

## **ELO Webseminar**

### **Tomcat Monitoring via JDK**

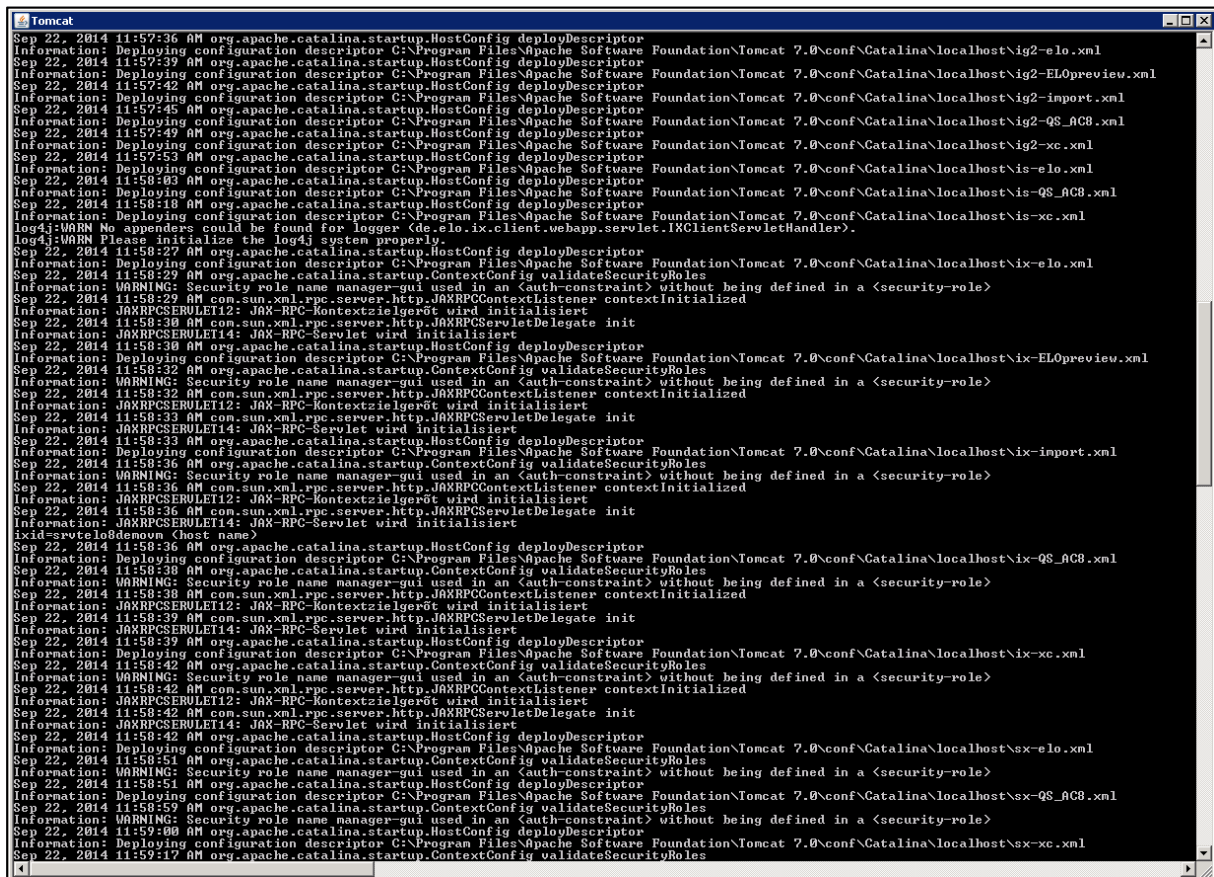
Christian Neumann  
Leiter Support  
ELO Digital Office GmbH

## Inhaltsverzeichnis

1	Thema .....	3
2	Zweck .....	3
3	Einrichtung der Tomcat / ELO Server Engine Überwachung .....	3
4	Java Developer Kit.....	5
5	Monitoring .....	5
6	Basis Monitoring Optionen .....	7
7	Erweiterte Monitor Optionen .....	9
8	Plugins installieren .....	11
9	Fazit .....	17



