

# NASA Workflow Tool

## Administrator Guide

September 29, 2010

---

[NASA Workflow Tool](#)

[Administrator Guide](#)

[1. Overview](#)

[2. Deploying the NED Server](#)

[3. NED Server Operations](#)

---

## 1. Overview

This document covers building, installing, configuring, and managing the NED related software.

The NED software suite is comprised of two key components:

- **NED Client:** The client software distributed to end users. This communicates with server for performing actions like submission.
- **NED Server:** The server software that carries out actions such as interacting with repositories, generating necessary workflow files, and running/monitoring the workflows.

Additional software packages that are useful to workflow developers include:

- **LWWE Workflow Editor:** This is a standalone application for creating, editing, and testing workflows. It contains an embedded instance of the LWWE workflow engine.
- **NED Server Manager:** Allows limited remote monitoring and control of a NED server.

## 2. Deploying the NED Server

### System Requirements for NED

Below are our general system admin-level requirements for the Workflow Tool portion:

- Java 1.6 or later is required
- The JAVA\_HOME environment variable must be set
- The PATH environment variable must contain Java
- The account running the server/workflows should be set to umask to 0002
- For convenience, the stacksize should be set to unlimited for all users
- Common group code for all users set as the DEFAULT group for users
- A user account for running the server and workflows

- An open port that corresponds to the server port, unless tunnels are being used
- Shared folders for installing all tools and various libraries
- Subversion and/or CVS client for workflow storage is recommended
- eXist XML database is recommended
- Additionally, workflows themselves may impose their own requirements, including:
  - Python 2.6 or 2.7
  - marco 0.21 (shared workflow utility package)
  - Tunnel to the cvsac GEOS5 source control server

It is recommended to set up a group specific to workflow administration. This prevents general users from accidentally or maliciously altering essential server configurations.

### Java

Java version 1.6 or later must be installed on the system in order to run any of the NED software. The JAVA\_HOME environment variable must be set universally for all accounts, and the java binaries added to the default PATH. This will prevent numerous issues related to starting the software.

### Group Codes

It is highly recommended that group codes are established to delineate user privileges on the machine. The recommended basic setup for these group codes are:

- Workflow User: This group code refers to general users of the workflow tool. Users belonging to this group will be able to access necessary directories and files for working with the tools.
- Workflow Design: This group code refers to the users who also create/implement workflows. This group has further directory access to allow writing to “shared experiment” directories that other users use for their workflows
- Workflow Admin: This group code refers to users who have administrative privileges in regards to the tool. These users have access to all workflow directories and also have the capability of performing server/software maintenance.

Typical names for these groups are wf\_user, wf\_designer, and wf\_admin respectively.

### Workflow Account

An account must be established for running the workflow server. Typically, the account is named workflow. The workflow account must be made a member of all workflow groups (user, designer, and admin).

The workflow account is responsible for running the workflow account, along with any other processes that are necessary for the workflow server. This account is also responsible for setting up any other items, such as ssh tunnels to machines or repositories.

**THE WORKFLOW ACCOUNT MUST HAVE A UMASK OF 0002 AND HAVE ITS PRIMARY GROUP AS THE WORKFLOW USER GROUP.**

If the umask is incorrect and/or the group is set incorrectly then workflows may fail to run.

Users and designers may also have issues access submitted workflows as they will not have the permissions to access the files and directories generated by the workflow tool.

### **Folders**

The workflow server requires that a working directory be set in the server configuration. This directory is where all the submitted workflows will be generated to and run from. This directory should NOT be in the workflow account, but located in a common area on disk.

Optionally, folders for “workflow testing” file repositories and shared experiments can be created. These folders can then be referenced from server components. These folders should also be placed in a shared area on the disk. See the NED server setup documentation for more details.

### **Aliases (Optional)**

Global aliases can be established for the NED Client and LWWE editor applications. This makes tool use easier for those who are trying to use the software locally on the server.

### **Software Installation**

The NED software should be installed in an area accessible by members of the workflow admin group. The software is distributed as zipped packages. To install, unzip the files. Ensure that the permissions are correct on the run scripts within the folders. The NED Client and LWWE editor files should be group readable and executable for the workflow user group. The NED server folder should be locked down to only allow workflow admin access.

