1 JimysProbes

1.1 JimysProbes folder

+ jimysProbes
    - startProbes.sh
    - build.xml
    + dist
        - BootJimys.jar
        - yyyy.jar
    +conf
        - config.xml
        - logging.properties
        - mlet.xml
        - mletHardware.xml
        - mletJonas.xml
        - mletJonasHardware.xml
        - monitor.properties
        - monitorHardware.properties
        - monitorJonas.properties
        - monitorJonasHardware.properties
        + probes
            - joramProbeConfig.xml
            - probeconfig-414.xml
    + lib
        - libs to compile
    + libMini
        - mx4j.jar
        - mx4j-jmx.jar
        - mx4j-remote.jar
        - mx4j-tools.jar
        - xercesImpl.jar
        - xml-apis.jar

1.1.1 Folders

- **dist**: contains the libraries of the jimys probes. This folder normally contains only 2 jars.
- **conf** : contains the configuration files of the probes and JMX server
- **lib**: contains all the necessary libraries in order to compile the distribution jars from the sources.

- **libMini**: contains the minimum of libraries in order to be able to run the probes. These libraries are the mx4j libs and also the xml implementation. Attention: You must also have the lib “BootJimys.jar” in the **dist** directory!

1.1.2 Important files

- **startProbes.sh**: startup script to start the probe module. You can modify this script in order to take advantage of a different configuration (monitorXY.properties).
- **build.xml**: The ant script to build the binaries from the sources. The target “bootJar” produces the binary “BootJimys.jar”. The target “jar” produces the binary distribution “JimysProbes.jar” of the probes and pumps. This jar is designed to be on a central place (http server) and is downloaded dynamically by the probe module when this one starts. This has the advantage that you only need to replace the jar on a single place. (The url is in the monitorXY.properties which points to the mletXY.xml). The real url is coded in the mletXY.xml file which is also placed on the central place. Please consult section 1.4 for installing instructions.
- **Config.xml**: Configuration file of the mx4j server. You can modify the port which is opened for the probe module.
- **Logging.properties** : Config file of the logger.
- **mletXY.xm**l : These files define the mlet configuration. This file is intented to be located somewhere on a web server. But it is not mandatory. This file defines all the MBeans which are loaded by the JMX server. For example the “mletJonas.xml” defines only the probes (MBeans) which are able to monitor jonas parameters like log, jonas and joram. You can also use the “mletJonasHardware.xml” and decide only to start the jonas probes. But the specific libraries for the hardware probes will also downloaded by the probe module. In this file you define also where the libraries are located. Normally this is a url which points to a web server url but you can also let the url point to the file system. That means you don’t need to download the libraries dynamically from a webserver but you can also manually copy them somewhere on the file system. This reduces the network load on startup.
- **monitorXY.properties**: This file defines the mlet url and also which probes are deployed. Furthermore, it defines some parameters of each probe. Attention: this file is specified in the startup script “startProbes.sh”. This is one of the most important files for the probe module. For the different syntax, please look in section 1.4 for a detailed explication of this file.
- **joramProbeConfig**.xml : Configuration of the joram probe. The syntax of this file is similar to the one for the configuration of the J2EE probe done by the students last year
- **probeconfig-414.xml**: Configuration of the jonasjmx probe. The syntax of the configuration file is the same as the one for the configuration of the J2EE probe done by the students last year. You can even use a customised file you did for the other probe.

Remark : The “XY” in the name of the monitoringXY.properties and mletXY.xml files is an alias for the different example files in the conf directory. When XY is replaced by Jonas you use the jonas monitoring files. Using the Hardware files let you monitor the hardware architecture. Using the files where XY is replaced by nothing (mlet.xml), you have all the available probes. But be prepared to see some errors when using these files because some resources are not running.

1.2 Compiling the sources

There is an ant build script in the root of the directory.
The different targets are:

- **clean** : removes the compiled files and also the jar files
- **compile** : compiles the sources
- **jar** : produces the “JimysProbes.jar” in the dist folder.
- **bootJar** : produces the “BootJimys.jar” in the dist folder.
- **startProbes** : launches an MBeanServer and deploys all the probes defined in the monitoringXY.properties. Consult the “build.properties” file to change the path to the “monitoringXY.properties” file.
- **client** : launches a client thread which connects to a pump to retrieve the data. This application illustrates how to connect to the pumps and retrieve the data. You can extend this class in order to use the information collected by the pump. Consult the “build.properties” file to change the host and port to connect to.

