CA Lisa Release Automation 4.7

Zero Touch Deployment

Author : Keith Puzey

Version: 1.3

Filename: Zero Touch Deployment of CA Release Automation 4-7.docx

Date: Thursday, 17 July 2014

Contents

CA Release Automation Components

Database Component

CA Release Automation supported Database servers:

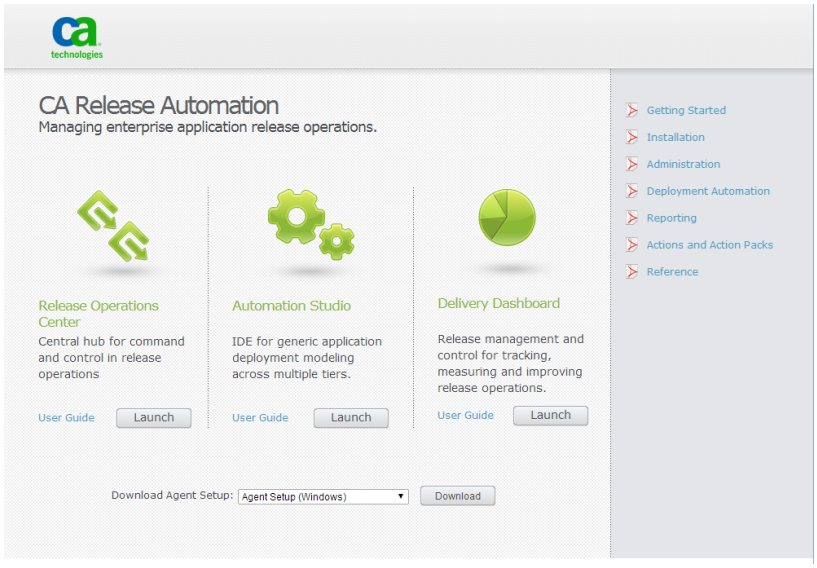
MySQL Version 5.1.50 and Higher

Oracle Version 10g and Higher

Microsoft SQL Server Version 2008 and Higher

Management Server (NAC)

The Data Management Server is the central point of a "CA Release Automation" deployment it has several functions which include the scheduling and process control, when a process is executed within the data management server the relevant instructions are sent to the agents via the Execution Servers that are assigned to the agents. The Data Management server also includes the UI server, when connecting to the Data Management Server with a browser the landing page has launch points to three UI's:



Release Operation Center – ROC, Data manager

As the controller of the system, the Center Server drives all executions, manipulates data in the database, calculates and updates flows states. The Center Server also temporarily holds all files that are needed for currently started executions.

Designer (ASAP)

IDE for generic application deployment modelling across multiple tiers. When launching the Automation Studio interface a Java Web Start application is downloaded to the local machine (Java JRE is required). The Java web start ensures that the latest client will be installed and updated.

Delivery Dashboard

Release management and control for tracking, measuring and improving release operations (Adobe Flash Player plugin required).

Repository

The repository provides storage for artifacts to be used in Release Operations Center releases and for all actions, core and custom, to be used in Automation Studio processes. The repository server (Nexus) is installed by default as part of the Data Management server installation. The installation of the Data manager includes an option to connect to an external repository.

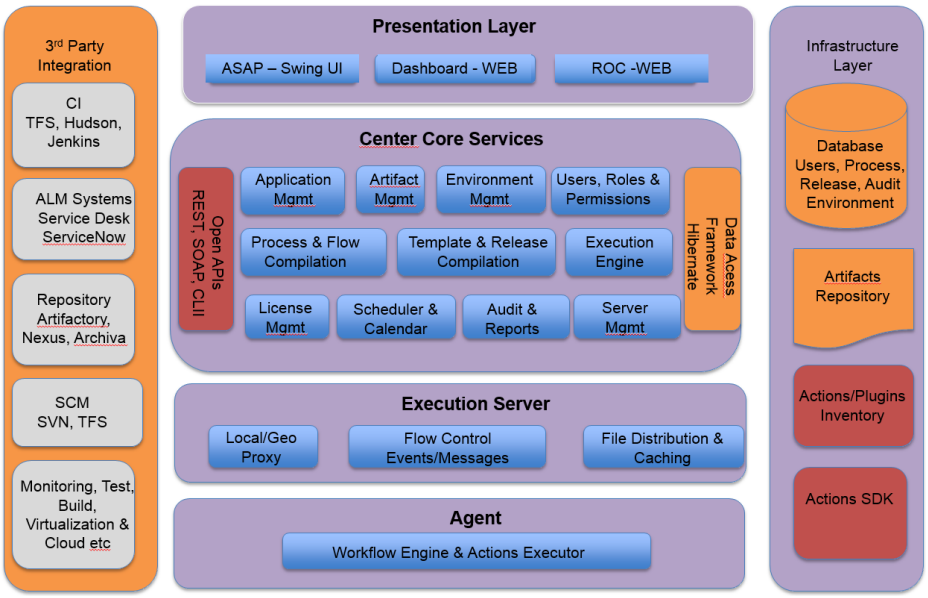
Execution Server (NES)

