INFORME FINAL
PROYECTO DE GRADO:

HERRAMIENTA DE MONITOREO Y ALERTA CONFIGURABLE (HMAC)

Guía para la instalación y configuración de HMAC

Ana Carolina Pascual, Marcos Rodríguez

Tutores: Ariel Sabiguero, Héctor Cancela

Instituto de Computación
Facultad de Ingeniería
Universidad de la República Oriental del Uruguay

Abril 2004

INDICE

1. OBJETIVOS

El objetivo de este documento es presentar el proceso de instalación de la herramienta HMAC, ubicación y especificación de los archivos de configuración necesarios para el normal funcionamiento de dicha herramienta.

1.1 Introducción

El monitor de alertas configurable (HMAC) está compuesto de tres componentes básicos Agente, Servidor y Consola de administración.
Estos componentes se ejecutan por separado con independencia entre sí de la máquina física en la que se instalan, es decir que no es necesario que se ejecuten en la misma máquina aunque pueden hacerlo.

Si se debe seguir un orden de ejecución de los componentes, para el buen funcionamiento de la herramienta deben ejecutarse los agentes antes que el servidor y por ultimo los clientes. La correcta ejecución del agente inicializa un conector que luego es invocado al inicializar el servidor para establecer la conexión entre ambos. Un mecanismo similar se da en la conexión entre un servidor y una consola de administración.

El inicio de la ejecución de cada componente comprende la lectura de un archivo de configuración que detalla la ubicación de los elementos necesarios para que cada uno cumpla con sus funcionalidades.

1.2 Alcance y contenido

En el documento se especifica el proceso de instalación y puesta en marcha del monitor, proceso de inicialización, ubicación de los archivos de configuración, formato y contenidos de dichos archivos.
Se comienza por comentar el proceso de inicialización de los componentes, definiendo archivos de configuración, ubicación y contenido de los mismos. Posteriormente se avanza en la construcción y formato de cada archivo.

En la sección 2 se menciona la instalación de Java para las plataformas W2K y Linux.
En la sección 3 se describen los pasos necesarios para instalar y configurar el ambiente para asegurar el correcto funcionamiento del monitor, archivos de configuración, rutas de los mismos y variables de entorno. También se incluye una lista de las librerías que deben ser incluidas en el CLASSPATH.
La sección 4 comenta en detalle el mecanismo de inicialización de los componentes, así como el proceso de instalación de cada uno de estos y una explicación detallada de los archivos de configuración necesarios.
Por ultimo se incluyen referencias a documentos de interés

1.3 Audiencia

Este documento busca ilustrar el proceso de configuración, instalación y puesta en marcha del monitor de alertas configurable.
Está destinado a aquellas personas responsables de la instalación y puesta en marcha de la herramienta.
También es de utilidad para quien esté encargado de diseñar los archivos de configuración ya que se detalla el contenido de cada unos de éstos.

Se asume que el lector tiene conocimientos básicos de la terminología y arquitectura de la tecnología JMX que da soporte a esta aplicación.
También se asume familiaridad por parte del lector con los sistemas operativos mencionados, con la instalación de herramientas y configuración de los ambientes. Igualmente se entra en detalle de algunos pasos necesarios para instalar Java y setear el CLASSPATH

Se puede hacer referencia a este contexto en el documento de “Informe final”, fundamentalmente en la sección 3.4 “Especificaciones de Diseño y Arquitectura” [Informe Final].

2. INSTALACIÓN DE JAVA

La instalación de Java es indispensable para que el monitor funcione adecuadamente. Para continuar con la instalación es necesario contar con Java Runtime Environment (JRE). Se puede obtener Java desde el sitio de Sun o desde el CD de instalación del HMAC.