### **Server Deployment Package**

The server is distributed in a zip file ([server\\_deployment.zip](#)). The archive contains everything needed to run the server. Unzip the file in the location you wish the server to be located at. A run script ([runNedServer.sh](#)) is included, though depending on your environment you may need to alter the script. The other scripts included is a stop script ([stopNedServer.sh](#)) and script that generates information about a running server ([infoNedServer.sh](#)).

After placing the directory where you want it to be, you should try starting the server. This will create the [.ned\\_server\\_configurations](#) directory in your home folder which contains all the configuration files that control the server and its components.

It is likely the server will NOT start as a result of your User Profile not having the permissions. This is covered in the next section.

### **Workflow User Configuration**

Within the [.ned\\_server\\_configurations](#) directory there is a folder called UserProfiles. This folder contains user profiles. User profiles describe properties in relation to what activities users may be allowed to perform, as well as store some useful information about the user. The files are simple XML files.

The default profile is a special profile. When a user first logs into the server, a profile is created

for them. The created profile is copy of the default profile. In this manner, you can set up a consistent profile for users.

The following are the properties that can be set within a profile:

- **Name:** The name of the user.
- **Groups:** A list of entries for the group codes the user may use. This should not be edited by hand, as the server automatically populates this.
- **ServerPrivileges:** Controls what a user can do with the server.
  - **CONTROL\_SERVER:** Allows the user to start and stop the server.
  - **ADMIN\_DATABASE:** Allows the user to perform management activities on databases (not implemented).
  - **ALL:** Allows the user to perform all server activities.
  - **NONE:** The user has no special access.

In order to be able to start the server successfully, you will need to set your profile's server privileges to CONTROL\_SERVER or ALL.

After these properties is a list of "accessible directories". These are directories that the user has access to on the server. This allows for users to open workflows remotely that may be stored on the server. The following properties are used to establish an accessible directory:

- **Alias:** A user friendly name for the directory. This makes it easy to reference common shared directories.
- **Directory:** The absolute path to the directory. Note that a user who has access to a directory will also have access to it's subdirectories.
- **Permissions:** Specifies the permissions the user has on that directory
  - **READ:** The user has read access to files and directories.
  - **WRITE:** The user has write access to files and directories.
  - **NONE:** The user has no access to files and directories.

NOTE: User profile directory permissions DO NOT override operating system permissions. If a directory is not accessible to the account that the server is running under then a user will not be able to access the directory no matter what the permissions in the profile may state.

Once your profile is configured with the correct server privileges, you will be able to start the server. However, there are several aspects of the server configuration that may need to be changed for your installation.

### **Workflow Server Configuration**

There are two different areas for configuration for the NED server. One file controls the overall server, while multiple files control individual components that have been "plugged-in" to the server. All of these files are stored in the `.ned_server_configurations` directory stored in the administrator's home folder. When the server is first started the "default" configurations and properties will be created and stored here.

### **ServerConfig.xml**

This is the main overall configuration file for the NED server. It has relatively few options that

control key aspects of the server.

- **ComponentPluginDirectory**: This specifies the relative folder where the pluggable components for the server resides. On startup, the server will attempt to traverse this folder to load the components.
- **WorkingDirectory**: This specifies the directory that the server will use as a “work directory” for workflows. When a user submits a workflow, directories and files will be created under this directory. This must be an absolute path.
- **LogDirectory**: This specifies where the server should store server logs. This must be an absolute path.
- **ServerExecutionType**: Determines what mode the server will be run in. These modes are discussed later.
- **ServerPort**: The port the server will use for communication. Depending on how the server is being executed, this field may be ignored.
- **ServerPortRange**: This should be specified twice, once for the start of the range and once for the end of the range. Depending on the execution mode of the server, these fields may be ignored.

The server execution type determines how the server is run. There are several different modes for the server.

