VERSION	2.6
DATE	Mai 2011
Copyright	SysFera

Table des matières

Chapitre 1

Deploying a DIET platform

Deployment is the process of launching a DIET platform including agents and servers. For DIET, this process includes writing configuration files for each element and launching the elements in the correct hierarchical order. There are three primary ways to deploy DIET.

Launching **by hand** is a reasonable way to deploy DIET for small-scale testing and verification. This chapter explains the necessary services, how to write DIET configuration files, and in what order DIET elements should be launched. See Section **??** for details.

GoDIET is a Java-based tool for automatic DIET deployment that manages configuration file creation, staging of files, launch of elements, monitoring and reporting on launch success, and process cleanup when the DIET deployment is no longer needed. See Section 1.1 for details.

Writing your own scripts is a surprisingly popular approach. This approach often looks easy initially, but can sometimes take much, much longer than you predict as there are many complexities to manage. Learn GoDIET– it will save you time !

1.1 GoDIET

GoDIET is an cross-platform tool that helps you automate ad-hoc deployment and management procedures for DIET infrastructure. It manages configuration file creation, staging of files, launch of elements, monitoring and reporting. GoDIET is extremely useful for large deployments on a complex physical infrastructure. The mains features are :

– Complete command line interface.
– Distributed command execution via SSH.
– Real time monitoring applications state.
– Complex physical infrastructure management with firewall and multiple network lan.

1.1.1 Installing GoDIET

The following operating systems are known to support GoDiet :

– Linux : Most recent distributions are likely to work
– Mac OS X 10.4 or later

You need to have the Sun Java 6 or OpenJDK6 installed. All operating system with Java 6 must support First download GoDIET on the project website[1].

1. http ://graal.ens-lyon.fr/DIET/godiet.html

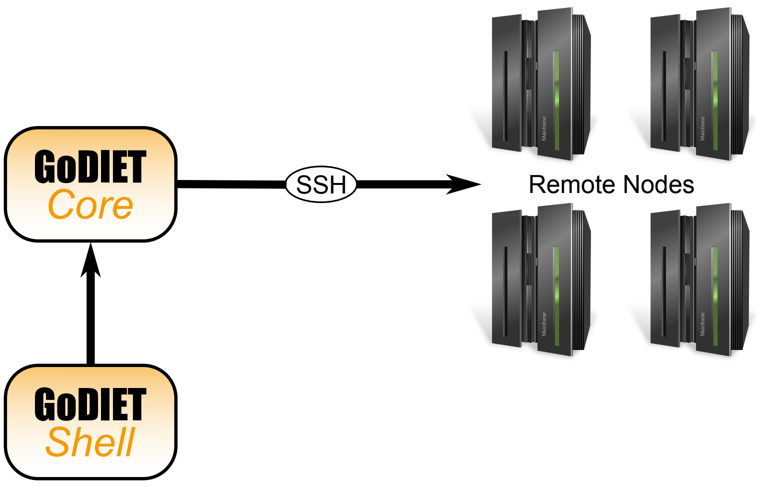


FIGURE 1.1 – Design principle of GoDIET.

Extract the archive just and launch run.bat or run.sh script. You need to load your physical platform on which will GoDIET will running. This platform must be describe in a XML file based on *Platform.xsd* grammar. You can find an simple file in the example directory

1.1.2 GoDIET setup

Avnt de pouvoir utiliser GoDIET

Configuration

Infrastructure

The user also defines all machines available for the deployment, disk scratch space available at each site for storage of configuration files, and which machines share the same disk to avoid unecessary copies.

Diet platform

The user of GoDIET could describes the desired deployment in an XML file including all needed external services (*e.g.*, omniNames and LogService) ; the desired hierarchical organization of agents and servers is expressed directly using the hierarchical organization of XML.

1.1.3 Godiet shell

Completion historique coloration syntaxique le fichier de configuration est chargé automatiquement depuis le répertoire home/.godiet.

