used in the above attribute names stands for "version control system".

Project Transfer Information

The **Project Transfer Information** tab is a Lifecycle Manager-specific tab. Information on this tab is always required (see also *Natural Project Settings for Different Scenarios*). The following attributes are available:

Attribute	Description

Transfer Directives File	The name of the file which describes the objects that are to be transferred. This file must exist in (or must later be copied to) the build directory. You need not specify a path to this file.
	
	This information is used by the policy which performs the unload. The syntax is the same as for the `select-clause` and `parameter-clause` of the Natural Object Handler. For detailed information, see *Utilities* in the Natural documentation for the appropriate platform.
	
	Each transfer directive must be specified in a single line. Example:
	
	`* LIB BONNLX OBJT E NATT * FMNUM 1 TONUM 100`
	`LXP06* LIB BONNLX OBJTYPE N SCKIND A NATTYPE PNS DATE TODAY WITH ↩`
	`NEWLIBRARY BERLINLX`
	`LXDDM001* LIB * OBJT D`

Transfer Directives	Instead of using a transfer directives file, you can enter the directives directly on this tab. If both is specified, only the directives on this tab are used. If you need an additional line, click the button showing the plus (+) sign.

Delete Directives File	The name of the file which describes the objects that are to be deleted. This file must exist in (or must later be copied to) the build directory. You need not specify a path to this file.
	
	This information is used in the target environment. For example, when an object has been moved from library A to a steplib, it needs to be deleted in library A. The syntax for a delete directive is the same as for a transfer directive (see above).
	
	Each delete directive must be specified in a single line. Example:
	
	`LXP6* LIB BONNLX OBJTYPE N SCKIND A NATTYPE PNS EXCEPT (LXP606)`
	`LXS01* LIB BONNLX OBJTYPE N USERID FRED EXCEPT (* DATE TODAY -21)`
	
	Note: If there are no files that are to be deleted, the specification of a delete directives file is not required.

Delete Directives	Instead of using a delete directives file, you can enter the directives directly on this tab. If both is specified, only the directives on this tab are used. If you need an additional line, click the button showing the plus (+) sign.
Repository Access Log File	The name of the file into which log information on repository check-out and update operations is written.
Compile Log File *	The name of the file into which log information on the compiled objects is written.

Attribute	Description
Object Log File *	The name of the file into which log information on the transferred objects is written.
Delete Log File *	The name of the file into which log information on the deleted objects is written.

* The above log files are automatically created as soon as the appropriate policy is executed. Their names will then be shown on this tab. When you click a name, the log information which is stored in CentraSite is shown. Using the **Download** button, you can download this log information. However, the same log information is also available in your build directory.

Important: Do not use the **Attach** button to attach files manually.

Source Transfer Information

The **Source Transfer Information** tab is a Lifecycle Manager-specific tab. Information on this tab is only required if you do not keep your sources in a versioning repository during the development phase. Instead, you develop your sources directly in the `FUSER` system file, and you use the content of a quality-assured test environment to compile the product (see also *Natural Project Settings for Different Scenarios*).

Note: This tab is not used by a Natural for Ajax application.

The following attributes are available:

Attribute	Description
Copy Sources Directives File	The name of the file which describes the sources that are to be copied from the test environment to the compile environment. This file must exist in (or must later be copied to) the build directory. You need not specify a path to this file. The syntax is the same as for the `select-clause` and `parameter-clause` of the Natural Object Handler. For detailed information, see *Utilities* in the Natural documentation for the appropriate platform. Each directive must be specified in a single line. For an example, see the description of the **Transfer Directives File** attribute under *Project Transfer Information*. Make sure to specify `SCKIND S` in this case.
Copy Sources Directives	Instead of using a copy sources directives file, you can enter the directives directly on this tab. If both is specified, only the directives on this tab are used. If you need an additional line, click the button showing the plus (+) sign.
Copy Sources Log File *	The name of the file into which log information on the copied sources is written.

Attribute	Description
Remove Objects Directives File	The name of the file which describes the objects that are to be removed from the compile environment. This file must exist in (or must later be copied to) the build directory. You need not specify a path to this file. In this case, both the sources and the generated objects are removed from the compile environment. The syntax for a remove objects directive is the same as for a copy sources directive (see above). Each directive must be specified in a single line. For an example, see the description of the **Transfer Directives File** attribute under *Project Transfer Information*. Make sure to specify `SCKIND A` in this case. **Note:** If there are no objects that are to be removed, the specification of a remove objects directives file is not required.
Remove Objects Directives	Instead of using a remove objects directives file, you can enter the directives directly on this tab. If both is specified, only the directives on this tab are used. If you need an additional line, click the button showing the plus (+) sign.
Remove Objects Log File *	The name of the file into which log information on the removed objects is written.

* The above log files are automatically created as soon as the appropriate policy is executed. Their names will then be shown on this tab. When you click a name, the log information which is stored in CentraSite is shown. Using the **Download** button, you can download this log information. However, the same log information is also available in your build directory.

Important: Do not use the **Attach** button to attach files manually.

Versioning Source Information

The **Versioning Source Information** tab is a Lifecycle Manager-specific tab. Information on this tab is only required if you do not keep your sources in a versioning repository during the development phase, and if you want to store your quality-assured sources from the compile environment in a versioning repository. This tab is usually used together with the **Source Transfer Information** tab (see also *Natural Project Settings for Different Scenarios*).

Note: This tab is not used by a Natural for Ajax application.

The following attributes are available:

Attribute	Description
Versioning Sources Directives File	The name of the file which describes the objects that are to be copied from the compile environment to the versioning repository. This file must exist in (or must later be copied to) the build directory. You need not specify a path to this file. The syntax is the same as for the *`select-clause`* and *`parameter-clause`* of the Natural Object Handler. For detailed information, see *Utilities* in the Natural documentation for the appropriate platform. Each directive must be specified in a single line. For an example, see the description of the **Transfer Directives File** attribute under ***Project Transfer Information***. Make sure to specify `SCKIND S` in this case.
Versioning Sources Directives	Instead of using a versioning sources directives file, you can enter the directives directly on this tab. If both is specified, only the directives on this tab are used. If you need an additional line, click the button showing the plus (+) sign.
Versioning Delete Directives File	The name of the file which describes the objects that are to be deleted from the versioning repository. A delete directive consists of the library name and the name of a file (max. 8 characters) plus extension, separated by a comma. Example: `MYLIB,MYFILE001.NSD`
Versioning Delete Directives	Instead of using a versioning delete directives file, you can enter the directives directly on this tab. If both is specified, only the directives on this tab are used. If you need an additional line, click the button showing the plus (+) sign.
Versioning Sources Log File *	The name of the file into which log information on the sources that are to be versioned is written.