- **UNSECURED\_CENTRAL\_SERVER**: This runs the server as a central server that has no authentication. This mode should ONLY be used for debugging purposes
- **CENTRAL\_SERVER**: This is the standard mode of operation. This starts up the server using secure SSL communications and authentication. This mode uses the port field in the server configuration. This mode requires an open port.
- **MULTI-SERVER**: This is a special mode of operation for very restrictive environments, like NAS. This allows for each user to have their own remote server. This mode uses to the server port range field in the server configuration file.

### **Server Component Properties**

The NED server has a pluggable architecture that allows for new functionality to be added in a convenient way. Each component may have one or more editable properties files that may need to be customized based on how and where the server is installed.

These properties files are stored in the ServerComponentProperties directory under the .ned\_server\_configurations directory.

### **Authentication Component Configuration**

This controls how the server authenticates users. This file is a “sectioned” properties file, and should only be readable/writable by the admin user as it may contain sensitive information.

The following control which method(s) will be used for user authentication.

- **USE\_PAM\_PASSWORD** (deprecated): If true, will attempt to use PAM password authentication. This requires root level access to set up the platform specific PAM module. This will not be supported in the future.
- **USE\_PAM\_SECURID** (deprecated): If true, will attempt to use PAM passcode authentication. This requires root level access to set up the platform specific PAM

- module. This will not be supported in the future.
- [USE\\_LDAP](#): If true Will attempt to use LDAP for authentication. The LDAP properties must be set up correctly for this to work.
- [USE\\_RADIUS](#): If true, will attempt to use RADIUS SecureID for authentication. The properties under the RADIUS properties must be set correctly in order for this to work.
- [USE\\_JSCH](#): If true, will attempt to use SSH for authentication. This is the easiest method as the server simply tries to create an ssh session using the user's credentials. If it is successful, the user is authenticated.

Once the method(s) are selected, you may need to set the properties associated with the methods. This may require the you to gather information from your network administrator.

If using PAM:

- [PAM\\_JNI\\_LIB\\_PATH](#) (deprecated): Specifies the path to the platform specific JAVA-Native PAM library.

If using LDAP:

- [LDAP\\_SERVER](#): Specifies the LDAP server to use for authentication.
- [LDAP\\_GROUPS](#): Specifies the LDAP groups to use to search for users.

If using RADIUS:

- [RADIUS\\_SERVER](#): Specifies RADIUS server to use.
- [RADIUS\\_PORT](#): Specifies the RADIUS Port to use
- [RADIUS\\_AUTH\\_PROTOCOL](#): Specifies the RADIUS protocol to use.
- [RADIUS\\_SHARED\\_SECRET](#): Specifies the shared secret to use.

## Workflow Repository Manager Configuration

This component is responsible for controlling which repositories are available to the user when it comes to selecting a repository for pulling down workflow templates. The configuration file is a simple xml file that contains repository entries. There is a default entry, then a list of other entries.

All entries have the same properties:

- [Alias](#): The "user friendly" name of the repository.
- [ComponentClass](#): This corresponds to a NED repository component. Currently NED has components to handle three repositories:
  - Subversion:  
NEDServerComponents.SubversionRepository.SubversionRepositoryComponent
  - CVS: NEDServerComponents.CVSRepository.CVSRepositoryComponent
  - Flat Directory:  
NEDServerComponents.FilesystemRepository.FilesystemRepositoryComponent
- [RepositoryRoot](#): The root of the repository. The syntax of this string depends on which type of repository is being used. The root is used in conjunction with the submitted branch from a workflow to determine where to pull the workflow template from.

NOTE: The default entry MUST also exist in the list of repositories in order to operate correctly.

### **Workflow Database Manager Configuration**

This component is responsible for interacting with one or more databases to track submissions and store workflows. The configuration file is a simple xml file that contains database entries. There is a default entry, then a list of other entries.

All entries have the same properties:

- **Alias**: The “user friendly” name of the database.
- **ComponentClass**: This corresponds to a NED database component. Currently NED has only one component:
  - Exist: NEDServerComponents.eXistDatabase.eXistDatabaseComponent
- **ConnectionString**: The connection string to connect to the database. This may have a different syntax depending on the database used.
- **WorkflowsRoot**: The root location within the database where workflows are stored.
- **UniqueIDRoot**: The location within the database where unique IDs for experiments are stored.
- 

NOTE: The default entry MUST also exist in the list of databases in order to operate correctly.

