1.1 USED TECHNOLOGY

Towards greater feasibility and integration with external processes to the plug-in developed in this project the Java programming language was chosen to provide input in the absence of the Nagios system restrictions as well as the large amount of documentation available and the wide range of libraries this language. For the perfect execution of the plug-in it was also necessary to use a script that runs the Java program returning to the Nagios results of this execution.

The script interprets the result of the Java program that is executed with the passage of some parameters Nagios system as shown in Figure 14. Return the scan data to the plugin. This script can observe the path where you will find the program that checks the hosts what is termed "serviço_verificador.jar". It is responsible for checking the status of each host that is registered in the verification service plug-in. For alertador service, the interpreter should run "serviço_alertador.jar" program, and the rest of the script is exactly the same.

```
$ output = "java -jar /usr/local/Nagios/libexec/servico_verificador.jar $ 1
$ 2 $ 3 $ 4 $ 5 $ 6"
# $ Output = "java -jar /usr/local/Nagios/libexec/servico_alertador.jar $ 1
$ 2 $ 3 $ 4 $ 5 $ 6"
java_resultado = $?
echo $ java_resultado
exit $ java_resultado
```

Figure 1. Script verification service interpreter .

1.1.1 Settings Plug-in

The great diversity of ways to configure Nagios system was necessary to create parameters that indicate which places are the Nagios configuration files. Only in this way the plug-in will be able to interpret and run perfectly together with Nagios.

1.1.1.1 Registration Service

The plug-in allows to be created a new service in Nagios. The service name must be unique. As a first step should be to inform the ways:

- **Services of Nagios:** folder path where you will find the services already operating in Nagios.
- **H OSTs of Nagios:** folder path where is the current hosts of Nagios.
- **Nagios:** Nagios path of the folder that you want to deploy the plug-in.
- **Nagios Plugins:** folder path where is the plug-ins Nagios.
- **Logs plugin:** path where the plug-in will store the generated reports.

This configuration data is saved in a text file named "configuracoesPluginVerde.txt" as shown in Figure 15. In this way every time you need to create a new green service, will not be required for these settings again.

```
# Plug-in Nagios for Waste Management of Electricity
################################################## #######################
<CaminhoServico> / etc / NagiosQL / services / <caminhoServico>
<CaminhoHosts> / etc / NagiosQL / hosts / <caminhoHosts>
<CaminhoNagios> / etc / NagiosQL / <caminhoNagios>
<CaminhoNagiosLibexec>   /   usr   /   local   /   nagios   /   libexec   /
<caminhoNagiosLibexec>
<CaminhoNagiosLog> / usr / local / nagios / var / <caminhoNagiosLog>
################################################## #######################
```

Figure 2. File registration settings .

The next steps to complete the registration of new services are as follows:

1. **Name for the verification service:** enter the name for the new service checks.
2. **Host registration (s) to be monitored:** the plug-in shows which are the host (s) existing and available, then the administrator must be informed which or what the hosts (separated by commas) that the service should do checks .
3. **Group registration (s) of host (s) to be monitored:** the plug-in tells you which group (s) hosts are available, and should be introduced hosts groups (comma separated), which must be verified.
4. **Powers of equipment:** the plug-in tells you the name of each host to be added the value of the power of each device.
5. **Service verification period:** the plugin informs all periods available and the user must enter a period that this service should perform the checks.
6. **Alertador service name:** enter the name for the new alert service.
7. **Host registration (s) to be warned:** the plug-in shows which are the host (s) existing and available and then must be informed which or what the hosts (separated by commas) that the service should carry warnings.
8. **Group registration (s) of host (s) to be warned:** the plug-in tells you which group (s) hosts are available, and should be introduced hosts groups (comma separated) which should carry warnings .
9. **Alerts period:** the plug-in list what periods are available in sequence and insert the name of a period for performing the posting of e-mail.
10. **Alerted Service name:** the plug-in lists all the services available in Nagios, and should

be introduced a verification service name to generate alerts reports.

11. **Contacts or warnings contact group:** the plug-in tells you which contacts and contact groups that are previously registered in Nagios and sequence should be informed which of these (separated by commas) will be alerted.

12. **Alert report:** must be told what will be the time in h (hours) or d (days) to be included in the report, counting of the report generation.

13. **KWH rate:** for each reporting period and the registration of a rate value to generate approximate values of unnecessary spending is necessary.

14. **MySQL server:** must be told the name or address of the MySQL server that will store the information for each active verification plug-in.

15. **Database:** must inform the database name that exists or to be created to store the plug-in information.

16. **Username and Password** for the MySQL connection should still inform the user name and password with permissions to create and edit informed database.

17. **SMTP server:** for sending alerts are needed to be informed the name or address of the SMTP server.

18. **Authentication:** If the server does not require authentication must be informed "false", otherwise must leave to "true".

19. **SMTP port: it** must be told the SMTP port number.

20. **User and Password** to connect to the mail server, are still required to inform the user name and its password.