A CA Release Automation deployment must have at least one Execution server, The Execution Server mediates between the Center Server and the Agents. The Execution Server distributes control messages and files to the Agents and collects status data from the Agents which is retrieved later by Center Server. The Execution Server serves as a bridge when an Agent needs to Communicate with other Agents and a direct route is not available. More than one Execution Server can participate in the communication between two agents.

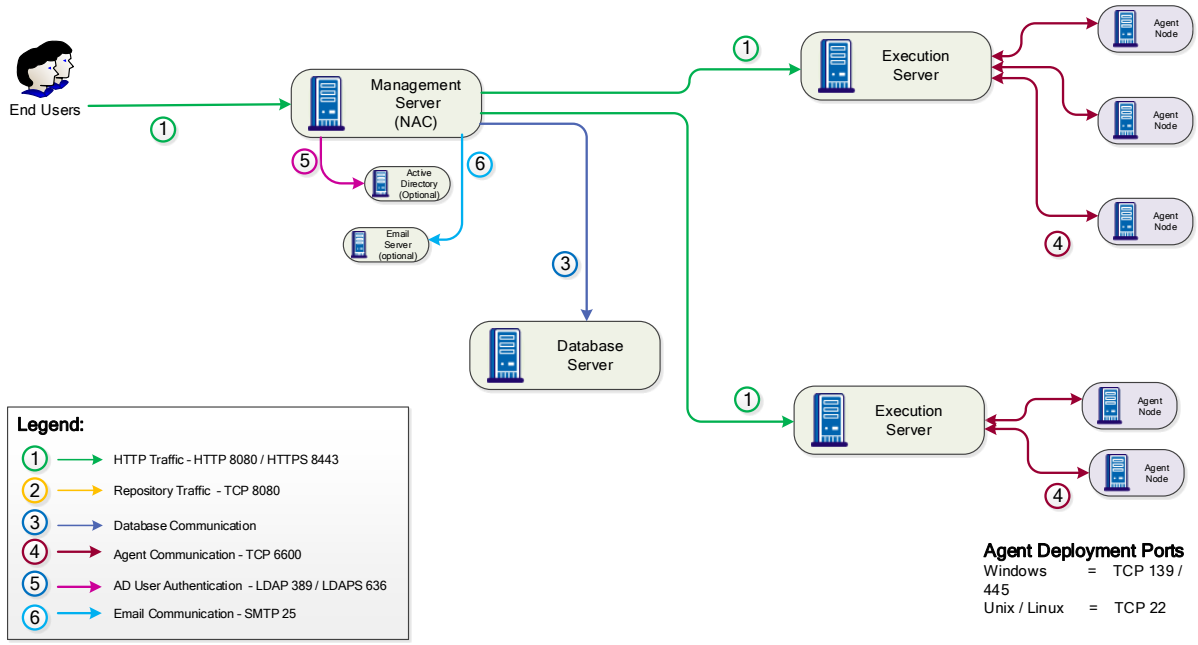
Agents (AGT)

All Machines that are part of the CA Release Automation environment require an Agent to be installed; agent installation can be remotely deployed from the IDE administration screen or installed manually. Agents are the main workers of the system. Each Agent is deployed on a server which it manages. The Agent gets the process flow and files from the Execution Server and sends process execution status back to the Execution Server. Agents can also be used to retrieve artifacts from the Repository.

Logical Architecture



High Level Component and Port Architecture



The schematic shows the various components that make up a typical CA Release Automation deployment.

