

Type of Monitors and Their Usage

There are two types of monitors: performance monitors and non-performance monitors. The performance monitors have history and alerting mechanisms, which includes thresholds (used to compare the actual monitored value with a predefined value, and according to the result, to map a color to the monitor and create alerts). The performance monitors are: availability-monitor, integer-monitor, long-monitor, frequency-monitor, quality-rate-monitor, and duration-monitor.

The non-performance monitors do not have history and beside the state monitor, no alert mechanisms. The non-performance monitors are: text-monitor, state-monitor, table-monitor, version-monitor, and configuration-monitor.

1) Non-performance Monitors

text-monitor

Usage: Monitors a String.

This monitor just look to the String value and shows it in the monitor tree.

Example:

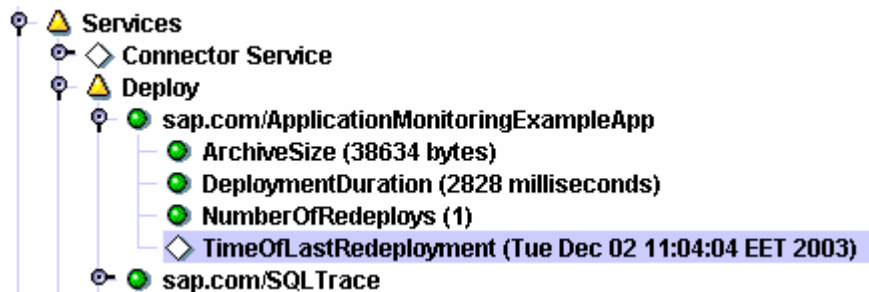
```
<text-monitor name="TimeOfLastRedeployment" configuration-  
group="DEPLOY.TimeOfLastRedeployment">  
  <monitored-resource name="deploy" type="SERVICE"/>  
  <text-attribute-mapping>  
    <text-attribute>  
      <observed-resource-attribute name="TimeOfLastRedeployment"/>  
    </text-attribute>  
  </text-attribute-mapping>  
</text-monitor>
```

where method returning the monitored String is :

```
public String getTimeOfLastRedeployment();
```

(in this example this is a String representation of the time of the last redeployment)

representation in the tree:



representation in the viewer:

<i>General Info:</i>	
Name:	TimeOfLastRedeployment
Description:	Time of last redeployment
Type:	StringMonitor
Configuration group:	DEPLOY.TimeOfLastRedeployment
Creation date:	Tue Dec 02 11:04:58 EET 2003
Last change date:	Tue Dec 02 11:05:03 EET 2003
 <i>Monitored Data:</i>	
Monitored string:	Tue Dec 02 11:04:04 EET 2003

state-monitor

Usage: Monitors a String but with additional property of assigning colors (green, yellow, red) to the String values.

Must be defined a mapping from specific states to different colors (red, yellow, and green). The monitor compares the monitored states with the predefined ones, and according to the result, colors the state monitor nodes and potentially generates alerts.

For example state monitor can be used for the expiration state of some certificate.

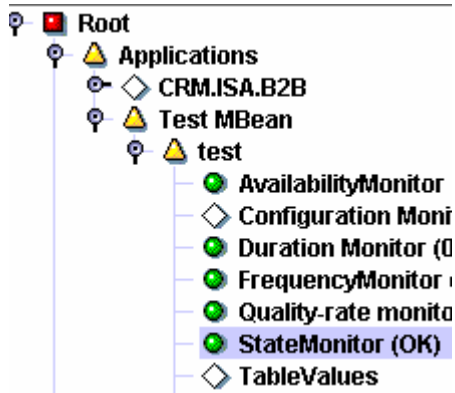
Example:

```
<state-monitor name="StateMonitor" configuration-group="StateMonitor">
  <monitored-resource name="TestMonitorMBean:j2eeType=SAP_TestMonitor_MBeans"
type="MBean"/>
  <state-attribute-mapping>
    <state-attribute>
      <observed-resource-attribute name="TestState"/>
    </state-attribute>
  </state-attribute-mapping>
</state-monitor>
```

where the method returning the monitored state is :

```
public String getTestState();
```

representation in the tree:



representation in the viewer:

