

VERSION 2.6
DATE Mai 2011
Copyright SysFera

Table des matières

1	Deploying a DIET platform	5
1.1	GoDIET	5
1.1.1	Installing GoDIET	5
1.1.2	Quickstart	6
1.1.3	GoDIET setup	6
1.1.4	Godiet shell	8

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.

Using **GoDIET**, a Java-based tool for automatic DIET deployment that manages the creation of configuration file, the staging of files, the launch of elements, the monitoring and reporting upon successful launch, and the cleanup process 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 expected, as there are many complexities to manage. Learn how to use GoDIET— it will save you time !

1.1 GoDIET

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

- complete command-line interface ;
- distributed command execution via SSH ;
- real-time monitoring of applications' states ;
- handling of complex physical infrastructure, with firewalls and multiple local- and wide-area networks.

1.1.1 Installing GoDIET

The following operating systems are known to support GoDiet :

- Linux : the most recent distributions are likely to work ;
- Mac OS X 10.4 or later.

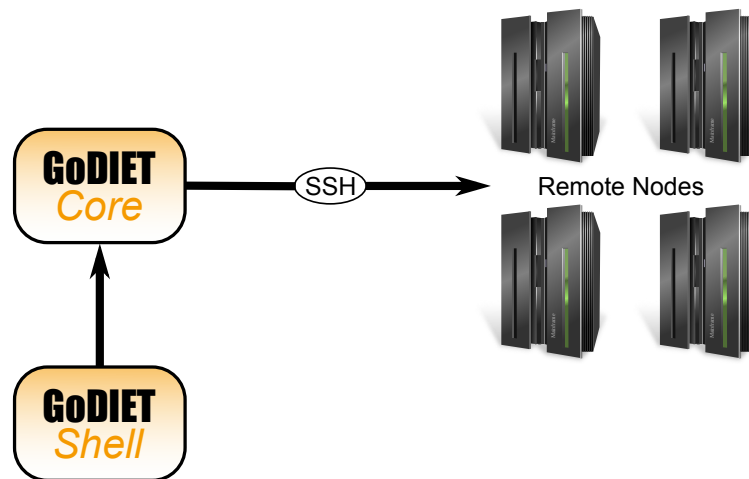


FIGURE 1.1 – Design principle of GODIET.

You need to have Sun Java 6 or OpenJDK6 installed. Download GODIET on the project's website¹. Check that the `run.sh` script works : it should display something similar to figure 1.3.

1.1.2 Quickstart

The four steps for people in a hurry :

- create your GODIET configuration file in `${HOME}/.godiet/configuration.xml` 1.1.3. It contains information for remote connection ;
- create your infrastructure description file 1.1.3. It describes your compute nodes, gateways and storage nodes ;
- create your DIET platform description file 1.1.3. It describes all the DIET elements (agents and SeDs) that you want to deploy and manage ;
- run the GODIET shell ??.

And that's it !

1.1.3 GODIET setup

Before using GODIET, you need to create three files : one to describe GODIET's configuration, one to describe your infrastructure and one that contains your DIET platform's description. These files use an XML-format and conform to the `Configuration.xsd`, `Infrastructure.xsd` and `Diet.xsd` grammar files (provide with GODIET), respectively. Sample files are provided in the **examples** directory.

Configuration.xml

This file aggregates information about the local node from where GODIET was launched. It also contains information about user authentication. By default, GODIET looks in the `${HOME}/.godiet/config` directory.

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

The mains elements are :

- **localNode** : the name of the node on which GODIET was launched. This name must be present in the infrastructure description file ;
- **localScratch** : the working directory where GODIET stores its temporary files ;
- **keys** : the paths of your private ssh keys, which are loaded at GODIET's startup. You can give the public key's path, too. GODIET tries to load a file with the same name as the private key and ending with **.pub**. See the **ssh initkeys** command [1.1.4](#) to initialize passwords if your keys are encrypted (i.e. need a passphrase).

General layout of the configuration description (some parts are omitted) :

```
<configuration schema="Configuration.xsd">
<goDietConfiguration localNode="local">
<localScratch dir="/tmp/scratch_godiet" />
  <user>
    <ssh>
      <key path="/home/.ssh/id_dsa"/>
      <key path="/home/.ssh/admin_cluster2"/>
    </ssh>
  </user>
</goDietConfiguration>
```

Infrastructure.xml

GODIET needs to have the description of the infrastructure on which DIET will be running.