Para el resto del manual se asume que se utilizará la maquina virtual Java versión 1.4.2.
Si se desea datos más detallados sobre la instalación puede referirse al manual de Sun para la instalación de Java en el sitio http://java.sun.com/j2se/1.4.2/jre/install.html

2.1.1 Windows

La correcta instalación y prueba de los componentes del HMAC fue realizada y verificada para WNT, W2000 SP4 y WXP SP1, no se ha verificado el funcionamiento W98, WME y W3.x

La instalación de Java consta de tres pasos

1. ejecutar el setup de Java. Por defecto el instalador sugiere instalar java en c:\j2sdk1.4.2
2. crear o actualizar la variable de entorno JAVA_HOME con el path donde se instalo Java. Por defecto será JAVA_HOME= c:\j2sdk1.4.2\jre\bin
3. crear o actualizar la variable de entorno PATH con el path de los binarios de java. Por defecto sera PATH= %JAVA_HOME%\bin

El seteo de las variables de entorno puede hacerse solo por la sesión en curso o de forma permanente.

Para cambiar las variables de entorno solo por la sesión en curso se debe ejecutar los siguientes comandos. NOTA: si se desea automatizar este proceso se puede crear un archivo .bat o .cmd

```
set     JAVA_HOME= c:\j2sdk1.4.2\jre\bin
set     PATH= %PATH%;%JAVA_HOME%\bin
```

NOTA: en Windows el separador es el caractér "punto y coma" ";".

Para cambiar las variables de entorno de manera permanente se deben seguir los siguientes pasos. NOTA: instrucciones solo para Windows NT, 2000 y XP

- abrir el panel de control.
- doble click en Sistema.
- en Windows NT seleccionar la lengüeta Entorno, En Windows 2000 y XP seleccionar la lengüeta Avanzado y después el botón Variables de entorno.
- la variable PATH es probable que ya exista, así que debe buscarla y agregar %JAVA_HOME%\bin asumiendo que la variable JAVA_HOME ya fue creada.

NOTA: en Windows el separador es el caractér "punto y coma" ";".

2.1.2 Linux

La instalación de Java en Linux consta de tres pasos y no es necesario tener permisos de root.

1. ejecutar el setup de Java. Por simplicidad asumimos que java ha sido instalado en /usr/local/j2sdk1.4.2
2. crear o actualizar la variable de entorno JAVA_HOME con el path donde se instaló Java. Por defecto sera JAVA_HOME= /usr/local/j2sdk1.4.2/jre

3. crear o actualizar la variable de entorno PATH con el path de los binarios de java. Por defecto sera PATH=${JAVA_HOME}\bin

El seteo de las variables de entorno puede hacerse solo por la sesión en curso o de forma permanente

Amodo de ejemplo, en una sesión BASH esto se haría con la ejecución de los siguientes comandos:

```
export JAVA_HOME=/usr/local/j2sdk1.4.2/jre
export PATH=${PATH}:${JAVA_HOME}\bin
```

NOTA1: en Linux el separador es el caractér "dos puntos" ":"

NOTA2: si se desea automatizar este proceso se puede crear un archivo .sh el cual debe ser invocado de la siguiente manera "source nombreScript" si el directorio donde el script reside esta en el path o de lo contrario si el directorio donde reside el script no esta en el path se debe usar "source /nombreScript"

Para cambiar las variables de entorno de manera permanente se deben modificar estas variables en los scripts de inicio. Por ejemplo si se usa bash como shell se debe modificar el archivo ".bashrc" que se encuentra en el home directory del usuario

3. INSTALACIÓN DEL MONITOR

3.1 Instalación

El proceso de instalación es simple y se resume en forma genérica en los cinco puntos siguientes:

- Dado que los procesos que corren los componentes se cargan en la máquina virtual Java es necesario tener la máquina virtual Java (jre 1.4.2) instalado para el correcto funcionamiento de la aplicación, según se explica en el capitulo anterior.
- Crear un directorio para alojar los archivos de configuración correspondientes a cada componente. Copiar los archivos de configuración correspondientes a la ruta creada.
- Crear el archivo configuration.ini, en este se detallan las rutas de los archivos de configuración de cada componente (las rutas creadas en el paso anterior).
  Por ejemplo **C:\monitor\configuration\configuration.ini.**
- Copiar el archivo de configuración (configuration.ini) en una ruta definida por el usuario, esta ruta debe ser pasada como parámetro al ejecutar el componente como se explica más adelante.
- Fijar en el CLASSPATH las librerías necesarias para la ejecución.