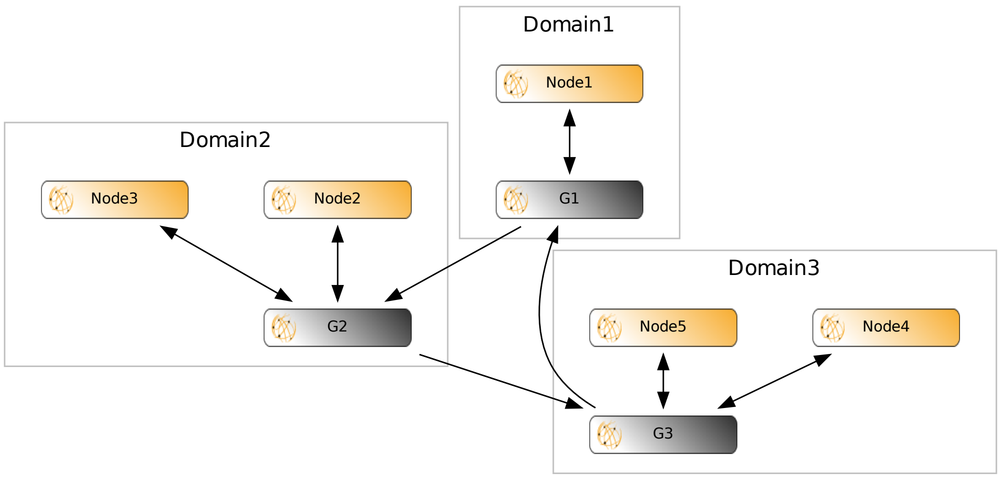


FIGURE 1.2 – The help command.

Help command

SSH command

Load command

calcule et création automatique des Forwarders

Start & Stop commands

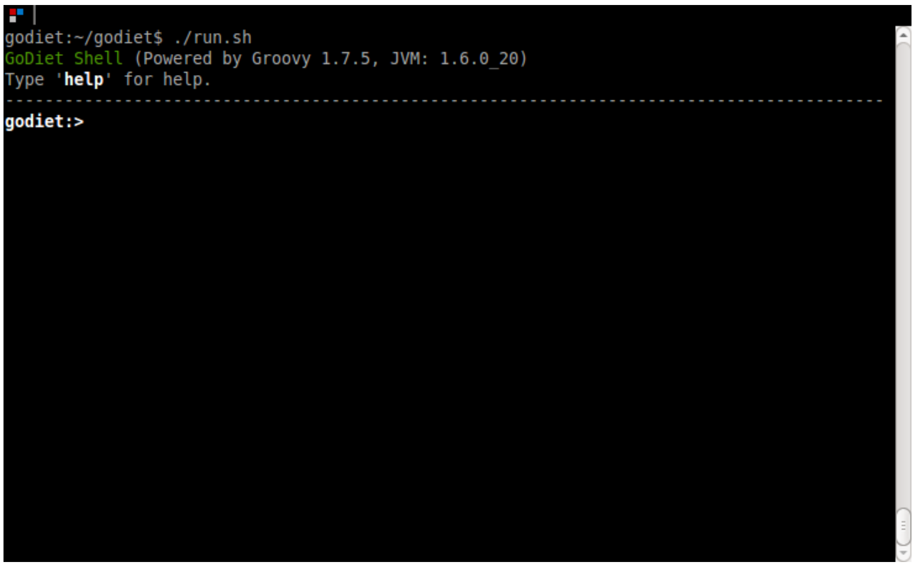
Status command

Status command options :

- **ma** : Display the masters agents status
- **la** : Display the locals agents status
- **seds** : Display the servers daemons status
- **all** : Display all diet softwares status

La commande status affiche sous forme de tableau l'état des éléments gérés pas GoDiet. Un exemple d'execution est affiché sur la figure 1.5. De gauche à droite les informations affichés sont :

- Le nom du logiciel comme décrit dans le fichier de description d'infrastructure
- Son état et depuis quand il s'y trouve.
- La machine sur laquelle le logiciel va être executé ou est actuellement executé (selon son état)
- Son



```
godiet:~/godiet$ ./run.sh
GoDiet Shell (Powered by Groovy 1.7.5, JVM: 1.6.0_20)
Type 'help' for help.
--------------------------------------------------------------------------------
godiet:>
```

FIGURE 1.3 – GoDiet shell startup.

