

Apache DS tools are command line simple operations that help user to interact with the Apache DS server.

Developing the Apache DS Tools plugin for LDAP Studio, I wanted to use this tool in that plugin. But, since it is to be used with command line, it couldn't efficiently be bundled as is in the plugin.

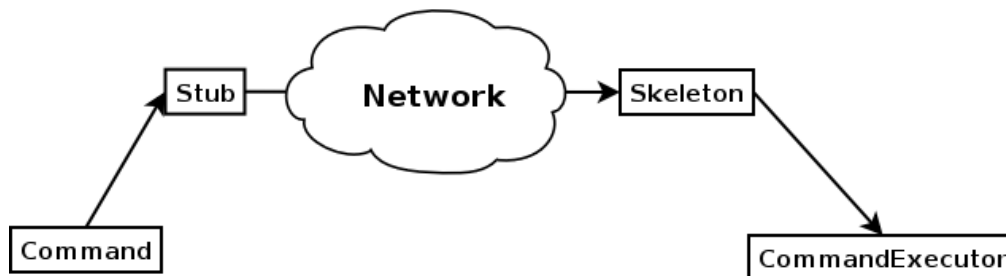
So we decided to rewrite the tools that can still be used command line, but also from any java code.

We have separated the command request from the command execution. Mainly because the server is not always located on the same machine that on which the command is executed.



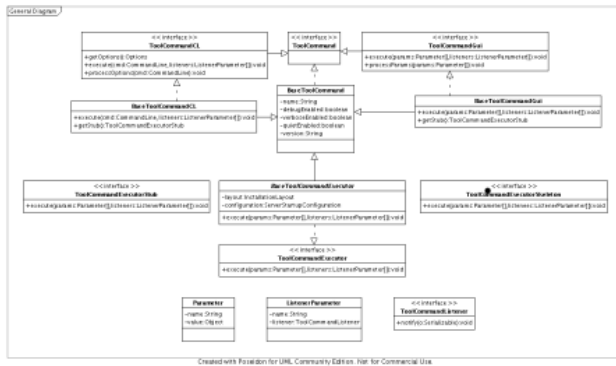
The request can now be addressed on a machine that will send the request over network to the other machine where the server is located (or directly if the server is running on the same machine).

To do so, we have to add a layer that will handle the network communication of the remote invocation. A stub and a skeleton will handle the serialization/deserialization of objects used for execution and the execution call itself.



To get information on what is going on after the request is sent to the server, specific listeners can be hooked and receive notifications (in case of debug information or on error for example). This way, it is as easy to display an error on a console (when using tools as a command line) as to launch a graphical error message box (embedding tools in a graphical application).

Here is a class diagram representing the new architecture of the Tools.



See at the bottom of this document for a larger version

On the Request side (client side), it defines two types of commands, Command Line commands (BaseToolCommandCL) and commands that can be used in a graphical application (BaseToolCommandGui).

They both extend the BaseToolCommand class, which is the class representing a simple command. Each type implements a specific interface that defines how to call the command.

On the Execution side (server side), a unique standard way of calling the execution of the command has been defined, so it is easy to define new request commands that will call the execution.

This class diagram also exposes classes and interfaces to use to implement Stub, Skeleton, Listeners, Parameters, etc.

General Diagram