La ejecución de los componentes se inicia pasándole como parámetro la ubicación del archivo configuration.ini de la forma siguiente:

Java monitor.nombrecomponente **C:\monitor\configuration\configuration.ini**

En caso de que no se pase la ruta como parámetro el componente buscará el archivo en el directorio por defecto del usuario como se indica a continuación:

- Copiar el configuration.ini al directorio por defecto:
  $HOME\projects\monitor\configuration\.

Como se ha mencionado antes se debe seguir un orden en la ejecución de los componentes, primero se deben ejecutar los agentes, seguido del Servidor (pueden ser mas de uno) y por ultimo los clientes o consolas graficas. Esto se debe al mecanismo seguido para establecer las conexiones entre los componentes.

3.2 Descripción de Librerías necesarias

A continuación se detallan las librerías requeridas para que el HMAC funcione y un breve comentario del archivo setMonitor que puede ser usado si se desea automatizar solo para la sesión en curso el proceso. Estas de incluyen en el CD de instalación del HMAC.

3.2.1 setMonitor

En él se configura el path para la correcta ejecución de la máquina virtual java, así como también la incorporación de los path de todas las librerías necesarias para la ejecución de las herramientas HMAC. Cada una de estas rutas debe ser coherente con la ubicación de los archivos *.jar correspondientes. La ubicación es la siguiente, un nivel más bajo del directorio lib se crean los siguientes directorios:

lib/mx4j. Librerías para la utilización de mx4j.
- mx4j.jar
- mx4j-jmx.jar
- mx4j-remote.jar
- mx4j-rjmx.jar
- mx4j-tools.jar

lib/snmp Librerías para el manejo de snmp.
- snmp.jar

lib/mail Librerías para el manejo de envió de e-mail.
- mail.jar
- activation.jar

lib/db Librerías para el acceso a bases de datos

- jtds-0.6.jar encapsula la conexión a MSSqlServer 2000
- pg73jdbc3.jar encapsula la conexión a MSSqlServer

lib\adaptor Librerías para la interacción con el K&S Middleware
- KSAdaptor.jar

El Archivo KSAdaptor.jar es distribuido bajo licencia por la empresa K&S asociados y no se incluye en el CD.

4. INICIALIZACIÓN DE COMPONENTES

En esta sección se describe el procesote inicialización de los distintos componentes que integran la herramienta y se profundiza en detalles de los archivos de configuración.
Se detalla el proceso de inicialización de cada componente.

Los elementos básicos que integran la estructura del sistema de monitoreo y alertas son los ítems de monitoreo, los monitores y condiciones de alarma que evalúan el estado de los atributos de los ítems y las acciones que se ejecutan en caso de darse condición de alarma.
Es decir que el proceso de configuración para el correcto funcionamiento de la herramienta pasa por la correcta especificación de estos elementos en los archivos XML correspondientes.
Se puede consultar la guía del desarrollador para ver detalles de la implementación de estos elementos [GD].

A continuación de describe el proceso de inicialización de cada uno de los componentes por su orden lógico (Agentes, Servidor y Consolas gráficas).

4.1 Preparando el ambiente

Previo a la ejecución de cada componente se deben seguir dos pasos iniciales:

- Setear el CLASSPATH
  - Para fijar el CLASSPATH en Windows se ejecuta el archivo setMonitor.bat. y en Linux se ejecuta el archivo setMonitor.sh

- Cargar el servicio de registro RMI
  - Ejecutar *rmiregistry*

En la versión actual se debe ejecutar un rmiregistry por componente, de esta forma se asegura el funcionamiento por separado de cada componente. Se estudia la posibilidad de mantener un rmiregistry por sistema.