1.3 Installing the probe module

1. Define a central machine where you want to put all the libraries and mlet config file. This machine needs to have a running http server. You can use a JOnAS server which does the monitoring for this purpose.
2. Put the mletXY.xml file on the HTTP server in a way that it is accessible in this way : http://host:9000/mletXY.xml
3. Put all the libraries which are referenced by the mlet file to the same place or to the url which is specified in the mlet file.
4. Copy the “jimysProbes” folder on all the machines which should be monitored. You can delete the “lib” folder if you want to save disk space. The only jars you absolutely need are “dist/bootJimys.jar” and the libs in the “libMini” folder. You can also delete the source and build directory.
5. Now the configuration of the probe module. If you used in step 2 for example “mletJonas.xml”, please use now also the monitorJonas.properties. But it is not mandatory.
6. Adapt the different urls in the monitorXY.properties in order to match your network settings (jonas port, protocol, …) and file system settings (the absolute path to the probe, jmx configuration files)
7. Adapt the “startProbes.sh” script in order to take the right monitorXY.properties file as argument.
8. Launch the “startupProbes.sh” script and you hopefully see no errors.

Remarks:

- The default port for the JMX server is 1999
- If you try to monitor a JOnAS server or another JMX server, this server needs to run before the probes are started.
- Please be sure to have defined the JAVA_HOME environment variable

1.4 Configuring the probe module

We will only explain the mletXY.xml file and monitorXY.properties file. The config.xml file can be modified according mx4j documentation. And the "probeconfig-414.xml", configuration of the jonas probe, is explained in the user guide of the J2EE probe done by the students last year.

mletXY.xml :

In this file you define all the MBeans which are deployed on the client machines (monitored machines). This file is intended to be accessible through a http url. But you can also copy it to the file system.

One entry defines one MBean.

```
<mlet     code="mbeanImplementationClassName"
          name="mbeanName"
          archive="libs,you,need,to,download"
          codebase="urlWhereTheLibsAreLocated">
```

code: The class you name here is instantiated by the MBean server.
name: The Object name of the MBean. With this name you are able to access the MBean.
archive: A list of jar files which are mandatory in order to instantiate the MBean. That means, if you instantiate a class in the MBean implementation specified by "**code**", this class must be in one of the archives defined here.
codebase: The url where the MBean server is able to download the jar files specified in the **archive** attribute.

Remarks:

- Do not bother that the file is not a valid xml file. (No root node is specified).
- You can define also mbeans here which are not used afterwards by the probe module. It is possible to have a large mlet.xml where all the possible mbeans are defined but in the monitor.properties you specify which one are used by the pump. Attention when you specify mbeans here, the mbean server will instantiate them. So it is possible that you receive some errors on startup for all the mbeans which produces an error. For example when you specify the CJDBC mbean, this probe will immediately try to establish a connection to the cjdbc server. But when no server is running you will see an error. This error will not cause any trouble for the other mbeans.

monitorXY.properties :

We will explain how you configure the JimysProbes and which possibilities you have.

Global configuration of the MBean server:

mlet.file : defines the url where the mlet file will be found. This has to be a valid URL. It con be on the file system or on a web server.
mlet.name : defines the object name of the mlet bean in the MBean server when it is deployed. This property may stay like this, no need to change.
mx4j.config : the url where the configuration file of the mx4j server is stored.

Remark : From now on we will not repeat the prefix "org.objectweb.jimys.monitoredSystem" for each property.

Pump configuration:

pump.name : the object name of the pump mbean in the mbeanserver
pump.compress : enable/disable the compress on the pump. Enabling the compression reduces the amount of data send over the network but increases the cpu load on both sides. When you us a client or the mediator of JimysJ2EE you must also enable the decompression to be able to see the data.
pump.sampletime : Specify the sample rate of the pump. This is the time between two messages sent from the pump to the client (mediator). This parameter is arbitrary, if not specified the sample rate is 5 seconds.

Admin configuration:

admin.name : the object name of the admin mbean in the mbeanserver.

List of the deployed probes:

probes : the list of the probes which will be registered in the pump. Do not confuse with the mbeans which will be deployed on the mbeanserver. These are specified in the mlet file. Each name of the list is separated by a ";". The names specified may be extended and customized. But when you specify a new name in this list, you also have to specify the probe specific proberties which will be: "org.objectweb.jimys.monitoredSystem.probe"+"NEW_NAME"!

Probe specific configuration:

Each probe has specific properties which must be defines. Some are mandatory, some are arbitrary.

Mandatory:

name : the object name of the probe in the mbeanserver. This is the same name specified for the mbean in the mlet file !
attributes : the list of attributes which are collected by the pump. This is a list of attribute names which are available through the mbean interface of the mbean probe. The different attributes are sperated by a ";".
sampleTime : this is the time the pump waits to collect the attributes for the next periode.