* The above log file is automatically created as soon as the appropriate policy is executed. Its name will then be shown on this tab. When you click a name, the log information which is stored in CentraSite is shown. Using the **Download** button, you can download this log information. However, the same log information is also available in your build directory.

Important: Do not use the **Attach** button to attach files manually.

Audit Log

The **Audit Log** tab is a standard CentraSite tab. It shows when and by whom this version of the Natural project has been installed. When an installation was not successful, reasons for this error are given on this tab.

Versions

The **Versions** tab is a standard CentraSite tab. It shows all the different versions which have already been created for this Natural project and when they have been created.

Associations

The **Associations** tab is a standard CentraSite tab. It shows in which environments the current version of the project is installed. This is only intended to facilitate the retrieval.

Caution: Do not change the information on this tab manually because this will make the Lifecycle Manager inconsistent.

For each entry, one of the following association types is shown:

Association Type	Meaning
Installed_in	The Natural project has already been installed in the target environment.
To_be_installed_next	The Natural project has not yet been installed in the target environment.
Supersedes	This version of the Natural project has replaced the previously installed version in the target environment.
Contains	The Natural project is part of a compound project.

Tip: Use the (Select Columns) button at the top right of the **Associations** tab to display the **Source Object** and **Target Object** columns. You can then see the different version numbers of your assets.

Permissions

By default, everyone in your organization is permitted to view the assets that you publish. However, only you (as the owner of the asset) and users who belong to a role with the "Manage Assets" permission for your organization are allowed to view, edit and delete these assets. To enable other users to view, edit and/or delete an asset that you have published, you must modify the asset's permission settings on the **Permissions** tab. This is a standard CentraSite tab. For further information, see the CentraSite documentation.

Object-Specific Properties

The **Object-Specific Properties** tab is a standard CentraSite tab. The Lifecycle Manager supports the following properties (if required, you can add them using the **Add Property** button):

Name	Value
Trace	To activate tracing, you have to specify the value "Yes". All policy execution log files which are created during the execution of an action will then be kept. You can find them in the *Version_<n>* directory of your build directory.
Force_Hotfix	To force the installation of a hot fix without cleaning the environment, you have to specify the value "Yes". During a downgrade, this setting forces the installation of a hot fix without removing any higher versions (independent of other hot fixes, full versions or incremental versions). See also ***Installing a Hot Fix*** in the section *Version Handling*.

Important: Make sure to add the names exactly as shown above. For example, when you use lowercase characters instead of the required uppercase characters, an error will occur. Do not specify a namespace when adding a property. Namespaces are not supported by the Lifecycle Manager.

Compound Project

Compound Project assets do not have any special attributes. Such an asset is only intended as an anchor for a number of Natural Project assets that are to be installed together. The Natural Project assets are defined on the **Associations** tab (see below).

A compound project has its own lifecycle. This allows to define different versions of Natural projects that are contained in the compound project.

The following tabs are available for this asset type:

- Object-Specific Properties
- Permissions

- Associations

Object-Specific Properties

The **Object-Specific Properties** tab is a standard CentraSite tab. It is not used by the Lifecycle Manager.

Permissions

By default, everyone in your organization is permitted to view the assets that you publish. However, only you (as the owner of the asset) and users who belong to a role with the "Manage Assets" permission for your organization are allowed to view, edit and delete these assets. To enable other users to view, edit and/or delete an asset that you have published, you must modify the asset's permission settings on the **Permissions** tab. This is a standard CentraSite tab. For further information, see the CentraSite documentation.

Associations

The **Associations** tab is a standard CentraSite tab. To define the Natural Project assets which belong to the Compound Project asset, you have to add an association which uses the association type "Contains" (see the description below). The **Associations** tab will then show the Natural Project assets that belong to this Compound Project asset.

To define Natural Project assets that belong to the Compound Project asset

1. Go to the **Associations** tab of the Compound Project asset.
2. Click the **Add Associations** button.

   The **Add Associations** dialog appears.
3. From the **Please select the Association Type** drop-down list box, select **Contains**.
4. Click the **Search** button (on the **Keyword** tab).

   To limit the number of found objects, you can enter keywords or wildcards (for detailed information, see the CentraSite documentation) before clicking the **Search** button. For example, you can enter the first characters of the asset name followed by an asterisk.

   All objects that match your search criteria are shown at the bottom of the dialog.
5. Enable the check box in front of each Natural Project asset that is to be included in the compound project.
6. Click the **OK** button.

   This closes the **Add Associations** dialog. The names of all selected Natural Project assets are now shown on the **Associations** tab.

Application Server

An Application Server asset defines the target environment in which the user interface components of a Natural for Ajax application are to be installed. See also ***Natural for Ajax Applications***.

The following tabs are available for this asset type:

- Summary
- Details
- Permissions
- Audit Log
- Object-Specific Properties
- Associations

Summary

The **Summary** tab is a standard CentraSite tab. It shows which Natural projects are installed or are meant to be installed on this application server. This is only intended to facilitate the retrieval.

Details

The **Details** tab is a standard CentraSite tab. The Lifecycle Manager only uses the following attribute:

Attribute	Description
Hostname	The name of the host on which your application server is running.

Permissions

By default, everyone in your organization is permitted to view the assets that you publish. However, only you (as the owner of the asset) and users who belong to a role with the "Manage Assets" permission for your organization are allowed to view, edit and delete these assets. To enable other users to view, edit and/or delete an asset that you have published, you must modify the asset's permission settings on the **Permissions** tab. This is a standard CentraSite tab. For further information, see the CentraSite documentation.

Audit Log

The **Audit Log** tab is a standard CentraSite tab. It shows when and by whom a Natural Project asset of a specific version has been installed on this application server.

Object-Specific Properties

The **Object-Specific Properties** tab is a standard CentraSite tab. The Lifecycle Manager uses the following properties (you have to add them using the **Add Property** button):

Name	Value
User	The name of the user that is used for authentication on the SFTP/SSH server.
Key_Password	If the file specified with **Private_Key_File** is encrypted, you must specify a password.
Application_Server_Directory	The directory on the application server, relative to the configured root folder of the SFTP server. The web application will be uploaded into this directory.
Private_Key_File	A file containing a DSA or RSA private key of the user in OpenSSH key format (`-----BEGIN DSA PRIVATE KEY-----` or `-----BEGIN RSA PRIVATE KEY-----`). The private key file must be located on the CentraSite machine and the file must be specified with an absolute path.