4.2 Agente

Activar el servicio de registro de RMI con la ejecución del comando *rmiregistry* en una sesión independiente.
El proceso de inicialización del componente comienza con la siguiente ejecución:

```
java monitor.agent.AgentManger  ruta_archivo\configuration.ini
```

El parámetro especifica el camino desde donde el componente lee el archivo de configuración.
Si no se le pasa el parámetro, el componente intenta leer el archivo del directorio por defecto.

En este se especifican las rutas desde donde se cargan los elementos necesarios para inicializar los ítems, acciones, monitores, y conectores así como también mecanismo de registro.
Para el caso del Agente este archivo se divide en las siguientes secciones:

```
AGENTMANAGER=XML c:\monitor\configuration\confAgent.xml
LOGGER=XML c:\monitor\configuration\Logger.xml
```

El proceso lee del archivo configuration.ini las rutas marcadas por las etiquetas AGENTMANAGER y LOGGER.

Al procesar el archivo Logger.xml se va leyendo la información necesaria para configurar el servicio de registro.
El procesamiento del archivo confAgent.xml permite ir inicializando y registrando los datos correspondientes a los ítems a monitorear, las acciones utilizadas y los monitores con las asociaciones de (ítem, atributo, acción) para cada uno.
Al terminar de procesar este archivo se inician los monitores y se comienza la actividad de monitoreo.

En caso de haber problemas y el Agente no se pueda ejecutar se envía un mensaje de error a la consola estándar y en caso de estar habilitado el registro a archivos se registra en un archivo. El camino del archivo está especificado en el archivo Logger.xml , el nombre AgentManager_fecha.log. En el ejemplo siguiente sería `c:\\monitor\\logMonitor\\AgentManager_fecha.log`.

Descripción de Logger.xml

En él se definen parámetros que determinan el comportamiento del registro de la aplicación.
Las secciones definidas permiten definir si el registro se hace en archivos de texto, base de datos o si se despliega en la consola de ejecución.

En cualquiera de los casos el valor “disabled” inhabilita la sección correspondiente

```
<level>WARNING</level>
<stackTrace>enabled</stackTrace>
<filehandler>
        <status>enabled</status>
      <path>c:\\monitor\\logMonitor</path>
</filehandler>
```

Define el comportamiento para el registro en archivos, el valor “enabled” en status indica que se debe registrar en archivo y la sección path define la ruta del archivo donde se registra los datos de la ejecución de la aplicación

```
<consolehandler>
      <status>disabled</status>
</consolehandler>
```

Si el valor de status indica “enabled”, se imprimen los datos de ejecución en la salida estándar

```
<databasehandler>
  <status>disabled</status>
 <driver>net.sourceforge.jtds.jdbc.Driver</driver>
  <databaseURL>jdbc:jtds:sqlserver://ServidorBD/nombreBD</databaseURL>
  <user>UserBD</user>
   <password>password</password>
</databasehandler>
```

Especifica el registro en base de datos, si el valor de status en “enabled” habilita el registro en la base de datos con los parámetros siguientes:

- Driver. Driver usado conexión a la base de datos.
- databaseURL. URL de conexión a la base de datos, en el se definen servidor y base de datos
- user. Usuario de la base de datos
- password. Contraseña del usuario en la base de datos

Descripción de AgentManager.xml

En el se especifican los datos necesarios para configurar todos los elementos para monitorear los recursos, ítems de monitoreo, acciones a ejecutar, monitores y conectores.
Se divide en cuatro secciones:

- Ítems
- Actions
- Monitors
- Connectors

Ítems. En la sección de Items se definen los parámetros necesarios para configurar los elementos a monitorear, nombre del recurso, y los argumentos que utiliza el constructor de de la clase que implementa al MBean que modela el recurso.

```
<item type=”NombreClase” objectName="ping1">
        <arg name="arg1" type="string">valor1</arg>
        <arg name="arg2" type="string">valor2</arg>
        <arg name="argn" type="string">valorN</arg>
</item>
```

Tipo del recurso. Indica la clase que implementa el recurso.
ObjectName. Nombre del objeto que instancia de la clase.
Los argumentos siguientes indican los parámetros que recibe el constructor de la clase, deben ser listados en el mismo orden que los recibe el constructor.