Arbitrary:

init : some probes must be initialized to work correctly. For example the jonasjmx probe must be initialized to receive the correct url of the mbeanserver of jonas. After initialization, the probe is able to connect to the other mbean server and collect the data. You have to look at the mbean interface of the probe to see how many attributes you must specify in the init property. The init method of the probe mbean interface has only "java.lang.String" parameters. How many of these parameters depends on the probe. Each parameter is separated by a ";"

1.5 Different probes

In general: All the possible attributes of the probes which may be monitored must be in the interface of the probe mbean. If you want to add a new attribute, you have to add the method of in the interface.

Cpu : This probe can monitor the load of the CPU. We chose not to use the CPU probe of the lewys probes because we had some problems to calculate the load. This probe only works on linux os. The load is calculated by taking two snapshots of the data in the proc file system (/proc/stat) for a given time between the snapshots. The load is calculated.

Network : This probe analyses the network traffic (Rx, Tx). The load is calculated the same way than the cpu. The probe is based on the lewys probes.

Memory : This probe calculates the memory usage of the system. The probe is based on the lewys probes.

Filesystem : This probe calculates the IO load of the file system. The probe is based on the lewys probes.

Cjdbc : This probe collects information of the cjdbc. Attention, this probe must be initialized. The probe is based on the lewys probes.

Apache : This probe collects the log of the apache web server. This probe needs to be initialised in order to be able to find the log file. Attention, this probe opens the file and

reads the whole file. If you have a large log file you may encounter performance problems.

Jonas : This probe is a wrapper around the j2ee probe developed by an other group of students. This probe needs to be initialised. Attention, this probe in jimys is depreciated because you should use the jonasjmx probe which is based on the genericjmx probe. DEPRECIATED

Log : This probe analyses the monolog output and can be used in parallel of the jonasjmx probe. This probe must be initialised. The probe registers a jmx listener in monolog which is running in a second mbeanserver.

Cartography : This probe analyses the hardware of the system. The data send by this probe stay constant (as long as you do not have any hot pluggable architecture). Therefore the sample rate should be very high. This probe is based on the lewys probes and works only on linux systems. (Preferable kernel 2.4.x)

Sunjvm : This probe analyses the sunjvm. This probe is depreciated because you should use the genericjmx probe. This probe permits to monitor the specific mbeans inside a jvm 1.5. DEPECIATED

Genericjmx : This probe is a generic probe which allows to probe attributes of mbeans deployed on other mbeanservers. In order to have multiple instances of the genericjmx probe, you must first have different configuration files (see next chapter). Furthermore you must specify different “name” in the mlet file but always the same “code”. That means, you can have 2 entries in the mlet file which have as code “.GenericJmx” but 2 different names. This probe must be initialised. The first parameter of the init method specifies the mbeanserver url of the mbeanserver to connect to. The second parameter is the location of the configuration file and the third parameter is the name of the probe which must be different from the other probes. A good example of using the genericjmx probe is the joram probe.

Jonasjmx : This probe collects the data from a jonas instance. This probe must be initialized. If you want to monitor multiple jonas instances on the same machine you must specify in the mlet file 2 elements of this probe with different names. You must also change the first and third parameters of the init method. The configuration file may stay the same if you want to retrieve the same data.

1.6 Configuration file of the GenericJmx probe

The syntax of the configuration file is the same as the one for the jonasJmx probe and the probe developed by the group of students last year.

Here is a small example:

```
<?xml version="1.0" encoding="UTF-8"?>
```

```
<probe-config>
  <probe-tasks dateFormat="yyyy_MM_dd_HH_mm" defaultRefreshPeriod="300000"
defaultSamplingPeriod="10000"/>
  <polls-descriptor logdateformat="dd/MM/yyyy HH:mm:ss" logtimeformat="HH:mm:ss" >
    <poll name="joramDestination">
      <discriminator>Joram:type=Destination,*</discriminator>
      <attribute-list>
        <attribute>
            <attribute-name>NbMsgsDeliverSinceCreation</attribute-name>
        </attribute>
        <attribute>
          <attribute-name>NbMsgsReceiveSinceCreation</attribute-name>
        </attribute>
        <attribute>
          <attribute-name>NbMsgsSendToDMQSinceCreation</attribute-name>
        </attribute>
      </attribute-list>
    </poll>
  </polls-descriptor>
</probe-config>
```

Remarks:

- The attributes of the elements “probe-tasks” and “polls-descriptor” are not used by JimysProbes. They are only there to be compatible to je J2EE probe developed by the other group of students.
- The element “poll” defines the name of the mbean to monitor. You can use regular expressions like “`*:j2eeType=JDBCDataSource,*` “. This means that all the mbeans matching this expression will be pulled to receive the information.
- The “attribute” elements define which attributes of the mbean are pulled.
- You can have multiple “poll” elements which define different mbeans in the same mbeanserver to pull. Like the sample configuration file of the jonasjmx probe.