General Info:

Name: StateMonitor
Description: State monitor
Type: StateMonitor
Configuration group: StateMonitor
Creation date: Tue Dec 02 13:54:52 EET 2003
Last change date: Tue Dec 02 13:57:57 EET 2003

Monitored Data:

State: OK

and the states for this example are :

General		States
red states	yellow states	greenStates
ERROR	WARNING	OK
not working	unknown	working

table-monitor

Usage: Monitors a table (each table element is a serializable object).

Example:

```
<table-monitor name="System properties" filename="SYSTEM.SYS_PROPS" configuration-group="SYSTEM.System_Props ">
```

```

    <monitored-resource name="monitor" type="SERVICE"/>
      <table-attribute-mapping>
        <table-entries-attribute>
          <observed-resource-attribute name="SystemProperties"/>
        </table-entries-attribute>
      </table-attribute-mapping>
    </table-monitor>

```

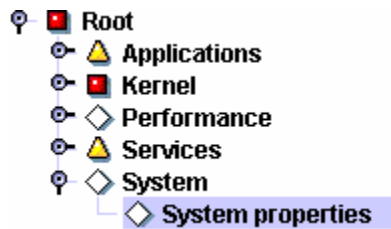
where method returning the monitored table is :

```

public Serializable[][] getSystemProperties ();

```

representation in the tree:



representation in the viewer:

General Info:

Name: System properties
Description: J2EE system properties
Type: TableMonitor
Configuration group: J2EE System props
Creation date: Tue Dec 02 13:48:40 EET 2003
Last change date: Tue Dec 02 13:48:42 EET 2003

Monitored Data:

System Property	Value
AUDIT	jni
JCO:JNI Layer Version	
JCO:Java Middleware Part Version	1.0.3
JCO:Java Part Version	6.30.3 (2003-11-21)
JCO:RFC Layer Version	
JCO:jco.middleware.libjrfc_path	
JCO:jco.middleware.librfc_path	system defined path
LoadBalanceRestricted	no
SAPDBHOST	
SAPMYNAME	Tsvetelina-T_C11_00
SAPSTARTUP	4

version-monitor

Usage: Monitors VersionInfo, which is a version information of a component. The reported version consists of an application name, a major version, a minor version, a support

package number, a build time, a change list number, and an optional entry for additional information.

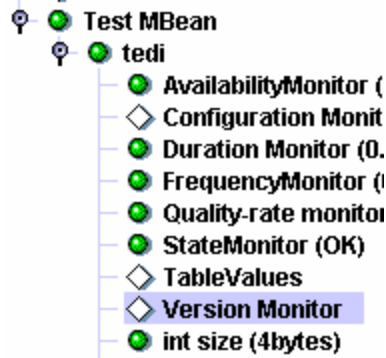
Example:

```
<version-monitor name="Version Monitor" configuration-group="TEST.VersionMonitor">
  <monitored-resource name="TestMonitorMBean:j2eeType=SAP_TestMonitor_MBeans"
type="MBean"/>
  <version-attribute-mapping>
    <version-attribute>
      <observed-resource-attribute name="VersionParameters"/>
    </version-attribute>
  </version-attribute-mapping>
</version-monitor>
```

where the method returning the monitored version information is :

```
public com.sap.jmx.monitoring.api.VersionInfo getVersionParameters ();
```

representation in the tree:



representation in the viewer:

General Info:

Name:	Version Monitor
Description:	Version monitor
Type:	PropertyListMonitor
Configuration group:	VersionMonitor
Creation date:	Tue Dec 02 11:00:40 EET 2003
Last change date:	Tue Dec 02 11:20:49 EET 2003

Monitored Data:

property	value
Application	sap.com/TestMonitor
Major version	222
Minor version	111
Support package	support package name
Build time	Friday, September 12, 2003 00:04 GMT
Changelist number	5353

configuration-monitor

Usage: Monitors ConfigurationList - some configuration parameters of a particular component.