Damit startet Java und der Tomcat läuft dann in diesem „Konsolenfenster“. Beispiel Screenshot:



```

Sep 22, 2014 11:57:36 AM org.apache.catalina.startup.HostConfig deployDescriptor
Information: Deploying configuration descriptor C:\Program Files\Apache Software Foundation\Tomcat 7.0\conf\Catalina\localhost\ig2-elo.xml
Sep 22, 2014 11:57:39 AM org.apache.catalina.startup.HostConfig deployDescriptor
Information: Deploying configuration descriptor C:\Program Files\Apache Software Foundation\Tomcat 7.0\conf\Catalina\localhost\ig2-ELOPreview.xml
Sep 22, 2014 11:57:42 AM org.apache.catalina.startup.HostConfig deployDescriptor
Information: Deploying configuration descriptor C:\Program Files\Apache Software Foundation\Tomcat 7.0\conf\Catalina\localhost\ig2-import.xml
Sep 22, 2014 11:57:45 AM org.apache.catalina.startup.HostConfig deployDescriptor
Information: Deploying configuration descriptor C:\Program Files\Apache Software Foundation\Tomcat 7.0\conf\Catalina\localhost\ig2-QS_AC8.xml
Sep 22, 2014 11:57:49 AM org.apache.catalina.startup.HostConfig deployDescriptor
Information: Deploying configuration descriptor C:\Program Files\Apache Software Foundation\Tomcat 7.0\conf\Catalina\localhost\ig2-xc.xml
Sep 22, 2014 11:57:53 AM org.apache.catalina.startup.HostConfig deployDescriptor
Information: Deploying configuration descriptor C:\Program Files\Apache Software Foundation\Tomcat 7.0\conf\Catalina\localhost\is-elo.xml
Sep 22, 2014 11:58:03 AM org.apache.catalina.startup.HostConfig deployDescriptor
Information: Deploying configuration descriptor C:\Program Files\Apache Software Foundation\Tomcat 7.0\conf\Catalina\localhost\is-QS_AC8.xml
Sep 22, 2014 11:58:18 AM org.apache.catalina.startup.HostConfig deployDescriptor
Information: Deploying configuration descriptor C:\Program Files\Apache Software Foundation\Tomcat 7.0\conf\Catalina\localhost\is-xc.xml
log4j:WARN No appenders could be found for logger Gde.elo.ix.client.webapp.servlet.IXClientServletHandler.
log4j:WARN Please initialize the log4j system properly.
Sep 22, 2014 11:58:22 AM org.apache.catalina.startup.HostConfig deployDescriptor
Information: Deploying configuration descriptor C:\Program Files\Apache Software Foundation\Tomcat 7.0\conf\Catalina\localhost\ix-elo.xml
Sep 22, 2014 11:58:22 AM org.apache.catalina.startup.ContextConfig validateSecurityRoles
Information: WARNING: Security role name manager-gui used in an <auth-constraint> without being defined in a <security-role>
Sep 22, 2014 11:58:29 AM com.sun.xml.rpc.server.http.JAXRPCContextListener contextInitialized
Information: JAXRPCSERULET12: JAX-RPC-Kontextzielgerät wird initialisiert
Sep 22, 2014 11:58:29 AM com.sun.xml.rpc.server.http.JAXRPCServletDelegate init
Information: JAXRPCSERULET14: JAX-RPC-Servlet wird initialisiert
Sep 22, 2014 11:58:30 AM org.apache.catalina.startup.HostConfig deployDescriptor
Information: Deploying configuration descriptor C:\Program Files\Apache Software Foundation\Tomcat 7.0\conf\Catalina\localhost\ix-ELOPreview.xml
Sep 22, 2014 11:58:32 AM org.apache.catalina.startup.ContextConfig validateSecurityRoles
Information: WARNING: Security role name manager-gui used in an <auth-constraint> without being defined in a <security-role>
Sep 22, 2014 11:58:32 AM com.sun.xml.rpc.server.http.JAXRPCContextListener contextInitialized
Information: JAXRPCSERULET12: JAX-RPC-Kontextzielgerät wird initialisiert
Sep 22, 2014 11:58:33 AM com.sun.xml.rpc.server.http.JAXRPCServletDelegate init