Important: Make sure to add the names exactly as shown above. For example, when you use lowercase characters instead of the required uppercase characters, an error will occur. Do not specify a namespace when adding a property. Namespaces are not supported by the Lifecycle Manager.

Associations

The **Associations** tab is a standard CentraSite tab. It shows which versions of Natural projects are installed or are meant to be installed on this application server. This is only intended to facilitate the retrieval.

Caution: Do not change the information on this tab manually because this will make the Lifecycle Manager inconsistent.

For each entry, one of the following association types is shown:

Association Type	Meaning
Installed_in	The Natural project has already been installed in the target environment.
To_be_installed_next	The Natural project has not yet been installed in the target environment.
Supersedes	This version of the Natural project has replaced the previously installed version in the target environment.

Tip: Use the (Select Columns) button at the top right of the **Associations** tab to display the **Source Object** and **Target Object** columns. You can then see the different version numbers of your assets.

8 Natural Project Settings for Different Scenarios

- General Information 88
- Processing the Sources from a Versioning Repository 88
- Processing the Sources Directly in the Natural Environment 89

General Information

Note: The information in this chapter only applies for traditional Natural applications. It does not apply for Natural for Ajax applications where the sources of the user interface components must be stored in a versioning repository.

The Lifecycle Manager supports different scenarios for developing your Natural projects. You can decide whether you want to take the sources that are to be processed by the Lifecycle Manager from a versioning repository (this is the recommended scenario), or whether you want to take the sources directly from the Natural environment. Each scenario requires specific settings on the below mentioned tabs of a Natural Project asset.

It is always required that you specify information on the following tabs:

- Build Information
- Project Transfer Information

Information on the following tabs is only required when the sources to be processed are not taken from a versioning repository:

- Source Transfer Information
- Versioning Source Information

Detailed information on these tabs is provided in the section *Asset Types for Natural Applications*. The required settings for the different scenarios are described below.

Processing the Sources from a Versioning Repository

The recommended scenario is that you keep your sources in a versioning repository during the development and test phase. The Lifecycle Manager can then easily access the sources in the versioning repository via the deployment file.

In this case, you proceed as follows:

1. You develop and test your sources using NaturalONE.
2. You store your sources in a versioning repository. A specific tag or revision is the starting (or synchronization) point for the Lifecycle Manager.
3. You change the lifecycle state to "Built". The Lifecycle Manager takes the sources from the versioning repository, uploads them to a Natural server (NDV), compiles them there and creates a transfer file in the build directory.

4. You change the lifecycle state to "Deployed". The Lifecycle Manager installs the content of the transfer file in the target environment(s).

With this scenario, you have to specify information on the following tabs:

- **Build Information**
  - The repository type must be **CVS** or **SVN**.
  - The deployment file must contain the required settings for accessing the versioning repository.
- **Project Transfer Information**

Processing the Sources Directly in the Natural Environment

There are two scenarios:

- A versioning repository is not used at all
- A versioning repository is used to store the sources from the compile environment

A versioning repository is not used at all

When a versioning repository is not used, the processing is done directly in the Natural environment. This should be a quality-assured environment. Since a versioning repository is not used, it is your responsibility to store the sources in a safe place.

In this case, you proceed as follows:

1. You develop your sources using either Natural on a development server (NDV) or NaturalONE.
2. You store your quality-assured sources in the `FUSER` system file on a Natural server. This is the starting (or synchronization) point for the Lifecycle Manager.
3. You change the lifecycle state to "Built". There are two cases:
   - Your test environment is the same as your quality-assured compile environment:

     Sources need not be copied. However, before you start the processing with the Lifecycle Manager, you have to compile (catalog) the sources yourself. In this case, only the action "SAG Unload Natural Project via NDV Server" needs to be executed in order to create the transfer file in the build directory.
   - Your test environment is different from your quality-assured compile environment:

     Sources have to be copied; they will be compiled automatically. In this case, the following actions need to be executed: "SAG Copy and Compile" in order to copy the sources from the test environment to the compile environment and to compile them there, and "SAG Unload Natural Project via NDV Server" in order to create the transfer file in the build directory.

4. You change the lifecycle state to "Deployed". The Lifecycle Manager installs the content of the transfer file in the target environment(s).

With this scenario, you have to specify information on the following tabs:

- Build Information
  - The repository type must be **None**.
  - The deployment file must not contain any settings for accessing a versioning repository.
- Project Transfer Information
- Source Transfer Information

Note: If your compile environment is the same as your test environment, the sources need not be copied to a different environment. In this case, it is not required to specify information on this tab.

A versioning repository is used to store the sources from the compile environment

You do not want to keep the sources in a versioning repository during the development phase. Instead, you want to store the quality-assured sources in your versioning repository, after development and testing has been completed. The sources that are to be committed to the versioning repository are then taken from the compile environment. This is an extension to the above described scenario where a versioning repository is not used at all.

In this case, the following is also done when you change the lifecycle state to "Built":

- The Lifecycle Manager copies the sources from the compile environment to the build directory. You must then manually commit them to your versioning repository.

With this scenario, you also have to specify information on the following tab, in addition to the tabs that are listed under the heading *A versioning repository is not used at all*:

- Versioning Source Information

9 Version Handling

- General Information 92
- Installing the Natural Modules 93
- Installing the User Interface Components 96

General Information

For each change in your source code (which may be either an error correction or new functionality), you create a new version of your Natural project. This new version will enter the initial lifecycle state. It will then pass through the entire lifecycle and execute the policies that are defined for the different lifecycle states. The execution of the policies will end in one of the following:

- **Success**
  The state transition was successful. The relationship between the Natural project and the target environment (this may be a Natural environment and/or an application server environment) has been adapted. This means, the **To be installed next** entry has been deleted, and information on the state transition is shown on specific tabs of these assets.
- **Pending**
  The state transition is pending and can only be reverted. As a rule, this indicates that a part of the policy could not be executed successfully. The reason can be found on the **Audit Log** tab of the Natural project and in the policy log.
- **Failure**
  The state transition was not successful. The reason can be found in the policy log.

Note: To view the policy log, go to **Home > My CentraSite** and check your inbox. There you can view the list of policies that failed during events that you triggered. Or go to **Administration > Logs > Policy Log** and search for objects of type "Natural Project". If you belong to the CentraSite Administrator role, you can view all entries in the policy log.