1.1.1.2 Structure

The plug-in in its initial phase has only three programs, an overview of the structure of the plug-in conjunction with Nagios is shown in Figure 16. The **"PluginNagiosVerde.jar"** that contains all the necessary code to automatically generate all settings of a new checker service and a alertador service in Nagios, the **"VerificadorNagiosVerde.jar"** which is the program that checks the hosts for active equipment, and the **"AlertadorNagiosVerde.jar"** is the program that generates reports quantification of the checks made by the verification service.

Figure 3. Plug-in structure.

After creating a new service some Nagios files are changed, such as the file **"commands.cfg"** and the file of each host that is part of the new service. The hosts files a new variable called "_potencia" which is related to electric power of each host, such as "host_cisco.cfg" and can be seen in Figure 17 are still generated other new configuration files plug-in are: **"chama_alertador" "chama_verificador"** “ **configuracoes_verificador** ”, “ **alertador.cfg** ”, “ **verificador.cfg** ”, “ **serviço_alertador.jar** ” e **"Serviço_verificador.jar".**

```
define host {
              host_name              CISCO
              alias              CISCO
              address              192.168.1.247
              max_check_attempts              5
              check_period              24 HOURS
              contacts              administrator
              notification_interval              60
              notification_period              24 HOURS
              _potencia              255
              register              1
}
```

Figure 4. Host_cisco.cfg file .

File "commands.cfg"

In the Nagios command file as shown in Figure 18, are added the name and the command to be executed when called by Nagios. The line "command_name" is the name of the command used by Nagios, and the line "command_line" is what command will be executed, the remaining parameters are referred to the external program called "serviço_plugin_verde.jar" and are explained below:

- **$ USER1 $ / chama_verificador:** The default macro is used "USER1" that tells the plug-ins folder of Nagios followed by the file name "chama_verificador" which is the program interpreter that makes checks and can be seen in Figure 14.
- **$ HostAddress $:** macro that tells the host address to be checked.
- **$ HOSTNAME $:** macro that tells the host name that will be checked.
- **$ SERVICEDISPLAYNAME $:** macro that tells the name of the service being performed.
- **$ USER1 $ /:** macro that tells the plug-ins folder of Nagios followed bar.
- **$ _HOSTPOTENCIA $:** Variable created by the plug-in that captures the power of the host.
- **$ HOSTGROUPNAMES $:** macro that tells the name (s) group (s) that the host belongs.
- **$ USER1 $ / `chama_alertador`:** The default macro is used "USER1" that tells the plug-ins folder of Nagios followed by the file name "chama_alertador" which is the program interpreter that sends alerts.
- **$ _SERVICEEMAIL $:** Variable created by the plug-in that captures the (s) address (s) email the alertador service.
- **$ _service TEMPORELAT $:** variable created by the plug-in that captures the value of time to generate alert reports for sending e-mail.
- **$ _SERVICEVALORKWH $:** Variable created by the plug-in that corresponds to the value of KWH rate, which is captured from the running service.
- **$ _service SERVICOALERTA $:** variable created by the plug-in that captures the name check service to be generated the alert reports.

Variables defined in the services are available in Nagios as macros, and to use these macros you use the symbol $ (dollar) before and after the variable name you want to get the result.

```
################## HOME check_verificador ######################

define command {

              command_name                check_verificador

              command_line                $ USER1 $ /
chama_verificador HostAddress $ $ $ $ $ HOSTNAME $
SERVICEDISPLAYNAME $ USER1 $ /

_HOSTPOTENCIA $ $ $ $ HOSTGROUPNAMES

              register                                                 1

}
```

```
################### Check_verificador FINAL ######################


###################### HOME check_alertador ####################

define command {

              command_name
              check_alertador

              command_line
              $ USER1 $ / chama_alertador $ USER1 $ / $ $ $
_SERVICEEMAIL _SERVICETEMPORELAT $ $ $ $ _SERVICEVALORKWH
_SERVICESERVICOALERTA $ $$ HOSTGROUPNAMES

              register                                          1

}

###################### ###################### Check_alertador FINAL
```

Figure 5. Commands.cfg file .

File "hosts.cfg"

Each host has been added for verification received a new value of electric power as shown in Figure 17, this value has been added in the "_potencia" and will be used as a parameter for quantifying reports to be sent in alerts.

File "chama_verificador" and "chama_alertador"

These are the files that are generated by the plug-in when the service checks and alerts services are created. They are responsible for sending the Nagios variables and macros to the Java program and the return of these check information for the plug-in.

File "configuracoes_verificador"

In this file are contained all checks service settings as shown in Figure 19, and also used by alertador service. The data that comprise this file are:

caminhoNagiosLog: that is the folder where the plug-in will store the reports that were generated by the alert service.

caminhoNagios: root folder that contains the Nagios configuration.