You can find the full list of options in the **Infrastructure.xsd** grammar file. You can also look in the **examples** directory. The most important fields are :

- **Domain** : aggregates a set of infrastructure elements (nodes, gateways, storage) which are able to communicate with DIET's exchange protocol (i.e., CORBA). Typically, elements separated by a firewall and/or a router must be described in separate domains ;
- **Storage** : describes a remote disk access ;
- **Node** : describe a computing node where agents or SeDs will be running ;
- **Gateway** : describes an access point for a domain.
- **Link** : defines a directional link through which two gateways can communicate, using SSH.

General layout of the infrastructure description (details are omitted) :

```
<infrastructure schema="Infrastructure.xsd">
  <domain id="idDomain1">
    ..
    <storage id="idStorage">..</storage>
    <node id="idNode" storageRefId="idStorage">..</node>
    <gateway id="idGateway1">..</gateway>
```

```

</domain id="idDomain2">
..
  <gateway id="idGateway2"/>
..
</domain>
<link from="idGateway1" to="idGateway2"/>
</infrastructure>

```

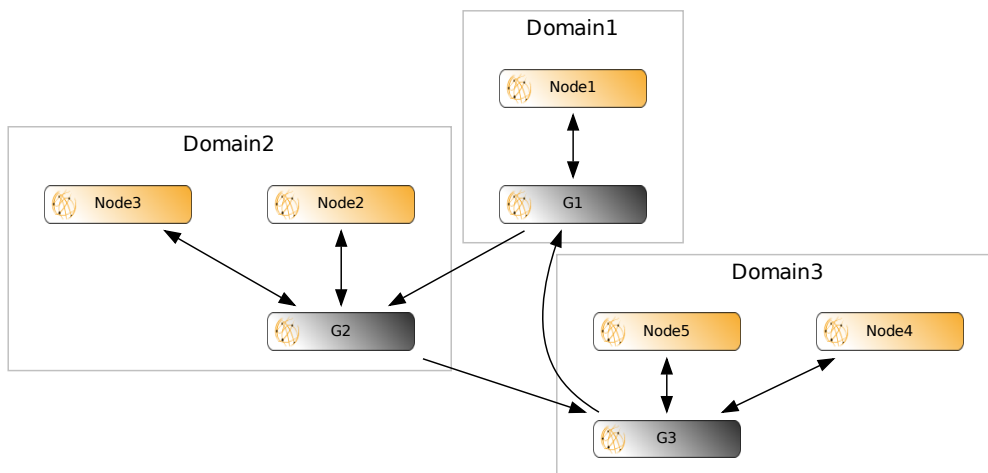


FIGURE 1.2 – Example of the representation of a three-domain infrastructure.

DIET-platform.xml

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

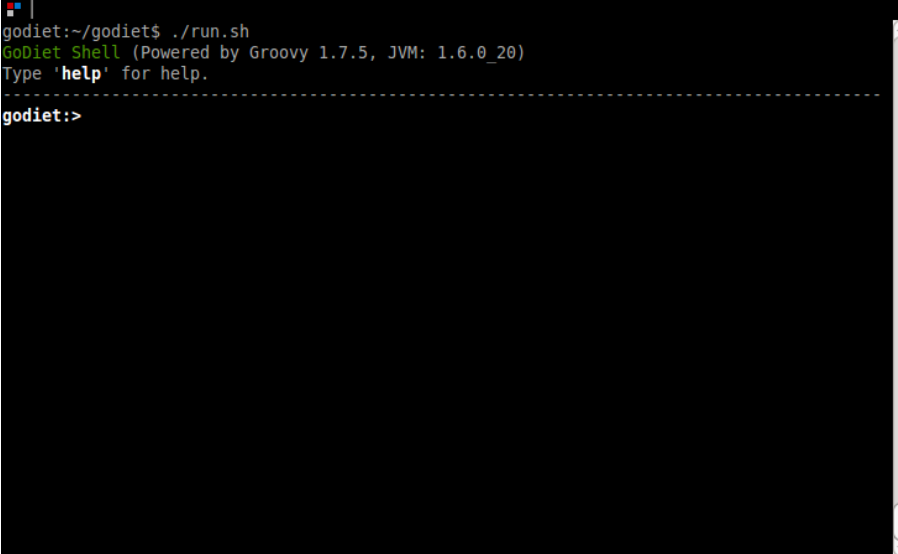
1.1.4 Godiet shell

GoDIET shell is the interface to manage your DIET platform. It includes facilities features like syntax highlighting, command completions and history commands. After executing run script you will have a prompt like on the figure [1.3](#)