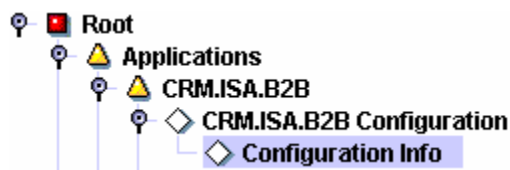
Example:

```
<configuration-monitor name="Configuration Info" configuration-group="J2EE Configuration Attribute">
  <monitored-resource
name="com.sap.jmx.monitoring.examples.MyConfigResourceMBean" type="APPLICATION"/>
  <configuration-attribute-mapping>
    <configuration-attribute>
      <observed-resource-attribute name="ConfigurationParameters"/>
    </configuration-attribute>
  </configuration-attribute-mapping>
</configuration-monitor>
```

where the method returning the monitored configuration is :

public com.sap.jmx.monitoring.api.ConfigurationList getConfigurationParameters();

representation in the tree:



representation in the viewer:

General Info:

Name:	Configuration Info
Description:	Configuration information
Type:	PropertyListMonitor
Configuration group:	J2EE Configuration Attribute
Creation date:	Tue Dec 02 11:04:13 EET 2003
Last change date:	Tue Dec 02 11:05:29 EET 2003

Monitored Data:

property	value
parameter1	valueA
parameter2	valueB
parameter3	valueC
parameter4	valueD

2) Performance Monitors

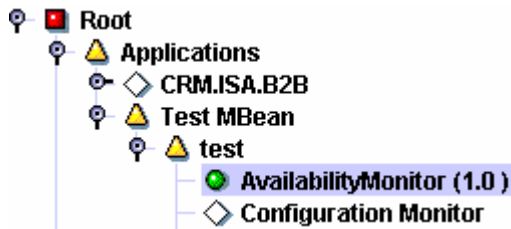
availability-monitor

Usage: Monitors a boolean value e.g. used to monitor the availability of a resource.
For example does

```
<availability-monitor name="AvailabilityMonitor" configuration-group="AvailabilityValue">  
  <monitored-resource name="TestMonitorMBean;j2eeType=SAP_TestMonitor_MBeans"  
type="MBean"/>  
  <availability-attribute-mapping>  
  <availability-attribute>  
    <observed-resource-attribute name="Availability"/>  
  </availability-attribute>  
  </availability-attribute-mapping>  
</availability-monitor>
```

where the method returning the monitored value is
public boolean getAvailability ();

representation in the tree:



representation in the viewer:

<i>General Info:</i>	
Name:	AvailabilityMonitor
Description:	Avalability Value
Type:	AvailabilityMonitor
Configuration group:	AvalabilityValue
Creation date:	Tue Dec 02 13:54:52 EET 2003
Last change date:	Tue Dec 02 13:54:57 EET 2003
<i>Monitored Data:</i>	
Availability:	true
Availability in Percent:	100.00%
<i>Available time in ms:</i>	<i>5031</i>
<i>Non available time in ms:</i>	<i>0</i>

integer-monitor

Usage: Monitors an int value. It can be for the number of threads, database connections, classloaders and so on.