You create a new version of a project using the **Add New Version** command. For information on how to use this command, see *Creating a New Version of a Project* in the *First Steps*.

Note: When you are using compound projects, changes to an included Natural project may make it necessary to create a new version of the compound project.

The Lifecycle Manager supports upgrade and downgrade installations:

- **Upgrade**
  If a Natural project of version 2.0 is to be installed into an environment where version 1.0 of the project is already installed, this modification is called an upgrade installation.
- **Downgrade**
  If a Natural project of version 1.0 is to be installed into an environment where version 2.0 of the project is already installed, this modification is called a downgrade installation. See also *Downgrading to a Previous Version* in the *First Steps*.

The above mentioned version numbers 1.0 and 2.0 are the internal system version numbers which are automatically generated by CentraSite. These system version numbers are independent from the version numbers you specify.

Installing the Natural Modules

For traditional Natural applications and for the Natural modules within a Natural for Ajax application, the Lifecycle Manager supports the following installation types:

- **Full Version**
  The transfer file contains the complete project. It is loaded completely into the target environments.
- **Incremental Version**
  The transfer file contains only those modules which have changed as compared to the previous version. Depending on the version which is currently installed in the target environment, the transfer file plus all intermediate incremental versions are loaded into the target environment.
- **Hot Fix**
  The transfer file contains only those modules which have changed as compared to the previous version. Only this transfer file is loaded into the target environments, regardless of the version which is currently installed in a target environment.

The installation type is defined on the **Build Information** tab of the Natural project.

Notes:

1. NaturalONE's deployment wizard for Natural applications and the Lifecycle Manager define the term “increment” differently. For the deployment wizard, “increment” means that only changed source files are to be processed. For the Lifecycle Manager, “increment” means that the objects that are defined in the transfer directives file are just a subset of the full version.
2. If you want to find out which Natural modules will be installed in the target environment, you can simulate the installation. For detailed information, see ***Simulating the Installation of a Natural Project***.

The following topics provide more detailed information on the different installation types:

- Installing a Full Version
- Installing an Incremental Version

- Installing a Hot Fix

Installing a Full Version

A full version contains all objects of a project. It may also contain a list of objects that are to be deleted from the previous version.

Upgrade

When you upgrade to a full version, the deployment starts from the version which is currently installed in the environment. The following steps are performed:

1. Objects are deleted according to the list.
2. The transfer file containing the complete project is loaded.

Downgrade

When you downgrade to a full version, the deployment starts from the version which is currently installed in the environment. The following steps are performed:

1. The transfer file containing the complete project is loaded.
2. Any objects are restored that have been deleted in the current version.

Installing an Incremental Version

An incremental version contains only the objects which have changed as compared to the previous version. It may also contain a list of objects that are to be deleted from the previous version.

Upgrade

When you upgrade to an incremental version, the deployment starts from the version which is currently installed in the environment. The following steps are performed:

1. Objects are deleted according to the list.
2. The transfer file containing the incremental version is loaded.

The above steps are repeated until the target version has been reached.

If an intermediate version is a hot fix, this version is ignored because it is expected that a hot fix solution is included in a subsequent incremental or full version.

Special case: if a full version needs to be loaded during the deployment, the upgrade directly starts with this version.

Downgrade

When you downgrade to an incremental or full version, the deployment starts from the version which is currently installed in the environment. The following steps are performed:

1. All objects are deleted which were added in a version that is higher than the target version.
2. All other objects are loaded from the transfer file of the target version and, if appropriate, its predecessor versions.

Special case: if a full version needs to be loaded during the deployment, the downgrade directly starts with this version.

A hot fix is handled in the same way as an incremental load. It is also backed out.

Installing a Hot Fix

A hot fix contains only a subset of the objects which have changed as compared to the previous version. It may also contain a list of objects that are to be deleted from the previous version.

Upgrade

When you upgrade to a hot fix, the deployment starts from the version which is currently installed in the environment. The following steps are performed:

1. Objects are deleted according to the list.
2. The transfer file containing the hot fix is loaded.

Note: Intermediate versions are ignored.

Downgrade - Standard Handling

When you downgrade to a hot fix, the deployment starts from the version which is currently installed in the environment. The following steps are performed:

1. All objects are deleted which were added in a version that is higher than the target version.
2. All other objects are loaded from the transfer file of the target version and, if appropriate, its predecessor versions.

The Lifecycle Manager assumes that an incremental version always includes all corrections that have been provided with previous hot fixes. During a downgrade, a hot fix therefore behaves in the same way as an incremental version.

Important: In case the corrections of the hot fix are not included in the subsequent incremental version, the hot fix has to be applied to the incremental version (that is, you have to create the hot fix once more and then install it).

Downgrade - Special Handling with the `Force_Hotfix` Property

You can force the installation of a hot fix without cleaning the environment. To do so, you add the `Force_Hotfix` property to the **Object-specific Properties** tab of your Natural project, and you set the value of this property to "Yes".

When you downgrade to a hot fix, the deployment starts from the version which is currently installed in the environment. The following step is performed:

- All objects are loaded from the transfer file of the target version.

Existing objects are not deleted.

The version which is currently installed in the environment remains unchanged, that is, the entry for the association type "Installed_in" is still the same after the deployment.

The Lifecycle Manager assumes that an incremental version always includes all corrections that have been provided with previous hot fixes. During a downgrade, a hot fix therefore behaves in the same way as a full version.

Caution: In case the corrections of the hot fix are also included in subsequent hot fixes, this may lead to unpredictable results.

Installing the User Interface Components

The user interface components of a Natural for Ajax application are always installed as full versions in the target environment. The installation type which is defined on the **Build Information** tab of the Natural project is not used in this case. There are no incremental versions or hot fixes for the user interface components.

With an upgrade or downgrade installation, all previously installed user interface components are deleted from the target environment and the new user interface components are installed.

Note: When you **simulate** the installation, the user interface components of a Natural for Ajax application are skipped in the resulting report.

10 Sample Policies for Natural Applications

- General Information 98
- Sample Policies for Use with an NDV Server 98
- Sample Policies for Use when an NDV Server is Not Available in the Target Environment 99
- Sample Policies for General Use 100

General Information

A policy specifies a set of actions to be executed when a specified event occurs to an instance of a specified object type. Detailed information is provided in *Working with Design/Change-Time Policies* in the CentraSite documentation. This also explains how you can create your own policies.

The sample policies which are delivered by the Lifecycle Manager are listed below.