Information: JAXRPCSERULET14: JAX-RPC-Servlet wird initialisiert
Sep 22, 2014 11:58:33 AM org.apache.catalina.startup.HostConfig deployDescriptor
Information: Deploying configuration descriptor C:\Program Files\Apache Software Foundation\Tomcat 7.0\conf\Catalina\localhost\ix-import.xml
Sep 22, 2014 11:58:36 AM org.apache.catalina.startup.ContextConfig validateSecurityRoles
Information: WARNING: Security role name manager-gui used in an <auth-constraint> without being defined in a <security-role>
Sep 22, 2014 11:58:36 AM com.sun.xml.rpc.server.http.JAXRPCContextListener contextInitialized
Information: JAXRPCSERULET12: JAX-RPC-Kontextzielgerät wird initialisiert
Sep 22, 2014 11:58:36 AM com.sun.xml.rpc.server.http.JAXRPCServletDelegate init
Information: JAXRPCSERULET14: JAX-RPC-Servlet wird initialisiert
ixid=svtelo8denoun (host name)
Sep 22, 2014 11:58:36 AM org.apache.catalina.startup.HostConfig deployDescriptor
Information: Deploying configuration descriptor C:\Program Files\Apache Software Foundation\Tomcat 7.0\conf\Catalina\localhost\ix-QS_AC8.xml
Sep 22, 2014 11:58:38 AM org.apache.catalina.startup.ContextConfig validateSecurityRoles
Information: WARNING: Security role name manager-gui used in an <auth-constraint> without being defined in a <security-role>
Sep 22, 2014 11:58:38 AM com.sun.xml.rpc.server.http.JAXRPCContextListener contextInitialized
Information: JAXRPCSERULET12: JAX-RPC-Kontextzielgerät wird initialisiert
Sep 22, 2014 11:58:39 AM com.sun.xml.rpc.server.http.JAXRPCServletDelegate init
Information: JAXRPCSERULET14: JAX-RPC-Servlet wird initialisiert
Sep 22, 2014 11:58:39 AM org.apache.catalina.startup.HostConfig deployDescriptor
Information: Deploying configuration descriptor C:\Program Files\Apache Software Foundation\Tomcat 7.0\conf\Catalina\localhost\ix-xc.xml
Sep 22, 2014 11:58:42 AM org.apache.catalina.startup.ContextConfig validateSecurityRoles
Information: WARNING: Security role name manager-gui used in an <auth-constraint> without being defined in a <security-role>
Sep 22, 2014 11:58:42 AM com.sun.xml.rpc.server.http.JAXRPCContextListener contextInitialized
Information: JAXRPCSERULET12: JAX-RPC-Kontextzielgerät wird initialisiert
Sep 22, 2014 11:58:42 AM com.sun.xml.rpc.server.http.JAXRPCServletDelegate init
Information: JAXRPCSERULET14: JAX-RPC-Servlet wird initialisiert
Sep 22, 2014 11:58:42 AM org.apache.catalina.startup.HostConfig deployDescriptor
Information: Deploying configuration descriptor C:\Program Files\Apache Software Foundation\Tomcat 7.0\conf\Catalina\localhost\sx-elo.xml
Sep 22, 2014 11:58:51 AM org.apache.catalina.startup.ContextConfig validateSecurityRoles
Information: WARNING: Security role name manager-gui used in an <auth-constraint> without being defined in a <security-role>
Sep 22, 2014 11:58:51 AM org.apache.catalina.startup.HostConfig deployDescriptor
Information: Deploying configuration descriptor C:\Program Files\Apache Software Foundation\Tomcat 7.0\conf\Catalina\localhost\sx-QS_AC8.xml
Sep 22, 2014 11:58:59 AM org.apache.catalina.startup.ContextConfig validateSecurityRoles
Information: WARNING: Security role name manager-gui used in an <auth-constraint> without being defined in a <security-role>
Sep 22, 2014 11:59:00 AM org.apache.catalina.startup.HostConfig deployDescriptor
Information: Deploying configuration descriptor C:\Program Files\Apache Software Foundation\Tomcat 7.0\conf\Catalina\localhost\sx-xc.xml
Sep 22, 2014 11:59:17 AM org.apache.catalina.startup.ContextConfig validateSecurityRoles