Automatic NES Configuration

The following sections contain details for silently installing the primary components of CA Lisa Release Automation, once the components are installed the NAC needs to be configured to connect to the installed Execution Servers this can be achieved by using the Administrator functionality within the Designer UI. This can also be achieved using a Zero touch approach by adding the details of the deployed Execution servers to the Release Automation Database.

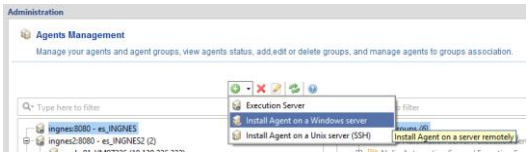
The table that needs to be updated is called “exec_servers” and the following query can be used to insert the relevant data of the Execution Server. The NAC will need to be restarted after inserting the Execution server details for the discovery process to complete.

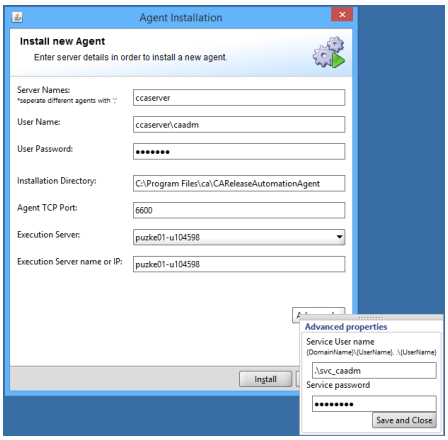
```
INSERT INTO exec_servers (ES_ID, hostname, jxta_name, scheme, port, reachable)
VALUES (‘ESID’ 'NESIPAddress', 'es_NESNAME', 'HTTP', '8080', 1);
```

Value Name	Description
ESID	This value should be a Unique Number
Hostname	IP address or Hostname of the Execution Server
Jxta_name	Hostname of Execution server preceded by es_ i.e es_Executionserver1
Scheme	HTTP or HTTPS
Port	Default port is 8080
Reachable	1

Agent Silent installer details

The designer UI includes the functionality to remotely deploy agents from the Agent Management section which can be found on the Administration tab.





When deploying to Windows the advanced properties can be used to define the credentials the agent windows service will run under. To deploy an agent to windows remotely the credentials entered in the UI need administrator access to the agent machine. The deployment connects to the share ADMIN$, if UAC is enabled the members of the administrators group do not have access to the share please refer to this Microsoft tech document.

The Agent installer can also be run silently using the command line and response file which can be used when deploying the agent using existing software deployment applications.

Preparing the Agent installation folder

1. Create an installation folder.
2. You can download the Release Automation Agent Installation file in one of two ways:

(i) Goto http://localhost:ReleaseAutomation:8080 and download the agent installation file (Windows or Linux) to the installation folder.

(ii) Copy the relevant agent installation file (**nolio_agent_windows*.exe** for Windows or **nolio_agent_linux*.sh** for Linux) from **<Release Automation Install dir>/scripts** to the installation folder.

(iii) Copy the **agent.response.varfile** file from **<Release Automation Install dir>/scripts** to the installation folder.

Installing Release Automation Agents with a response file

1 Copy the **agent.response.varfile** file from **<Release Automation Install dir>/scripts** to the installation folder
2 Modify the **agent.response.varfile** with changes to the installation folder and ports
3 For non-Windows machines, grant "a+x" permission to the installation file:
   a. chmod a+x nolio_agent_<OS>_4_7_0_b<#>.sh
4 Execute the installation file:
   a. ./nolio_agent_<OS>_4_7_0_b<#>.sh –q –varfile response.varfile

Note: Additional flags can be added to the command line for logging more information.
./ nolio_agent_<OS>_4_7_0_b<#>.sh –q –varfile response.varfile –Dinstall4j.alternativeLogfile=[path] -Dinstall4j.keepLog=true

Installing Release Automation Agents without a response file

In every server you want to install Release Automation Agent, run the following command from the installation file directory:

- In Linux:./nolio_agent_linux_*.sh -q -dir (Install dir) -varfile agent.response.varfile -Vsys.installationDir=(install dir) -Vnolio.execution.name=(Execution server name) -Vnolio.execution.port=6600 -Vnolio.nimi.port=6600 -Vnolio.check.connectivity=false -Vnolio.nimi.secured=false