In addition to the sample policies, the Lifecycle Manager also delivers a number of mandatory policies which ensure the integrity of your lifecycle models. These mandatory policies are automatically activated by the installation script. A list of these mandatory policies can be found in the *Installation and Configuration* section under ***Running the Installation Script***.

Sample Policies for Use with an NDV Server

The following sample policies are intended to be used when a connection to an Natural Development Server (NDV) is available.

Policy	Action(s)
SAG Build and Unload Natural Project	SAG Checkout and Compile SAG Unload Natural Project via NDV Server If you skip the "SAG Checkout and Compile" action, you must specify the following parameters for the "SAG Unload Natural Project via NDV Server" action:

Parameter	Description
NDV Host Name	The name of the Natural server (that is, the name of the machine on which Natural Development Server has been installed).
NDV Server Port	The TCP/IP port number of the Natural server.
NDV User ID	The user ID that is to be used for accessing the Natural environment.
NDV Password	The password is only required if Natural Security is active on the Natural server.
NDV Session Parameters	Dynamic parameters that are required for the Natural environment.

Policy	Action(s)
SAG Copy Sources and Compile	SAG Copy Sources and Compile Uses the same parameters for the connection as described above. The sources are taken from the environment specified by the parameters. The compile environment is specified in the deployment file.
SAG Install Natural Project via NDV Server	SAG Install Natural Project via NDV Server
SAG Retire Natural Project	SAG Retire Natural Project
SAG Simulate Installation to Specific Environment	SAG Simulate Installation to Specific Environment See ***Simulating the Installation of a Natural Project***.
SAG Uninstall Natural Project via NDV Server	SAG Uninstall Natural Project via NDV Server See ***Uninstalling a Natural Project***. The project to be uninstalled must have been installed previously in at least one of the specified environments.
SAG Version Sources	SAG Version Sources Uses the same parameters for the connection as described above. The sources are taken from the environment specified by the parameters and are committed to the repository specified in the deployment file.

Sample Policies for Use when an NDV Server is Not Available in the Target Environment

The following sample policies are intended to be used when a Natural Development Server (NDV) is not available in the target environment. In this case, the project can only be installed when the corresponding state transition has been approved. CentraSite will initiate an approval workflow and your request for a state change will be submitted to the appropriate approver via email. Note that the Lifecycle Manager has no control over the files that are installed.

Policy	Action(s)
SAG Build and Unload Natural Project on a Separate Server	SAG Checkout and Compile SAG Unload Natural Project via NDV Server If you skip the "SAG Checkout and Compile" action, you must specify the following parameters for the "SAG Unload Natural Project via NDV Server" action:

Parameter	Description
NDV Host Name	The name of the Natural server (that is, the name of the machine on which Natural Development Server has been installed).

Policy	Action(s)
	NDV Server Port — The TCP/IP port number of the Natural server. NDV User ID — The user ID that is to be used for accessing the Natural environment. NDV Password — The password is only required if Natural Security is active on the Natural server. NDV Session Parameters — Dynamic parameters that are required for the Natural environment.
SAG Request Install Natural Project on a Separate Server	Send Email Notification Initiate Approval The above actions are standard CentraSite actions. For information on how to work with email notifications and how to use approval policies, see *Working with Design/Change-Time Policies* in the CentraSite documentation.

Sample Policies for General Use

The following sample policies can always be used, either with or without a connection to an Natural Development Server (NDV).

Policy	Action(s)
SAG Additional Deployment Compound Project	Send Email Notification The above action is a standard CentraSite action. For information on how to work with email notifications, see *Working with Design/Change-Time Policies* in the CentraSite documentation.
SAG Deploy Compound Project	Send Email Notification Initiate Approval The above actions are standard CentraSite actions. For information on how to work with email notifications and how to use approval policies, see *Working with Design/Change-Time Policies* in the CentraSite documentation.
SAG Propagate Natural Project to Next State	SAG Validate Additional Installation Environments Set State
SAG Remove Natural Project	SAG Delete External Components of Natural Project
SAG Schedule Tasks	SAG Schedule Tasks You must specify the following parameters for this action: **Parameter** — **Description** CentraSite User ID — User ID used to log on to CentraSite. CentraSite Password — Password used to log on to CentraSite.

Policy	Action(s)	
	Target State	The state to which the project is to be moved after the scheduled date and time has been reached. Make sure to select the **Deployed** state from this drop-down list box. See also *Scheduling the Installation of a Natural Project*.
	Scheduling User ID	ID of the user who manages the scheduling mechanism on the current machine.
	Scheduling Password	Password of the user who manages the scheduling mechanism on the current machine.

11 Frequently Asked Questions

- Is it possible to use the Lifecycle Manager without a versioning repository? 104
- Is it possible to deploy Natural projects without a deployment file? 104
- What is the structure of the build directory? 104
- How can I find out what will be installed? 105
- What happens if two installation actions try to install into the same target environment? 105
- How can I back out the last promoted objects? 106
- What is the purpose of the "SAG Demo Model Load Natural Project on a Separate Server"? 106
- How can I activate tracing? 106
- How can I trigger state changes without using CentraSite Control? 108
- Where can I find additional information for a failed policy? 109
- Can I generate reports for the Lifecycle Manager objects? 110
- Can I use Predict Application Control for deployment? 110

Is it possible to use the Lifecycle Manager without a versioning repository?

Yes. In this case, the Natural source objects are taken from a Natural system file. The following is required:

1. Do not specify a versioning repository in the deployment file.
2. Specify the **Upload** option and the **Catalog** or **Stow** option in the deployment file.

See also ***Processing the Sources Directly in the Natural Environment***.

If your master sources are not located in a Natural system file and you do not use a versioning repository, the following is also required:

1. Copy the sources into your build directory (for example, you can copy the sources from an Eclipse workspace).
2. Make sure that the build directory has the same structure as a NaturalONE workspace.

⚠ **Important:** If you do not use a versioning repository, it is your responsibility to ensure that the sources in the system file include the desired functionality.

Is it possible to deploy Natural projects without a deployment file?

This is possible for traditional Natural applications which only contain Natural modules. This is not possible for Natural for Ajax applications since they also contain user interface components.

Without a deployment file, it is not possible to use the action "SAG Checkout and Compile". In this case, you have to check out and compile the Natural project manually. When this has been done, you have to use the action "SAG Unload Natural Project via NDV Server" in order to create the transfer file in the build directory. Make sure to specify the appropriate parameters for this action; see also ***Sample Policies for Natural Applications***.