- MySQL database information that stores the information of checks and information that will be used by alertador service to generate measurement reports.

  serverName: name or the server address.
  m ydatabase: name of the database.
  username: username base that has permission to create tables.
  password: password of the user base.

- Information for sending alerts via e-mail.

  smtp: SMTP server address.
  Authentication: Authentication Type.
  door: port number.
  SMTP_AUTH_USER: user name
  SMTP_AUTH_PWD: User Password

```
# Plug-in Nagios for electricity waste management

### (Checker) ############################################ ######

<CaminhoNagiosLog> / usr / local / nagios / var / logverde /
<caminhoNagiosLog>

<CaminhoNagios> / etc / NagiosQL / <caminhoNagios>

<ServerName> localhost <serverName>

<Mydatabase> base_verde <mydatabase>

<Username> usuariosql <username>

<Password> senhasql <password>

<SMTP> smtp.email.com.br <smtp>

<Authentication> true <Authentication>

<Port> 587 <port>

<SMTP_AUTH_USER> nome@email.com.br <SMTP_AUTH_USER>

<SMTP_AUTH_PWD> Password <SMTP_AUTH_PWD>

################################################## ################

### (((Tester))) ###
```

Figure 6. Configuracoes_verificador file .

File "verificador.cfg"

This file has the same name check service, and contains the data created by the Green plug-in. The information necessary for the operation of this service can be seen in Figure 20 and are:

hostgroup_name: names of groups of hosts to be scanned.
service_description: Check service name.
max_check_attempts: number of verification attempts
check_interval: time between scans, this value is set as the plug-in sixty (60 minutes) and should not be changed, since the calculations spending as shown in Equation 1 is performed based on one (1) hour .
retry_interval: time interval between attempts.

```
define service {
```

```
hostgroup_name group 1, group 2, group 3
service_description checker
check_command check_verificador
max_check_attempts 1
check_interval 60
retry_interval 1
check_period PeriodoVerde
register 1
}
```

Figure 7. Verificador.cfg file .

File "alertador.cfg"

This file is given the same name alertador service and contains the data created by the plug-in at registration of a new alert service and its structure can be seen in Figure 21.

```
define service {
host_name host1, host2, host3
service_description alertadorMeioDia
check_command check_alertador
max_check_attempts 5
check_interval 60
retry_interval 1
check_period enviar_email
        _email ti@nagios.com.br
_servicoalerta checker
        _temporelat 3h
_valorkwh 0067
register 1
}
```

Figure 8. Alertador.cfg file .

File "servico_verificador.jar" and "servico_alertador.jar"

The veririficadores programs and alertadores are the programs responsible for carrying out the checks and alerts the plug-in. The formation of their names is always initiated by the constant "servico_". In turn the verification service is concatenated with the name given to the verification service of the hosts, while the alerts service is concatenated with the name given to the service of sending alerts the plug-in. These programs are copies of the programs "VerificadorNagiosVerde .jar" and ".jar AlertadorNagiosVerde" that are available in plug-in package.

The verifier program initially performs a communication test through the search method using the *ping* command. If this test results in a negative state the plug-in still performs one second search method that uses the Java API command isReachable InetAddress. If one of these checks is a positive result, ie, the host is active plug-in records data directly in the database, otherwise nothing is done and returns a result data to the Nagios system.

The verifier service returns to Nagios system any one of three values as shown in Table 2, and the return value "0" tells you that the host is turned off and therefore are not consuming electricity, the value "1" informs the host It is consuming power unnecessarily, and the value "3" tells you that the plug-in does not have all the variables correctly Nagios.

Table 1. Return States checker service.

Return	Service	Host
0	NOT FOUND	OFF
1	FOUND	CONSUMING ENERGY
3	CAUTION	VARIABLE MISSING

The alertador service returns to Nagios the following values as shown in Table 3, the value "0" is achieved if the alert is successfully generated, and or if an error occurred while building this report the return value must be "1 "that inform the Nagios the error that caused the failure. The return value "3" tells you that there are all variables required to perform the service properly.

Table 2. Return states alertador service.

Return	Service	Host
0	OK	ALERT SENT
1	ERROR	ERROR GENERATING / SENDING REPORT
3	CAUTION	VARIABLE MISSING

1.1.2 Data base

The plug-in registers when a new verification service will create if necessary a new table in the database that will serve to record the verification data of this service and checks or other services. The database table where the data will be recorded can be seen in Table 4. The alerts service is used the data contained in this table to generate alerts reports.

Table 3. Table scans of records.

v_cod_ocorrencia	**Code number of table records.**
v_nome_servico_ocorrencia	Check service name.
v_host_ocorrencia	Address of the scanned host.
v_hostname_ocorrencia	Hostname checked.
v_data_ocorrencia	Data verification.
v_hora_inicio_ocorrencia	Date / verification start time.
v_hora_fim_ocorrencia	Date / check end of the hour.
v_potencia_ocorrencia	Power value of the scanned host.
v_grupo_ocorrencia	Host group name checked.