Example:

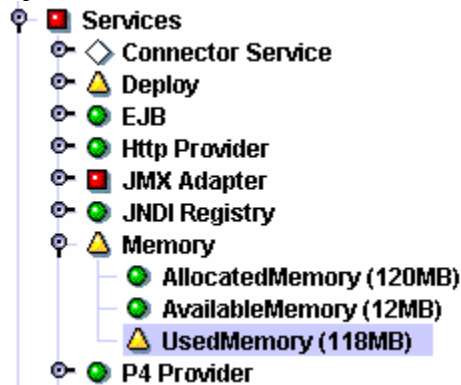
```
<integer-monitor name="UsedMemory" configuration-group="MEMORY.UsedMemory">
  <monitored-resource name="memory" type="SERVICE"/>
  <integer-attribute-mapping>
    <integer-attribute>
      <observed-resource-attribute name="UsedMemory"/>
    </integer-attribute>
  </integer-attribute-mapping>
</integer-monitor>
```

where the method returning the monitored integer value is:

```
public int getUsedMemory();
```

(in this example this is the currently used memory in MB)

representation in the tree:



representation in the viewer:

General Info:

Name:	UsedMemory
Description:	Currently used memory
Type:	IntegerMonitor
Configuration group:	MEMORY.UsedMemory
Creation date:	Tue Dec 02 13:48:45 EET 2003
Last change date:	Tue Dec 02 14:13:50 EET 2003

Monitored Data:

Value:	118 MB		
<i>Maximum:</i>	118	02.12.03	02:13 PM
<i>Minimum:</i>	88	02.12.03	02:08 PM

long-monitor

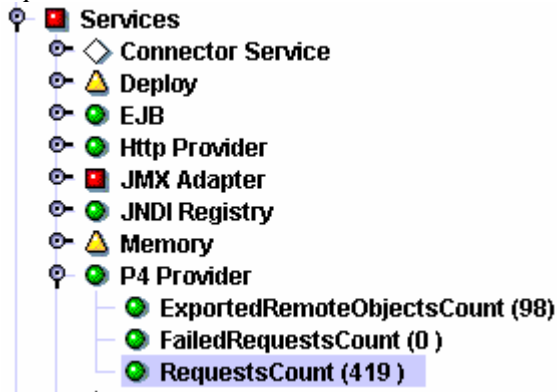
Usage: Monitors a long value. It can be used for

```
<long-monitor name="RequestsCount" configuration-group="P4.RequestsCount">
  <monitored-resource name="p4" type="SERVICE"/>
  <long-attribute-mapping>
    <long-attribute>
      <observed-resource-attribute name="RequestCount"/>
    </long-attribute>
  </long-attribute-mapping>
</long-monitor>
```

where the method returning the monitored long value is (in the example this is the number of P4 requests since server startup) :

```
public long getRequestCount();
```

representation in the tree:



representation in the viewer:

<i>General Info:</i>	
Name:	RequestsCount
Description:	Number of P4 requests since server startup
Type:	LongMonitor
Configuration group:	P4.RequestsCount
Creation date:	Tue Dec 02 13:48:49 EET 2003
Last change date:	Tue Dec 02 14:18:54 EET 2003
 <i>Monitored Data:</i>	
Value:	419
<i>Maximum:</i>	419
<i>Minimum:</i>	0

frequency-monitor

Usage: computes a frequency according to reported number of events given at specific times, by the rule:

frequency = reported number of events / time interval

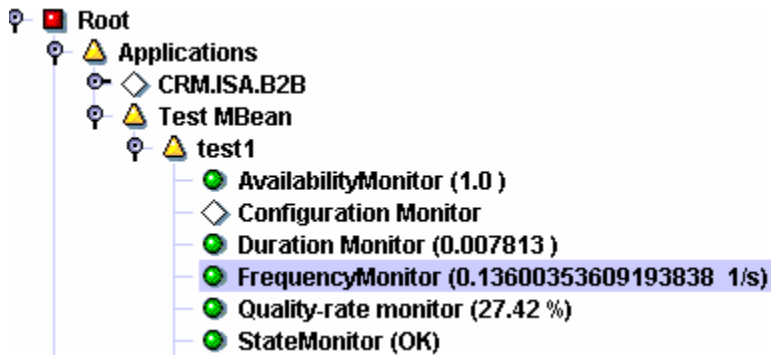
For example if you want to know for some application the number of http request per second
And there are 300 requests to the application for one minute, that means the the frequency is 5 requests/per second.