What is the structure of the build directory?

When you create a Natural Project asset, you have to specify a build directory. If you plan to access the Lifecycle Manager from different locations, make sure that the build directory is located on a shared device.

Initially, the build directory must contain the following files:

- deployment file

- transfer directives file (or make sure to specify the directives directly on the appropriate tab of your Natural Project asset)

See also the description of the **Build Information** and **Project Transfer Information** tabs in *Asset Types for Natural Applications*.

When you build a project, the following subdirectories are generated into the build directory:

Directory Name	Description
`project-name`	This directory contains the project which has been checked out from your versioning repository. The content of this directory will be updated each time an action fetches files from the versioning repository. Keep in mind that only those files will be updated which have been modified after the checkout or last update.
Version_*<n>*	*<n>* in the directory name refers to the internal version number of your Natural Project asset. For version 1.0, a directory named *Version_1* is created. For version 2.0, an additional directory named *Version_2* is created, and so on. Each directory contains a copy of the deployment file that was used to build the project.

Caution: Do not delete the build directory. Do not change the structure or contents of the build directory (except for adding changed deployment files and directives files). Otherwise, you will not be able to install incremental upgrades or to downgrade to a previous version. All information that is required for an incremental upgrade or for a downgrade is contained in the directories which have been generated into the build directory.

How can I find out what will be installed?

If you want to find out which Natural modules will be installed into a specific environment, you can simulate the installation of a Natural project. The affected Natural modules are then listed in a report. See ***Simulating the Installation of a Natural Project***.

The user interface components of a Natural for Ajax application are always installed as full versions in the target environment. Therefore, they are skipped in the report.

What happens if two installation actions try to install into the same target environment?

When installing into a target environment using the "SAG Install Natural Project via NDV Server" action, the CentraSite lock mechanism is used to make sure that no other installation occurs in the same target environment at the same time.

When the "SAG Install Natural Project via NDV Server" action is executed, it checks whether the required objects are locked. If the required objects have been locked by another installation action, the installation action waits and tries again after 20 seconds, for a maximum of 20 minutes. When the objects are still locked after this time, an error message is issued.

If the required objects are not locked, the installation action locks the objects itself and then starts the installation.

How can I back out the last promoted objects?

See ***Downgrading to a Previous Version*** and the appropriate downgrade description in ***Version Handling***.

What is the purpose of the "SAG Demo Model Load Natural Project on a Separate Server"?

For the transitions to the "Deployed" and "Uninstalled" states, this lifecycle model uses other policies than the "SAG Demo Model for Natural Projects". It is intended to be used when a Natural Development Server (NDV) is not available.

This lifecycle model uses email notification and approval policies. The project can only be installed or uninstalled when the corresponding state transition has been approved. Note that the Lifecycle Manager has no control over the files that are installed or uninstalled.

CentraSite will initiate an approval workflow and your request for a state change will be submitted to the appropriate approver. While the request is awaiting approval, the asset will appear in the "pending" mode.

CentraSite places the approval request in the **Pending Approvals** inbox of the approver. The approver can see this request by going to **Home > My CentraSite**.

How can I activate tracing?

You can do this in two different ways:

- You add the following lines to the file *wrapper.conf* which is located in `<CentraSite-install-directory>`*/profiles/CTP/configuration*. You have to add them after the parameters defined under the heading "# Java Additional Parameters".

```
#Log4j configuration
wrapper.java.additional.7=-Dlog4j.debug
wrapper.java.additional.8=-Dlog4j.configuration="file:///C:\SoftwareAG\profiles\CTP\configuration\log4j.xml"
```

Then create a file with the name *log4j.xml* and place it in `<CentraSite-install-directory>`*/profiles/CTP/configuration*. The following is an example of a *log4j.xml* file:

```
<?xml version='1.0' encoding='UTF-8'?>
<!DOCTYPE log4j:configuration SYSTEM 'log4j.dtd'>
<log4j:configuration xmlns:log4j="http://jakarta.apache.org/log4j/" debug="true">
  <!-- switch threshold to warn to turn off debug/info -->

<!-- >>> LCM Logging configuration ====================================================-->
    <!--LCM Log4j appender-->
  <appender name='LCMLoggerAppender' class='org.apache.log4j.RollingFileAppender'>
    <param name='file' value="C:\\SoftwareAG/profiles/CTP/logs/lcm.log"/>
    <param name='MaxFileSize' value='5MB'/>
    <param name='MaxBackupIndex' value='10'/>
    <layout class='org.apache.log4j.PatternLayout'>
      <!--
   log {day month year time} down to microseconds. %M: output the method name
   (very costly, but compared to the rest of the execution hardly an issue)
   param name='ConversionPattern' value='%d{dd MMM yyyy HH:mm:ss,SSS} [%t] %-5p %c.%M %x - %m%n'
   see http://logging.apache.org/log4j/1.2/apidocs/org/apache/log4j/PatternLayout.html
   -->
      <param name='ConversionPattern' value='%d{yyyy-MM-dd HH:mm:ss} %-5p %c.%M %x - %m%n'/>
    </layout>
  </appender>

 <appender name='LCMLoggerXml' class='org.apache.log4j.RollingFileAppender'>
    <param name='file' value="C:\\SoftwareAG/profiles/CTP/logs/lcmLoggerXml.log"/>
    <param name='MaxFileSize' value='50MB'/>
    <param name='MaxBackupIndex' value='2'/>
    <layout class='org.apache.log4j.xml.XMLLayout'/>
  </appender>

<appender name="DailyRolling" class="org.apache.log4j.DailyRollingFileAppender">
<param name="File" value="C:\\SoftwareAG/profiles/CTP/logs/lcmDaily.log" />
<param name="DatePattern" value="'.'yyyy-MM-dd"/>
<param name="Append" value="true"/>
<param name="Threshold" value="DEBUG"/>
<layout class="org.apache.log4j.PatternLayout">
<param name="ConversionPattern" value="%d{HH:mm:ss:SSS} - %p - %C{1} - %m%n"/>
</layout>
</appender>

<!-- <<<LCM Logging configuration ====================================================-->

<!-- Log the Rest on Console with warn will probably end in wrapper.log-->
  <appender name='CONSOLE' class='org.apache.log4j.ConsoleAppender'>
    <layout class='org.apache.log4j.PatternLayout'>
     <!-- print datetime in ISO format, time since application startup,[thread],5-char priority, ↩
3-levels classname, message -->
      <param name='ConversionPattern' value='%d{ABSOLUTE} %-5p: %m%n'/>
    </layout>
  </appender>

<!-- >>> LCM Logging configuration ====================================================-->
  <logger name="com.softwareag.lcm">
   <level value='TRACE'/>
   <appender-ref ref='LCMLoggerAppender'/>
   <appender-ref ref='LCMLoggerXml'/>
 <appender-ref ref='DailyRolling'/>
 <appender-ref ref='CONSOLE'/>
  </logger>
<!-- <<<LCM Logging configuration ==================================================-->

  <root>
    <!-- default for all loggers (classes) log warnings, errors and fatal errors -->
    <priority value='WARN'/>
    <!-- default log to console and to file -->
 <!-- Logged every message two times, testing if the reason was that Console appender was appended ↩
twice-->
    <!-- <appender-ref ref='CONSOLE'/> -->
  </root>

</log4j:configuration>
```