- In Windows: ./nolio_agent_windows_*.exe -q -dir (Install dir) -varfile agent.response.varfile -Vsys.installationDir=(install dir) -Vnolio.execution.name=(Execution server name) -Vnolio.execution.port=6600 -Vnolio.nimi.port=6600 -Vnolio.check.connectivity=false -Vnolio.nimi.secured=false

Note: -Vnolio.execution.port and -Vnolio.nimi.port are the ports in which the Nolio Execution server is configured to work. (default: 6600)
Vnolio.nimi.secured - specifies whether the communication between the agents and the Execution server is encrypted (default: false)

Optional values for CA Release Automation agents:

The following parameters can be used when deploying windows agents and requiring the windows service to run as a certain user defined with the parameters:

-Vnolio.service.user=[Service Account Username]
-Vnolio.service.pw=[Service Account Password]

Note: If this parameter is used the user must exist and be a member of the local administrator's group and have the right to Login Locally as a service

Verify Installation

Once the installation is complete, open the Agent Management menu in the Release Automation, to confirm that the agents are listed.

Manager Components Silent installer details

Preparing the Manager Installation folder

1. Create an installation folder.
2. Download the Release Automation installation media to the installation folder
3. Copy the modified **Manager.response.varfile** file to the installation folder.

Installing Release Automation Agents with a response file

1 For non-Windows machines, grant "a+x" permission to the installation file:
   a. chmod a+x nolio_server_ <OS>_4_7_0_b<#>.sh
2 Execute the installation file:
   a. ./ nolio_server_ <OS>_4_7_0_b<#>.sh –q –varfile response.varfile

Note: Additional flags can be added to the command line for logging more information.
./ nolio_server_ <OS>_4_7_0_b<#>.sh –q –varfile response.varfile –Dinstall4j.alternativeLogfile=[path] -Dinstall4j.keepLog=true

The following sections contains examples of response files for an Execution server only installation and a NAC / Oracle and Execution Server installation.

Execution Server Only Variable Response File

```
## MAIN Information
#Installation Path (if windows, path should include '\\' replacing single '\'.
# For example: C\:\\Program Files\\Nolio\\NolioAutomationCenter
sys.installationDir=C\:\\Program Files (x86)\\CA\\LISAReleaseAutomationServer
#Installation Type 0-CleanInstall, 1-Upgrade
nolio.installation.type.user$Integer=0
#Supernode IP Address (should include value only if server installation includes an Execution Server)
nolio.agents.supernode=127.0.0.1
#Execution Server Name (the name of the Execution Server as known to the Agent machines)
nolio.execution.name=<Execution_Server>
#Execution Server Node Name (a unique node id)
#Program Group Name
sys.programGroupName=CA
#Nolio Serice as Local System (true if installed with LocalSystem account. otherwise, false)
install.service.lsa$Boolean=false
#Nolio Service Password (blank if using LocalSystem account. otherwise, service password)
nolio.service.pw=<Service_Password>
#Nolio Service User (blank if using LocalSystem. otherwise, service owner. If the installation will use
mssql windows authentication, the service user must be defined administrator in the mssql database.
When using the format domain\user, the format should include '\\' replacing single '\'. For example:
mydomain\\myusername.)
nolio.service.user=<Service_Username>
#Add shortcut of Nolio to Desktop (true of false)
createDesktopLinkAction$Boolean=true
## DB Variables
#DB TYPE 0-MYSQL, 1-MSSQL, 2-ORACLE
# nolio.db.type$Integer=<DatabaseType>
#DB Hostname or IP Address
#nolio.db.host.name=<DBHostName>
#DB Username (when installing with mssql windows authentication, leave blank)
#nolio.db.user.name=<myusername>
#DB Password (when installing with mssql windows authentication, leave blank)
```

#nolio.db.password=<mypassword>

#DB Schema Name (for more details what is expected as database name see

instructions in the installation and administration guide)