### **Workflow Engine Manager Configuration**

This component is responsible for managing the workflow engine(s) that a user can use for running workflows. While there is a configuration file, it is a placeholder. Workflow engines are currently loaded automatically by the sever and require no external configuration information.

### **CVS Repository Component Configuration**

This component performs retrieval operations from a CVS repository. In order for this to work, the configuration must be set up with:

- **CVS\_RSH**: This specifies how to connect to a remote system. The default is ssh
- **CVS\_ACCOUNT**: The account name to use when performing cvs operations. The default is workflow.

### **GEOS History Component Configuration**

This is a specialized component for allowing user friendly operations in dealing with GEOS model history files. This is an optional component. The configuration file allows you to specify:

- **MACHINE**: The machine name.
- **INSTALLATIONS\_DIR**: The directory that contains the XML files that describe the template and fields for the GEOS history file.
- **SUPPORTED\_TAGS**: The supported tags of the GEOS history model.

### **SMS Engine Component Configuration (deprecated)**

This component allows for interaction with the SMS workflow engine. This is a legacy component and is here mainly to support older workflows. Newer workflows should use the LWWE engine.

The SMS configuration file only contains a single property:

- **CONNECTION\_STRING**: This is basically the path to the PRISM CDP installation. The default is `/home/workflow/prism_system/sms/bin/cdp`

Once you have ensured all configurations settings are correct, you should be able to start and use the server.

**IMPORTANT:** Unless you are operating in a firewalled environment, it is **HIGHLY** recommended that you configure the server to run with authentication enabled.

### 3. NED Server Operations

Starting the NED server is straightforward. Simply run the `runNedServer.sh` file or your customized equivalent. Depending on your operating system, you can background the server and keep it alive after you log out.

Stopping the server is equally straightforward. The `stopNedServer.sh` script will stop the currently running server started from the account.

You can also view information about a running server through the `infoNedServer.sh` script. This will list who is currently connected to the server, how long they have been connected, and other related information. As an alternative to getting info from a server and stopping a server via the command line, there is also a NED Server Manager GUI application.

**NOTE:** You can only start and stop the server if you have the correct permissions in your NED user profile.

#### Other Options

While the three scripts provided allow for basic operation, the NED server has other options that can be used for further functionality. By specifying the `-help` option on the command line you will receive an up-to-date list of options for the NED server. As of this document, the options are:

- **-start**: Starts the server. If `-port` is specified, then the server will attempt to use the specified port.
- **-stop**: Stops the server. If `-port` is specified, then the server will attempt to use the specified port.
- **-info**: Retrieves server information from the server. If `-port` is specified, then the server will attempt to use the specified port.
- **-port**: Overrides the port specified in the configuration when performing server operations.
- **-update**(experimental): Attempts to have the server re-initialize all its components while running to avoid a shutdown/restart procedure.
- **-help**: Displays a list of options the server can use.
- **-no\_warn**: Overrides the warning when the server is started in non-secure mode. This option shouldn't be used unless you really know what you're doing.
- **-config\_dir**: Specifies the configuration directory to use when starting the server. This allows for multiple servers with different configurations to run on the same machine.

**NOTE:** You can only start and stop the server if you have the correct permissions in your NED user profile.

### **Exist Database (OPTIONAL)**

NED currently comes with one database component that interacts with an eXist database. The eXist database must be installed and executing in order for the NED server to communicate with it.

To run the database, download and install the eXist database. It's easiest to use the standalone server on the local machine. To do this, navigate to the bin directory and run the [server.sh](#) script.

After the eXist server is running, you will need to add two "collections". One collection (Workflows) is used to store workflow configurations. The other collection (ExperimentIDs) is used to store unique ID's for submitted workflows). These collections can be created using the eXist DB client, which can be executed using the [client.sh](#) script. Once the collections are created the permissions should be set to guest so that the server can freely access them.

**IMPORTANT:** The collection names and locations **MUST** match the ones specified for the eXist component in the NED server. These collections are case sensitive. Failure to correctly specify the collections will result in server not being able to utilize the database.