Actions. En la sección Actions se definen los parámetros necesarios para configurar cada acción definida Se especifica tipo de acción y nombre correspondiente, también los argumentos que recibe el constructor de la clase que implementa la acción.

```
<action type="NombreClase" objectName="NombreInstancia">
        <arg name="arg1" type="string">valor1</arg>
        <arg name="arg2" type="string">valor2</arg>
    </action>
```

Tipo de Acción. Indica la clase que implementa la acción.
ObjectName. Nombre del objeto que instancia la clase acción.
Los argumentos deben seguir el orden en que los recibe el constructor de la clase.

Monitors. En la sección Monitors se especifican los monitores que controlan los atributos de los ítems declarados en secciones anteriores, también se relacionan con las acciones a ejecutar en caso de que se de alguna condición de alarma.
Es posible asociar a un atributo más de un monitor y por medio de estos más de una acción.
Cada monitor se configura para chequear el valor del atributo correspondiente con una frecuencia particular, es necesario asociar en éste el nombre del ítem propietario del atributo, el nombre del atributo o la condición de excepción y la acción o acciones que se ejecutarían en caso de excepción.
Es decir que el monitor se ejecuta cada una cierta frecuencia, chequeando el valor del atributo y en caso de darse una condición de excepción el monitor crea una notificación que es propagada y finalmente será manejada por la acción correspondiente definida en la configuración del monitor.

En las posibilidades actuales de monitoreo hay cuatro tipos de monitor, String, Counter, Gauge y Generic.

Especificación del String monitor.

```
<monitor type="String" objectName="monitor1">
        <arg name="observedObj" type="string">ping1</arg>
        <arg name="action" type="string">action1</arg>
        <arg name="attribute" type="string">Status</arg>
        <arg name="frequency" type="int">20000</arg>
        <arg name="condition" type="string">FALSE</arg>
    </monitor>
```

En el caso del String monitor se debe especificar el tipo de monitor “String”, el objectName que identificará la instancia del StringMonitor, en el ejemplo “monitor1”.
Los argumentos se detallan:
observedObj. Nombre del objeto que se está monitoreando.
Action. La acción a ejecutar es la acción “accion1”
Attribute. El atributo monitoreado es el atributo “status”, se chequea el valor del atributo ejecutando get +”status”
Frequency. Indica la frecuencia con que se recupera el valor del atributo. Se expresa en milisegundos.
Condition. Condición con la se compara el valor del atributo. Si coinciden se genera una condición de alarma.

Especificación del Counter Monitor

```
<monitor type="Counter" objectName="monitor6">
        <arg name="observedObj" type="String">SNMP1</arg>
        <arg name="action" type="string">action6</arg>
        <arg name="attribute" type="string">Value</arg>
        <arg name="frequency" type="int">10000</arg>
        <arg name="condition" type="String">25000</arg>
    </monitor>
```

En el caso del Counter monitor se debe especificar el tipo de monitor “Counter”, el objectName que identificará la instancia del CounterMonitor, en el ejemplo “monitor6”.
Los argumentos se detallan:
observedObj. Nombre del objeto que se esta monitoreando.
Action. La acción a ejecutar es la acción “accion6”
Attribute. El atributo monitoreado es el atributo “Value”, se chequea el valor ejecutando getValue()
Frequency. Indica la frecuencia con que se recupera el valor del atributo. Se expresa en milisegundos.
Condition. Expresa el valor superior aceptable. Si el atributo supera ese valor, se genera la condición de alarma.