#nolio.db.database.name=<mydatabasename>

#DB Create Schema (true or false. If set to false, database will not be created

and installation supports only DM and ES installation)

nolio.db.create$Boolean=false

#DB Port

#nolio.db.port=<DatabasePort>

#DB Demo create true or false (can be ignored)

nolio.db.isempty$Boolean=true

#MSSQL DBA User name (mssql instance username that can create database. Blank if not using mssql

or using mssql windows authentication)

nolio.db.mssql.dba.user=

#MSSQL DBA Password (blank if not using mssql or using mssql windows authentication)

nolio.db.mssql.dba.password=

#MSSQL Windows Authentication (true or false. blank if not using mssql)

nolio.db.mssql.winauth=false

#MSSQL DBA Windows Authentication (true or false. blank if not using mssql)

nolio.db.mssql.dba.winauth=false

#ORACLE DBA Username (oracle user that can create other users and database

objects. typically ‘sys’ or ‘system’ user. blank if not using oracle)

#nolio.db.oracle.dba.user=<OracleDBAUserName>

#ORACLE DBA Password (blank if not using oracle)

#nolio.db.oracle.dba.password=<OracleDBAPassword>

#ORACLE Tablespace Name (blank if not using oracle)

#nolio.db.oracle.tablespace=

#ORACLE DATAFILE Name (blank if not using oracle)

nolio.db.oracle.tablespace.file=

Ports Variables

#Agent NiMi PORT (default is 6900)

nolio.nimi.port=6900

#Execution Server NiMi Port (default is 6600)

nolio.execution.port=6600

#TOMCAT HTTP Secured Port (default is 8443)

tomcat.port.ssl=8443

#TOMCAT AJP PORT (default is 8009)

tomcat.port.ajp=8009

#TOMCAT HTTP Port (default 8080)

tomcat.port.http=8080

#TOMCAT Shutdown Port (default is 8005)

tomcat.port.shutdown=8005

#JMX FLAG (false if using default setting. True if the JMX port is to be changed)

nolio.hiddenport$Boolean=false

#JMX Port (provide value only if nolio.hiddenport$Boolean is set to true. Default is 20203)

port.hidden=20203

GENERAL parameters

#Installation Type 0 - Complete, 1 - Custom

nolio.install.type$Integer=1

#Mark to install Data Management (true or false)

nolio.install.dm$Boolean=false

#Mark to install Execution Server (true or false)

nolio.install.es$Boolean=true

#Mark to install Agent (true or false)

nolio.install.agent$Boolean=false

#Mark to Demo Processes (not at use)

nolio.install.flows$Boolean=false

#BRANDING TYPE (nolio or branded company. If left blank will use nolio)

nolio.branding.name=nolio

#Just for Windows - STARTUP Menu

sys.programGroupDisabled$Boolean=false

#Additional Execution Flag

sys.component.12751$Boolean=true

#Server Infrastructure Flag

sys.component.336$Boolean=true

#Additional DM Flag

sys.component.12750$Boolean=false

Other parameters

#Product Activation Key

nolio.product.key=1002336406

Language - not in use

sys.languageId=en

#Nimi Supernode

nolio.nimi.supernode=default

#Nimi Secured flag (true of false. will be update for both Execution Server and Agent. must be aligned between these components)

nolio.nimi.secured$Boolean=false

nolio.installation.mode$Integer=0

nolio.skip.install.db$Boolean=true

sys.adminRights$Boolean=true

NAC (Oracle) and Execution Server Variable Response File

```
## MAIN Information
#Installation Path (if windows, path should include '\\' replacing single '\'.
# For example: C\:\\Program Files\\Nolio\\NolioAutomationCenter
sys.installationDir=C\:\\Program Files\\Nolio\\NolioAutomationCenter
#Installation Type 0-CleanInstall, 1-Upgrade
nolio.installation.type.user$Integer=0
#Supernode IP Address (should include value only if server installation includes
an Execution Server)
nolio.agents.supernode=127.0.0.1
#Execution Server Name (the name of the Execution Server as known to the Agent
machines)
nolio.execution.name=puzke011856-2
#Execution Server Node Name (a unique node id)
nolio.nimi.node.id=es_puzke011856-2
#Program Group Name
sys.programGroupName=CA
#Nolio Serice as Local System (true if installed with LocalSystem account. otherwise, false)
install.service.lsa$Boolean=true
#Nolio Service Password (blank if using LocalSystem account. otherwise, service password)
nolio.service.pw=<AdministratorPassword>
#Nolio Service User (blank if using LocalSystem. otherwise, service owner. If the installation will use
mssql windows authentication, the service user must be defined administrator in the mssql database.
When using the format domain\user, the format should include '\\' replacing single '\'. For example:
mydomain\\myusername.)
nolio.service.user=<ServiceUser>
#Add shortcut of Nolio to Desktop (true of false)
createDesktopLinkAction$Boolean=true
## DB Variables
#DB TYPE 0-MYSQL, 1-MSSQL, 2-ORACLE
nolio.db.type$Integer=2
#DB Hostname or IP Address
nolio.db.host.name=puzke011856-1
```