Draw command

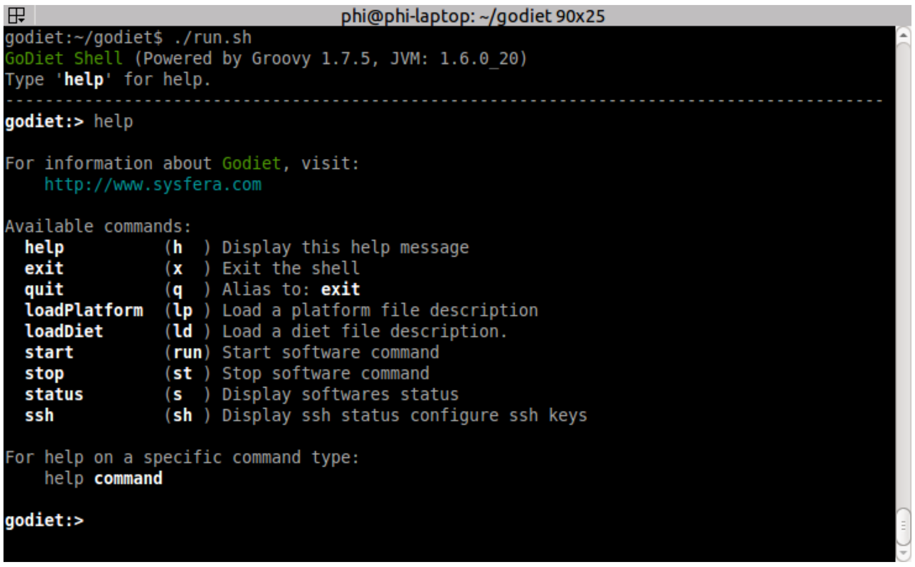
An example input XML file is shown in Figure 1.6 ; see [?] for a full explanation of all entries in the XML. You can also have a look at the fully commented XML example file provided in the GoDiet distribution under examples/commented.xml, each option is explained. To launch GoDiet for the simple example XML file provided in the GoDiet distribution under examples/example1.xml, run :

```
~ > java -jar GoDIET-x.x.x.jar example1.xml
XmlScanner constructor
Parsing xml file: example1.xml
GoDIET>
```

GoDiet reads the XML file and then enters an interactive console mode. In this mode you have a number of options :

We will now launch this example ; note that this example is intentionally very simple with all components running locally to provide initial familiarity with the GoDiet run procedure. Deployment with GoDiet is especially useful when launching components on multiple remote machines.

```
GoDIET> launch
* Launching DIET platform at Wed Jul 13 09:57:03 CEST 2005

Local scratch directory ready:
        /home/hdail/tmp/scratch_godiet

** Launching element OmniNames on localHost
Writing config file omniORB4.cfg
Staging file omniORB4.cfg to localDisk
```