Especificación del Gauge Monitor

El tipo de monitor es “Gauge” y se define como el CounterMonitor con la salvedad de que se chequean dos condiciones

```
        <arg name="conditionLow"  type="String">MinVal</arg>
        <arg name="conditionHigh" type="String">MaxVal</arg>
```

Especificación del Generic Monitor

```
<monitor type="Generic" objectName="monitor4">
        <arg name="observedObj" type="String">Adaptor</arg>
        <arg name="action" type="string">action8</arg>
        <arg name="frequency" type="int">20000</arg>
        <arg name="condition" type="String">Value > 0 </arg>
</monitor>
```

El tipo de monitor es “Generic” y el nombre que lo identifica “monitor4”.

Los argumentos son los siguientes:
observedObj. Nombre del objeto que se está monitoreando.
Action. La acción a ejecutar es la acción “accion8”.
Frequency. Indica la frecuencia con que se mira el valor del atributo en milisegundos.
Condition. Se detalla la expresión lógica a evaluar. En esta expresión las variables solo podrán ser aquellos atributos disponibles para el tipo de ítem que se está monitoreando, en el ejemplo es “Value”. La expresión puede tener tanto operadores lógicos como aritméticos, y la siguiente lista de funciones.

Connectors. En la sección connectors se especifica el mecanismo de conexión utilizado para que el Servidor o la Consola de administración se puedan conectar con el Agente. El Agente debe habilitar este conector para que los otros componentes accedan a él.

```
<connector protocol="rmi" objectName="connector1">
        <arg name="host" type="string">localhost</arg>
        <arg name="port" type="int">1099</arg>
        <arg name="idConn" type="string">/jndi/connector1</arg>
</connector>
```

Protocol. Define el protocolo de comunicación empleado
ObjectName. Nombre del conector
Host. Host donde corre el servicio rmiregistry
Port. Puerto en que se establece la conexión
IdConn. Identificador de la conexión

4.3 Servidor

El Proceso de inicialización del componente comienza con la siguiente ejecución:

java monitor.server.ServerManager ruta_archivo\configuration.ini

El parámetro especifica el camino desde donde el componente lee el archivo de configuración.
Si no se le pasa el parámetro, el componente intenta leer el archivo en el directorio por defecto.
En éste se especifican las rutas desde donde se cargan los elementos necesarios para inicializar los ítems, acciones, monitores, y conectores así como también mecanismo de registro.
Se divide en las siguientes secciones:

```
SERVERMANAGER=XML c:\monitor\configuration\confServer.xml
LOGGER=XML c:\monitor\configuration\Logger.xml
AGENTS=XML c:\monitor\configuration\AgentsConnectors.xml
```

La etiqueta SERVERMANAGER indica la ruta del archivo confServer.xml que configura ítems, acciones, monitores, y conectores para permitir a las Consolas Cliente acceder a él. Este archivo se divide en las mismas secciones que el confAgent.xml y el procedimiento para su construcción es similar.

La etiqueta LOGGER indica la ruta desde donde leer el archivo de configuración del mecanismo de registro, se sigue el mismo procedimiento que en el Agente.

AGENTS. Indica la ruta del archivo que registra los agentes a los que el Servidor intentará conectarse. Para cada agente se asocia un conector a través del cual se establecerá la conexión.

Al procesar el archivo Logger.xml se van leyendo la información necesaria para configurar el servicio de registro.
El procesamiento del archivo confServer.xml permite ir inicializando y registrando los datos correspondientes a los recursos a monitorear, las acciones utilizadas y los monitores con las asociaciones de (ítem, atributo, acción) para cada uno y los conectores. En general, en este archivo solamente se cargan los conectores para permitir a las Consolas de administración accede a él, pero no es un impedimento que también monitoree recursos como los agentes.
Luego de tener registrados los conectores es necesario leer el archivo AgentsConnectors.xml para establecer las conexiones con los Agentes y obtener la información de todos los ítems y monitores que corren en cada uno de éstos. El Servidor mantiene una referencia remota de cada uno de esos objetos.