Next, restart the Software AG Common Tomcat server. As a result, a file with the name *lcm.log* is written to the `<CentraSite-install-directory>`*/profiles/CTP/logs* directory. The *lcm.log* file contains important information for Software AG Support.

- Activate the trace option on the **Object-Specific Properties** tab of the Natural project. To do so, add a property with the name "Trace" and specify "Yes" as the value. All policy execution log files which are created during the execution of an action will then be kept. You can find them in the *Version_*`<n>` directory of your build directory.

How can I trigger state changes without using CentraSite Control?

You can change the lifecycle state of a Natural Project asset (for example, from "Built" to "Deployed") using a script or a command. You do this on the machine on which CentraSite Control is installed.

Your script or command has to invoke the `com.softwareag.lcm.changestate.ChangeState` class. The following arguments have to be passed to this class:

Argument	Description
`-centrasiteUrl`	URL of the CentraSite server. By default, this is *http://localhost:53305/CentraSite/CentraSite*.
`-user`	Name of the user who is authorized to trigger the status change.
`-password`	Password for the above user.
`-passwordencoded`	Whether the password is to be encoded ("yes" or "no").
`-assetKey`	UDDI key of the Natural project.
`-targetState`	Name of the target state (for example, "Deployed").
`-inmHome`	Path to the logging configuration file.

Make sure that your class path points to the following files:

`<CentraSite-install-directory>`*/redist/**
`<CentraSite-install-directory>`*/redist/csaf/**
`<CentraSite-install-directory>`*/redist/uddiv3ClientAPI/**

*<lcm-directory>/**
<lcm-directory>/ChangeState.jar

The following is a sample script which you can adapt to your requirements.

```
#!/bin/sh

# ------------- Settings -------------------------------
CP6=/opt/softwareag/CentraSite/redist/*
CP7=/opt/softwareag/CentraSite/redist/csaf/*
CP8=/opt/softwareag/CentraSite/redist/uddiv3ClientAPI/*
CP9=/home/xyz/LCM-Aux-Jars/*
CP10=/home/xyz/LCM-Aux-Jars/ChangeState.jar

# ------------- Set the Class Path ----------------------
CP=${CP6}:${CP7}:${CP8}:${CP9}:${CP10}

# ------------- URL to Centra Site ----------------------
URL=http://localhost:53305/CentraSite/CentraSite

# ------------- User Name / Default Password -----------
USR=$3
PW=<password>

# ------------- UDDI Key of Natural Project ------------
UI=<uddi-key-of-Natural-project>
# echo $UI
# ------------- Target State ---------------------------
ST="<name-of-target-state>"

# ------------- inmHome -------------------------------
INM="/opt/softwareag/profiles/CTP/configuration/log4j.xml"

# ------------- Main Program ---------------------------
java -cp $CP com.softwareag.lcm.changestate.ChangeState
 -centrasiteUrl $URL -user $USR -password $PW -passwordencoded no -assetKey $UI -targetState "$ST" ↩
-inmHome $INM
```

Where can I find additional information for a failed policy?

You can find this information in the policy log. To view the policy log, go to one of the following pages:

- Go to **Home > My CentraSite** and check your inbox. There you can view the list of policies that failed during events that you triggered.
- If you belong to the CentraSite Administrator role, go to **Administration > Logs > Policy Log** and search for objects of type "Natural Project". As an administrator, you can view all entries in the policy log.

If the policy log does not provide the information that you are looking for, go to the *Version_<n>* directory of your build directory. There you will find policy execution log files named *<action-*

`name`>*ExecLog.txt*. For example, if the transfer file could not be created by the Natural Object Handler, you will find detailed information in the file *unloadExecLog.txt*.

Explanations about the Natural exit codes which occurred can be found in the corresponding Natural log file. By default, this log file is located in the *temp* subdirectory of the directory which is defined by the `NATUSERDATA` property of the *LCM-Natural.prop file*. For further information on the Natural log file, see the description of the Natural parameter `NATLOG` in the Natural documentation for Windows or UNIX.

Can I generate reports for the Lifecycle Manager objects?

Yes. You can use the **Generate Report** command as for any other asset type. See the CentraSite documentation for detailed information.

Can I use Predict Application Control for deployment?

This is currently not possible. For a later version, it is planned to provide policies which interact with Predict Application Control.

12 Error Messages

- Version of Natural project is not an integer 113
- Error when accessing CentraSite to update an asset 113
- Predecessor version of Natural project could not be determined 113
- Association is in an invalid state 114
- Could not copy a file 114
- LCM property file could not be found 114
- Deployment file was not found 115
- Version directory could not be created 115
- SYNIN file was not written 115
- No object code file was written during the unload 116
- Policy parameters are invalid 116
- Error when accessing CentraSite 116
- Natural project does not contain KeyToHistory 117
- Error while processing the history file 117
- Only higher version can be added to history file (current: %1$s; new version %2$s) 117
- No "Next Installation Environment" defined and scheduled in Natural project 118
- Project version %1$s has not been created. Current created version of the project is: %2$s 118
- No environment defined for uninstallation 119
- Project version %1$s cannot be installed because a version has never been created for this project 119
- Problem occurred - check log files for information 119
- Could not execute Ant - check your Ant installation 120
- Could not execute SVN - check your SVN installation 120
- Build directory could not be found 120
- Version could not be determined 121
- %1$s deployment file specified in the master deployment file 121
- Project directory "%1$s" was not found 122
- Natural for Ajax application file "%1$s" was not found 122
- Update of the local SVN working copy failed 122
- Checkout of a project failed 123
- Failed to commit a local working copy to the SVN repository 123
- Invalid repository URL specified 123
- Specified value could not be found in the deployment file 124

- Neither delete nor transfer directives are set 124

Version of Natural project is not an integer

Error Number

1

Description

The "Version" property of the Natural project asset is not an integer. Normally, this error should not appear because integer version numbers are enforced by CentraSite.