#DB Username (when installing with mssql windows authentication, leave blank)

nolio.db.user.name=nolio

#DB Password (when installing with mssql windows authentication, leave blank)

nolio.db.password=interOP123

#DB Schema Name (for more details what is expected as database name see

instructions in the installation and administration guide)

nolio.db.database.name=orcl.domain.com

#DB Create Schema (true or false. If set to false, database will not be created and installation supports only DM and ES installation)

nolio.db.create$Boolean=true

#DB Port

nolio.db.port=1521

#DB Demo create true or false (can be ignored)

nolio.db.isempty$Boolean=true

#MSSQL DBA User name (mssql instance username that can create database. Blank if not using mssql or using mssql windows authentication)

nolio.db.mssql.dba.user=

#MSSQL DBA Password (blank if not using mssql or using mssql windows authentication)

nolio.db.mssql.dba.password=

#MSSQL Windows Authentication (true or false. blank if not using mssql)

nolio.db.mssql.winauth=false

#MSSQL DBA Windows Authentication (true or false. blank if not using mssql)

nolio.db.mssql.dba.winauth=false

#ORACLE DBA Username (oracle user that can create other users and database

objects. typically ‘sys’ or ‘system’ user. blank if not using oracle)

nolio.db.oracle.dba.user=sys as sysdba

#ORACLE DBA Password (blank if not using oracle)

nolio.db.oracle.dba.password=interOP123

#ORACLE Tablespace Name (blank if not using oracle)

nolio.db.oracle.tablespace=nolio_tbs

#ORACLE DATAFILE Name (blank if not using oracle)

nolio.db.oracle.tablespace.file=nolio_tbs.dbf

Ports Variables

#Agent NiMi PORT (default is 6900)

nolio.nimi.port=6900

#Execution Server NiMi Port (default is 6600)

nolio.execution.port=6600

#TOMCAT HTTP Secured Port (default is 8443)

tomcat.port.ssl=8443

#TOMCAT AJP PORT (default is 8009)

tomcat.port.ajp=8009

#TOMCAT HTTP Port (default 8080)

tomcat.port.http=8080

#TOMCAT Shutdown Port (default is 8005)

tomcat.port.shutdown=8005

#JMX FLAG (false if using default setting. True if the JMX port is to be changed)

nolio.hiddenport$Boolean=false

#JMX Port (provide value only if nolio.hiddenport$Boolean is set to true. Default is 20203)

port.hidden=20203

GENERAL parameters

#Installation Type 0 - Complete, 1 - Custom

nolio.install.type$Integer=1

#Mark to install Data Management (true or false)

nolio.install.dm$Boolean=true

#Mark to install Execution Server (true or false)

nolio.install.es$Boolean=true

#Mark to install Agent (true or false)

nolio.install.agent$Boolean=true

#Mark to Demo Processes (not at use)

nolio.install.flows$Boolean=false

#BRANDING TYPE (nolio or branded company. If left blank will use nolio)

nolio.branding.name=nolio

#Just for Windows - STARTUP Menu

sys.programGroupDisabled$Boolean=false

#Additional Execution Flag

sys.component.12751$Boolean=true

#Server Infrastructure Flag

sys.component.336$Boolean=true

#Additional DM Flag

sys.component.12750$Boolean=false

Other parameters

#Product Activation Key

nolio.product.key=1002336406

Language - not in use

sys.languageId=en

#Nimi Supernode

nolio.nimi.supernode=default

#Nimi Secured flag (true of false. will be update for both Execution Server and Agent. must be aligned between these components)

nolio.nimi.secured$Boolean=false

nolio.installation.mode$Integer=0

nolio.skip.install.db$Boolean=false

sys.adminRights$Boolean=true