Example:

```
<frequency-monitor name="FrequencyMonitor" configuration-group="FrequencyMonitor">
  <monitored-resource
name="TestMonitorMBean:j2eeType=SAP_TestMonitor_MBeans" type="MBean"/>
  <frequency-attribute-mapping>
    <reported-events-attribute>
      <observed-resource-attribute name="ReportedEvents"/>
    </reported-events-attribute>
  </frequency-attribute-mapping>
</frequency-monitor>
```

where the method returning the number of events is:
public int getReportedEvents();

representation in the tree:



representation in the viewer:

General Info:

Name: FrequencyMonitor
Description: Frequency monitor
Type: FrequencyMonitor
Configuration group: FrequencyMonitor
Creation date: Tue Dec 02 16:47:25 EET 2003
Last change date: Tue Dec 02 16:51:31 EET 2003

Monitored Data:

Frequency: 0.14 1/s
Maximal frequency: 2.39
Minimal frequency: 0.14

quality-rate-monitor

Usage: Computes an average (and actual) quality rate according to reported number of total tries and successful tries.

For example you have 1 successful try from a total amount of 10 tries. Then the average quality rate would be 10%.

Example :

Case 1)

```
<quality-rate-monitor name="Cache Hit Rate" configuration-group="ConfigMgr.CacheHitRate">
  <monitored-resource name="ConfigurationManager" type="MANAGER"/>
    <quality-rate-attribute-mapping>
      <total-tries-attribute>
        <observed-resource-attribute name="AccNrOfRequests"/>
      </total-tries-attribute>
      <total-hits-attribute>
        <observed-resource-attribute name="AccNrOfCacheRequests"/>
      </total-hits-attribute>
    </quality-rate-attribute-mapping>
  </quality-rate-monitor>
```

where the method returning the total tries from startup is

public long getAccNrOfRequests ();

and

where the method returning the total successful tries from startup

public long getAccNrOfCacheRequests ();

Case 2)

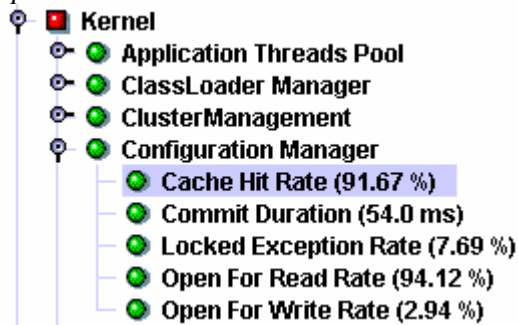
```

<quality-rate-monitor name="Cache Hit Rate" configuration-group="ConfigMgr.CacheHitRate">
  <monitored-resource name="ConfigurationManager" type="MANAGER"/>
    <quality-rate-attribute-mapping>
      <quality-rate-attribute>
        <observed-resource-attribute name="QualityRate"/>
      </quality-rate-attribute>
    </quality-rate-attribute-mapping>
</quality-rate-monitor>

```

where the method returning the necessary data is
public com.sap.jmx.monitoring.api.QualityRateValue getQualityRate();

representation in the tree:



representation in the viewer:

General Info:	
Name:	Cache Hit Rate
Description:	Cache Hit Rate
Type:	QualityRateMonitor
Configuration group:	ConfigMgr.CacheHitRate
Creation date:	Tue Dec 02 13:48:44 EET 2003
Last change date:	Tue Dec 02 13:58:49 EET 2003
Monitored Data:	
Current hit rate:	91.67%
Average hit rate since startup:	56.31%
Total tries:	1902
Total hits:	1071

duration-monitor

Usage: Compute the average time for a process. Defines a monitor for measuring the time for the http responses, session duration, etc.

If N and T are respectively the total number of responses and the total response time, the values under the last measurement node are calculated as follows:

number of requests = $N_i - N_{i-1}$

average time = $T_i - T_{i-1} / N_i - N_{i-1}$ or 0 if $N_i = N_{i-1}$.