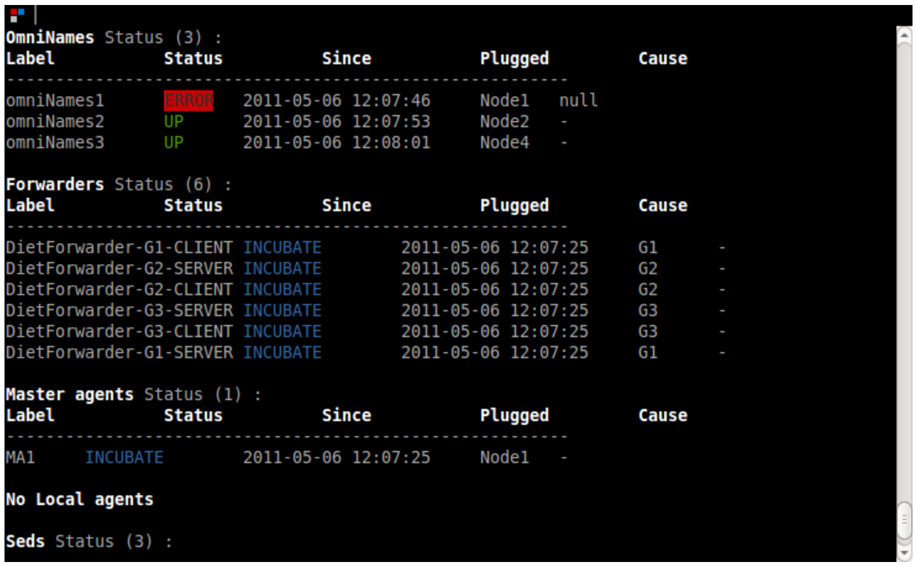
```
phi@phi-laptop: ~/godiet 90x25
godiet:~/godiet$ ./run.sh
GoDiet Shell (Powered by Groovy 1.7.5, JVM: 1.6.0_20)
Type 'help' for help.
------------------------------------------------------------------------------------------
godiet:> help

For information about Godiet, visit:
    http://www.sysfera.com

Available commands:
  help          (h  ) Display this help message
  exit          (x  ) Exit the shell
  quit          (q  ) Alias to: exit
  loadPlatform  (lp ) Load a platform file description
  loadDiet      (ld ) Load a diet file description.
  start         (run) Start software command
  stop          (st ) Stop software command
  status        (s  ) Display softwares status
  ssh           (sh ) Display ssh status configure ssh keys

For help on a specific command type:
    help command

godiet:>
```

FIGURE 1.4 – The help command.

```
Executing element OmniNames on resource localHost
Waiting for 3 seconds after service launch

** Launching element MA_0 on localHost
Writing config file MA_0.cfg
Staging file MA_0.cfg to localDisk
Executing element MA_0 on resource localHost
Waiting for 2 seconds after launch without log service feedback

** Launching element LA_0 on localHost
Writing config file LA_0.cfg
Staging file LA_0.cfg to localDisk
Executing element LA_0 on resource localHost
Waiting for 2 seconds after launch without log service feedback

** Launching element SeD_0 on localHost
Writing config file SeD_0.cfg
Staging file SeD_0.cfg to localDisk
Executing element SeD_0 on resource localHost
Waiting for 2 seconds after launch without log service feedback
* DIET launch done at Wed Jul 13 09:57:14 CEST 2005 [time= 11.0 sec]
```

The `status` command will print out the run-time status of all launched components. The `LaunchState` reports whether GoDiet observed any errors during the launch itself. When the user requests the launch of LogService in the input XML file, GoDiet can connect to the LogService after launching it to obtain the state of launched components ; when available, this state is reported in the `LogState` column.

```
GoDIET> status
```



```
OmniNames Status (3) :
Label           Status          Since           Plugged         Cause
------------------------------------------------------------
omniNames1      ERROR   2011-05-06 12:07:46     Node1   null
omniNames2      UP      2011-05-06 12:07:53     Node2   -
omniNames3      UP      2011-05-06 12:08:01     Node4   -

Forwarders Status (6) :
Label           Status          Since           Plugged         Cause
------------------------------------------------------------
DietForwarder-G1-CLIENT INCUBATE        2011-05-06 12:07:25     G1      -
DietForwarder-G2-SERVER INCUBATE        2011-05-06 12:07:25     G2      -
DietForwarder-G2-CLIENT INCUBATE        2011-05-06 12:07:25     G2      -
DietForwarder-G3-SERVER INCUBATE        2011-05-06 12:07:25     G3      -
DietForwarder-G3-CLIENT INCUBATE        2011-05-06 12:07:25     G3      -
DietForwarder-G1-SERVER INCUBATE        2011-05-06 12:07:25     G1      -

Master agents Status (1) :
Label           Status          Since           Plugged         Cause
------------------------------------------------------------
MA1     INCUBATE        2011-05-06 12:07:25     Node1   -

No Local agents

Seds Status (3) :
```