Help command

SSH command

- **initpasswords** : Allow you to initialize ssh key's passphrase loaded from configuration file ;



```
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.

- **addkey** : Register a new key ;
- **modifykey n** : Modify a key that already registred ;
- **status** : Display the keys status. Could be PASSWORDNOTSET,PRIVATEKEYERROR,PUBKEYERRR or LOADED.

Load commands

loadInfrastructure command to load an infrastructure description [1.1.3](#) **loadDiet** command se to load an [1.1.3](#)

calcule et création automatique des Forwarders

Start & Stop commands

start command Start command options :

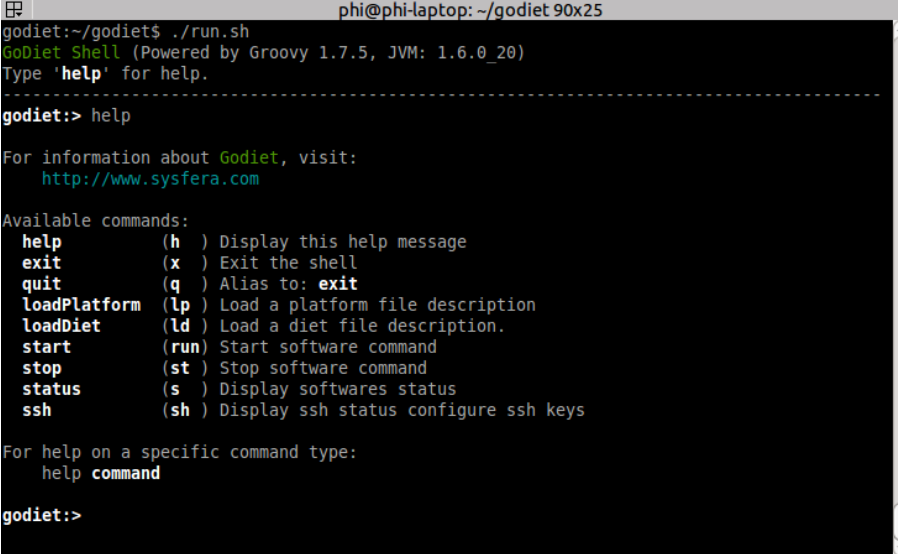
- **software** name : Start one software given his name.
- **services** : Start all services
- **agents** : Start all agents
- **seeds** : Start all seeds
- **all** : Start all softwares components

stop command have same parameters.

Status command

Status command options :

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



```

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.

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

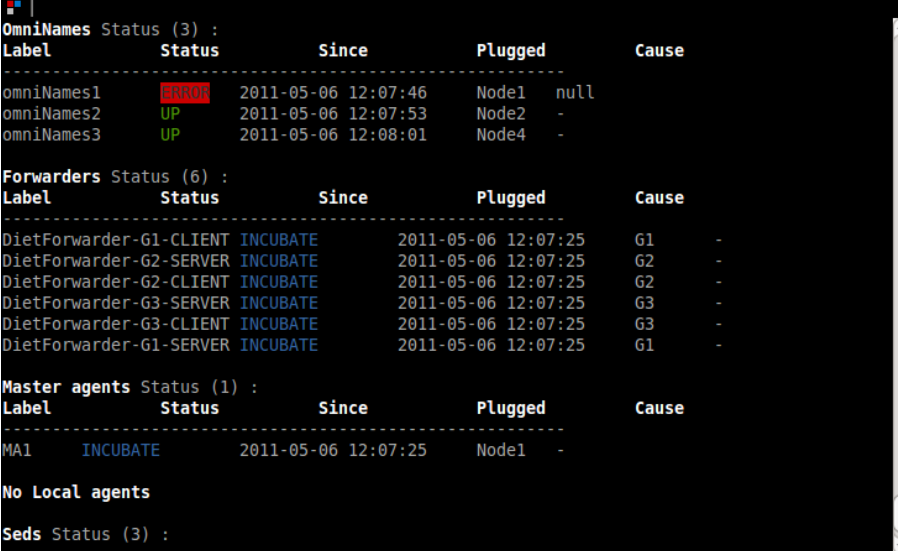
- Le nom du logiciel comme décrit dans le fichier de description d'infrastructure.
- Son état et depuis quand il s'y trouve.
- La machine sur laquelle le logiciel va être executé ou est actuellement executé (selon son état).
- Le message d'information dans le cas où la ressource est dans l'état erreur.

Les différents états possibles sont

- **Incubate** : The component is correctly loaded in GoDIET.
- **Ready** : The configurations file are created in the local scratch directory and component is ready to start.
- **Up** : The component is Up. His state is periodically checked.
- **Down** : The component is Down. Typically if GoDIET user call stop on this component.
- **Error** : Could appears if there is an error in the description, if the

Debugging & Gestion des erreurs

parler des fichiers de log et des messages d'erreurs.



```

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.