Example:

Case 1)

```
<duration-monitor name="Commit Duration" configuration-
group="ConfigMgr.CommitDuration">
  <monitored-resource name="ConfigurationManager" type="MANAGER"/>
  <duration-attribute-mapping>
    <total-number-attribute>
      <observed-resource-attribute name="AccNrOfCommits"/>
    </total-number-attribute>
    <total-time-attribute>
      <observed-resource-attribute name="AccCommitTime"/>
    </total-time-attribute>
  </duration-attribute-mapping>
</duration-monitor>
```

where the method returning the total number of tries

(in the example - Returns the accumulated number of commit-requests since the startup of the server)

```
public long getAccNrOfCommits();
```

and

the method returning the total time is :

(in the example - Returns the accumulated time of processing of all commit-requests since the startup of the server in milliseconds)

```
public long getAccCommitTime();
```

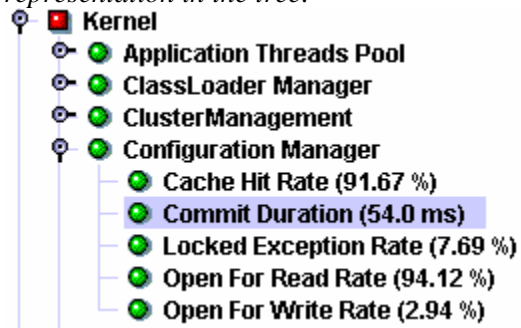
Case 2)

```
<duration-monitor name="Commit Duration" configuration-
group="ConfigMgr.CommitDuration">
  <monitored-resource name="ConfigurationManager" type="MANAGER"/>
  <duration-attribute-mapping>
    <duration-attribute>
      <observed-resource-attribute name="DurationValue "/>
    </duration-attribute>
  </duration-attribute-mapping>
</duration-monitor>
```

where the method returning the necessary data is :

```
public com.sap.jmx.monitoring.api.DurationValue getDurationValue();
```

representation in the tree:




representation in the viewer:

General Info:

Name:	Commit Duration
Description:	Commit Duration
Type:	DurationMonitor
Configuration group:	ConfigMgr.CommitDuration
Creation date:	Tue Dec 02 13:48:44 EET 2003
Last change date:	Tue Dec 02 13:58:49 EET 2003

Monitored Data:

current average time:	54.0ms
average time since startup:	169.48ms
total number:	60
total time:	10169

Monitor type	Example						
Text-monitor	Hello World						
Table-monitor	<table border="1"> <tr> <td>1</td> <td>2</td> <td>3</td> </tr> <tr> <td>4</td> <td>5</td> <td>6</td> </tr> </table>	1	2	3	4	5	6
1	2	3					
4	5	6					
Version-monitor	<table border="1"> <tr> <td>property</td> <td>value</td> </tr> <tr> <td></td> <td></td> </tr> </table>	property	value				
property	value						
Configuration-monitor							
Status-monitor	 Low performance (color + text)						
Availability-monitor	Cluster node 4711 available: true history						
Integer-monitor	Licence validity: 30 [days]						
Long-monitor	Number of milliseconds from year 0: 6307200000000						
Frequency-monitor	Request/s: 30.5 [1/s]						
Quality-rate-monitor	cache hit rate: 20%						
Duration-monitor	Average response time for http requests: 5 ms						

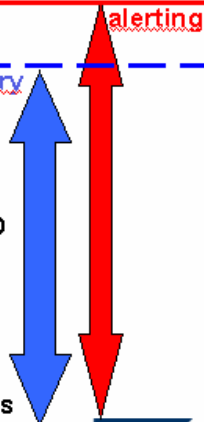


Fig. A slide from the presentation of Joerg Weller, which can be found here: http://bis.wdf.sap-ag.de:1080/general/docs/quickies/2003-11-19/ppt/ok_NEU_JMX_Monitoring.ppt