Solution

Contact customer support.

Error when accessing CentraSite to update an asset

Error Number

2

Description

An error occurred when accessing CentraSite in order to update an asset.

Solution

Make sure that the CentraSite backend database is up and running. If this is the case, contact customer support.

Predecessor version of Natural project could not be determined

Error Number

3

Description

The predecessor version of a Natural project is not set.

Solution

Make sure that there is only one entry in the "Supersedes" association of the Natural project.

Association is in an invalid state

Error Number

4

Description

An association is in a invalid state.

Solution

Check the association of the given Natural project.

Could not copy a file

Error Number

5

Description

A file could not be copied to a destination.

Solution

See the associated exception for more information about the location to which the file should be copied and make sure that the destination is not write-protected.

LCM property file could not be found

Error Number

6

Description

The property file of the Lifecycle Manager could not be found.

Solution

Make sure that the property file is at the correct location in WebDAV.

Deployment file was not found

Error Number

7

Description

The specified deployment file could not be found.

Solution

Check the path to the deployment file in CentraSite.

Version directory could not be created

Error Number

8

Description

The directory "Version_<n>" could not be created in the build directory (where <n> represents the current internal version number of your Natural project).

Solution

Make sure that CentraSite has read/write privileges for the build directory.

SYNIN file was not written

Error Number

9

Description

An error occurred when writing the SYNIN file for Natrt execution. See the policy log for further information.

Solution

Make sure that Centrasite has read-/write privileges for the version directory.

No object code file was written during the unload

Error Number

10

Description

Natrt executed successfully, but the file containing the generated object code cannot be found at the expected location.

Solution

See the Natrt log files for further information.

Policy parameters are invalid

Error Number

11

Description

One parameter of the executed policy is invalid.

Solution

See the associated exception to find out which parameter is invalid. Adjust that parameter.

Error when accessing CentraSite

Error Number

12

Description

An error occurred when accessing the CentraSite database.

Solution

Make sure that CentraSite is up and running and that it is working properly.

Natural project does not contain KeyToHistory

Error Number

13

Description

The required field "KeyToHistory" of a Natural project has not been set.

Solution

Contact customer support.

Error while processing the history file

Error Number

14

Description

An error occurred while processing the history file.

Solution

See the policy log for details. This is a generic error which may have different causes. For example, there is not enough memory for CentraSite's virtual machine, or CentraSite has insufficient read/write privileges for accessing the build directory.

Only higher version can be added to history file (current: %1$s; new version %2$s)

Error Number

15

Description

A version mismatch occurred. A version with a lower version number than the latest version specified in the history file was about to be added to the history file.

Solution

Check the version of the product and the history.txt file. When the history file is modified by the Lifecycle Manager only, this error should not occur. However, if it does occur, contact customer support.

No "Next Installation Environment" defined and scheduled in Natural project

Error Number

16

Description

No environment has been defined for the installation.

Solution

Specify the environment in which the version is to be installed.

Project version %1$s has not been created. Current created version of the project is: %2$s

Error Number

17

Description

There was an attempt to install a version which has not yet been built.

Solution

The product version must be built before it can be installed. See the history.txt file of the project for the current build version.

No environment defined for uninstallation

Error Number

18

Description

A "To be installed next" environment has not been defined for the uninstallation.

Solution

To uninstall a specific version of a product in a specific environment, the product must be installed in this environment (association "installedIn") and the environment must be defined on the "Next Installation Environments" tab (association "To_be_installed_next"). Check the associations.

Project version %1$s cannot be installed because a version has never been created for this project

Error Number

20

Description

There was an attempt to install a project which has not yet been created.

Solution

The project must be created before it can be installed.

Problem occurred - check log files for information

Error Number

21

Description

An unspecified error occurred while executing the action. See the policy log for details.

Solution

A situation has occurred that can not be resolved without user intervention. If the log files do not provide a solution, contact customer support.

Could not execute Ant - check your Ant installation

Error Number

22

Description

It was not possible to execute Ant.

Solution

Check your Ant installation.

Could not execute SVN - check your SVN installation

Error Number

23

Description

An SVN installation could be found.

Solution

Make sure that SVN has been installed and that it can be accessed by CentraSite.

Build directory could not be found

Error Number

24

Description

The build directory for a project could not be found.

Solution

Make sure that the build directory has been defined and that CentraSite has read/write privileges for it.

Version could not be determined

Error Number

25

Description

The version number of a Natural project version could not be determined.

Solution

Contact customer support.

%1$s deployment file specified in the master deployment file

Error Number

27

Description

The master deployment file contains more than one Natural or WAR deployment files or one of these deployment files is missing.

Solution

Make sure that the master deployment file contains exactly one Natural deployment file and one WAR deployment file.

Project directory "%1$s" was not found

Error Number

28

Description

The name of the project directory is defined in a property of the deployment file.

Solution

Copy the project directory into the specified build directory of the Natural project.

Natural for Ajax application file "%1$s" was not found

Error Number

29

Description

The name of the Natural for Ajax application file is defined in a property of the deployment file.

Solution

Copy the Natural for Ajax application file into the specified build directory of the Natural project.

Update of the local SVN working copy failed

Error Number

30

Description

SVN failed to synchronize the local working copy of the project with the version in the repository.

Solution

Make sure that the path given in the audit message exists and it is not write-protected.

Checkout of a project failed

Error Number

31

Description

SVN failed to check out a local working copy of the project.

Solution

Make sure that the working directory of the project exists and that it is not write-protected.

Failed to commit a local working copy to the SVN repository

Error Number

32

Description

SVN failed to commit a local working copy of the project.

Solution

See the audit log for further information. The versioning sources log file may also contain additional information.

Invalid repository URL specified

Error Number

33

Description

The given SVN repository URL is invalid. No repository could be found at that location.

Solution

Specify the correct repository URL.

Specified value could not be found in the deployment file

Error Number

34

Description

A value expected in the deployment file could not be found.

Solution

Make sure that the value given in the detailed description of the error is defined in the deployment file.

Neither delete nor transfer directives are set

Error Number

35

Description

Transfer directives and delete directives have not been defined.

Solution

Make sure that at least one of the directives (transfer or delete) is defined, either by specifying a file containing the directives or by specifying the directives directly in the user interface.