FIGURE 1.5 – The 'status all' command.

```
Status   Element   LaunchState   LogState   Resource    PID
         OmniNames running       none       localHost   1232
         MA_0      running       none       localHost   1262
         LA_0      running       none       localHost   1296
         SeD_0     running       none       localHost   1329
```

Finally, when you are done with your DIET deployment you should always run `stop`. To clean-up each element, GoDIET runs a `kill` operation on the appropriate host using the stored PID of that element.

```
GoDIET> stop

* Stopping DIET platform at Wed Jul 13 10:05:42 CEST 2005
Trying to stop element SeD_0
Trying to stop element LA_0
Trying to stop element MA_0
Trying to stop element OmniNames

* DIET platform stopped at Wed Jul 13 10:05:43 CEST 2005[time= 0.0 sec]
* Exiting GoDIET. Bye.
```

One of the main problem when writing a GoDIET XML input file is to be compliant with the dtd. A good tool to validate a GoDIET file before using GoDIET is **xmllint**. This tool exist on most platforms and with the following command :

```
$ xmllint your_xml_file --dtdvalid path_to_GoDIET.dtd -noout
```

you will see the different lines where there is problem and a clear description of why your XML file is not compliant.

```
<?xml version="1.0" standalone="no"?>
<!DOCTYPE diet_configuration SYSTEM "../GoDIET.dtd">
<diet_configuration>
  <goDiet debug="2" saveStdOut="yes" saveStdErr="yes" useUniqueDirs="no" log="no"/>
  <resources>
    <scratch dir="/tmp/GoDIET_scratch"/>
    <storage label="disk-1">
        <scratch dir="/tmp/run_scratch"/>
        <scp server="res1" login="doe"/>
    </storage>
    <storage label="disk-2">
        <scratch dir="/tmp/run_scratch"/>
        <scp server="res2" login="foo"/>
    </storage>
    <storage label="disk-3">
        <scratch dir="/tmp/run_scratch"/>
        <scp server="res3" login="bar"/>
    </storage>
    <compute label="res1" disk="disk-1">
        <ssh server="res1" login="doe"/>
        <env>
              <var name="PATH" value=""/>
              <var name="LD_LIBRARY_PATH" value=""/>
        </env>
    </compute>
    <compute label="res2" disk="disk-2">
        <ssh server="res2" login="foo"/>
               <env>
                       <var name="PATH" value=""/>
                       <var name="LD_LIBRARY_PATH" value=""/>
               </env>
    </compute>
        <cluster label="res3" disk="disk-3" login="bar">
      <env>
        <var name="PATH" value=""/>
        <var name="LD_LIBRARY_PATH" value=""/>
      </env>
      <node label="res3_host1">
        <ssh server="host1.res3.fr"/>
        <end_point contact="192.5.80.103"/>
      </node>
      <node label="res3_host2">
        <ssh server="host2.res3.fr"/>
      </node>
    </cluster>
  </resources>
  <diet_services>
          <omni_names contact="res1_IP" port="2121">
        <config server="res1" remote_binary="omniNames"/>
    </omni_names>
  </diet_services>
  <diet_hierarchy>
    <master_agent label="MA">
        <config server="res1" remote_binary="dietAgent"/>
           <cfg_options>
             <option name="traceLevel" value="1"/>
           </cfg_options>
            <SeD label="SeD1">
                <config server="res2" remote_binary="server_dyn_add_rem"/>
           <cfg_options>
             <option name="traceLevel" value="1"/>
           </cfg_options>
            </SeD>
        <SeD label="SeD2">
            <config server="res3_host1" remote_binary="server_dyn_add_rem"/>
           <cfg_options>
             <option name="traceLevel" value="30"/>
           </cfg_options>
           <parameters string="T"/>
        </SeD>
        <SeD label="SeD3">
            <config server="res3_host2" remote_binary="server_dyn_add_rem"/>
           <cfg_options>
             <option name="traceLevel" value="1"/>
           </cfg_options>
        </SeD>
    </master_agent>
  </diet_hierarchy>
</diet_configuration>
```

.

FIGURE 1.6 – Example XML input file for GoDiet.