En caso de haber problemas y el Servidor no se pueda ejecutar se envía un mensaje de error a la consola estándar o en caso de estar "enabled" se registra en un archivo el camino especificado en el archivo Logger.xml, en el ejemplo siguiente seria `c:\\monitor\\logMonitor\\ServerManager_fecha.log`.

Descripción de Agents.xml

```
<Agents>
        <Agent name = "AGENTE1">
                <protocol>rmi</protocol>
                <host>localhost</host>
                <port>1099</port>
                <name>/jndi/connector1</name>
        </Agent>
</Agents>
```

En la sección <Agents> se listan todos los conectores a ser usados, estos definen el mecanismo de comunicación que se utilizará desde el Servidor para conectarse a cada Agente.
Agent name. Nombre del agente con el cual se define el conector. Sólo tiene utilidad para la lectura del archivo.
Se va a crear un conector RMI con el nombre "/jndi/connector1", en el puerto 1099 y el servicio RMI se va a estar ejecutando en la máquina local al Servidor.

Descripción de connectors.xml

En la sección connectors se especifica el mecanismo de conexión utilizado para que la(s) Consola (s) de Administración puedan establecer la conexión con el Servidor.

```
<Connectors>
        <Connector name = "Connector2">
                <protocol>rmi</protocol>
                <host>localhost</host>
                <port>1099</port>
                <name>/jndi/connector2</name>
        </Connector>
</Connectors>
```

Protocol. Define el protocolo de comunicación empleado
Host. Host donde está el conector (servidor)
Port. Puerto en donde se quiere establecer la conexión
name. Identificador de la conexión

4.4 Consola de administración

El Proceso de inicialización del componente comienza con la ejecución siguiente:
java monitor.cliente.ManagementConsole ruta_archivo\configuration.ini

El parámetro especifica el camino desde donde el componente lee el archivo de configuración.
Si no se le pasa el parámetro, el componente intenta leer el archivo en el directorio por defecto.
En este se especifican las rutas desde donde se cargan los elementos necesarios para inicializar el conector que implementa la comunicación con el Servidor así como también el mecanismo de registro.
Se divide en las siguientes secciones:

```
CONNECTORS=XML c:\monitor\configuration\confClient.xml
LOGGER=XML c:\monitor\configuration\Logger.xml
```

En la sección connectors se especifica el mecanismo de conexión utilizado para establecer la conexión con el componente Servidor o con el Agente. Para poder establecer la conexión del lado del Servidor o Agente se debe haber habilitado el conector asociado previamente.

```
<Connectors>
        <Connector name = "Connector2">
                <protocol>rmi</protocol>
                <host>localhost</host>
                <port>1099</port>
                <name>/jndi/connector2</name>
        </Connector>
</Connectors>
```

protocol. Define el protocolo de comunicación empleado
host. Host donde está el conector
Port. Puerto en donde se quiere establecer la conexión
name. Identificador de la conexión

En caso de haber problemas y la consola no se pueda ejecutar se envía un mensaje de error a la consola estándar o en caso de estar “enabled” se registra en un archivo el camino especificado en el archivo Logger.xml, en el ejemplo siguiente sería `c:\monitor\logMonitor\ManagementConsole_fecha.log`

5. REFERENCIAS

[Informe Final] Informe Final del proyecto de grado: Monitor de Alertas Configurable.

[JMX]
http://java.sun.com/products/JavaManagement/ fecha último ingreso: 12/04/2004

[MX4J]
http://mx4j.sourceforge.net fecha último ingreso: 12/04/2004

[SUN]
http://java.sun.com fecha último ingreso: 12/05/2004