```

Als weiterer Vorteil ergibt sich, dass Meldungen zum Start der „Engine“ direkt in der Konsole ausgegeben werden. Der Dienst „ELO Server Engine“ ist vorher zu stoppen, da es sonst zum „Portkonflikt“ kommt, da der Dienst die Ports bereits verwendet, die durch den „catalina start“ erneut benutzt werden sollen.

## 4 Java Developer Kit

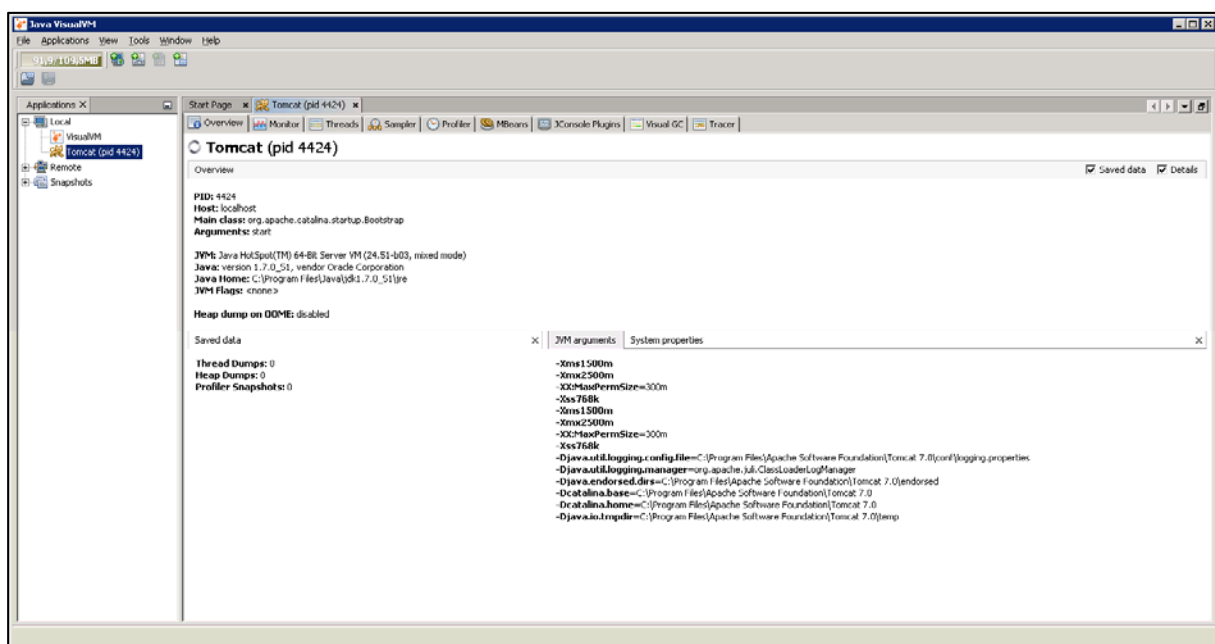
Benötigt wird ein aktuelles JDK Paket.

- 1) Download von <http://www.oracle.com/technetwork/java/javase/downloads/index.html>
- 2) Es können auf einem Server mehrere JRE bzw. JDK Pakete installiert sein. Gegebenenfalls ist auf die Anpassung der „JAVA\_HOME“ Umgebungsvariable zu achten.
- 3) Installation des JDK z.B. über die „setup.exe“. Eine Installation unter „Program Files“ ist unter Windows sinnvoll, kann aber auch individuell gesetzt werden.

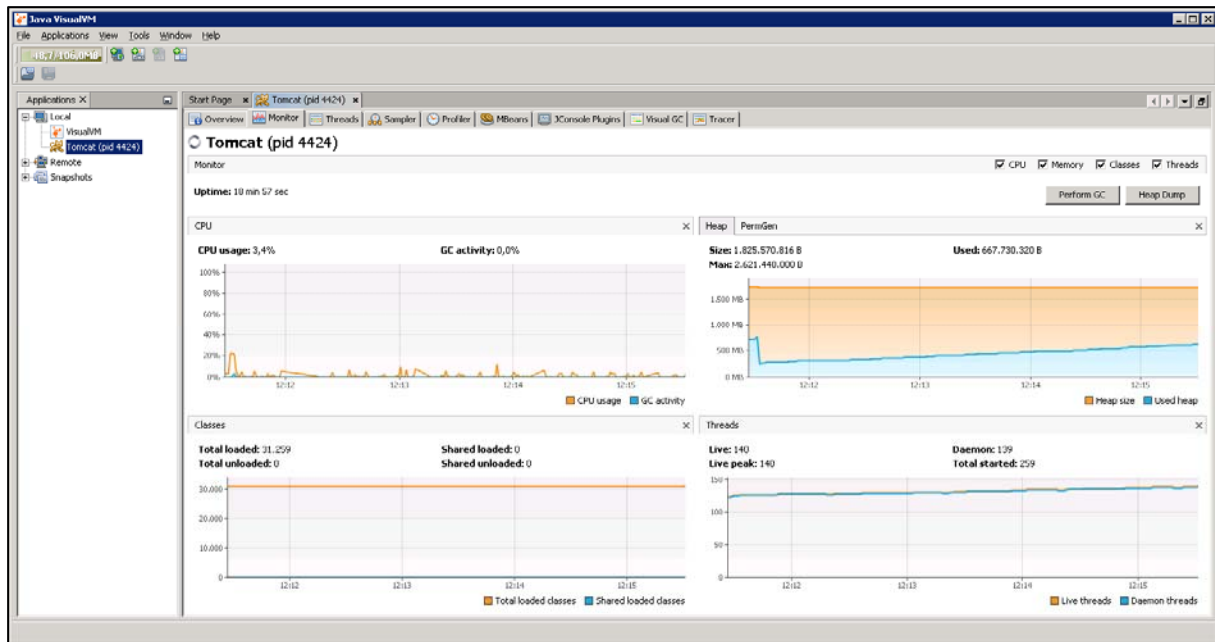
## 5 Monitoring

Monitoring mit der „jvisualvm.exe“.

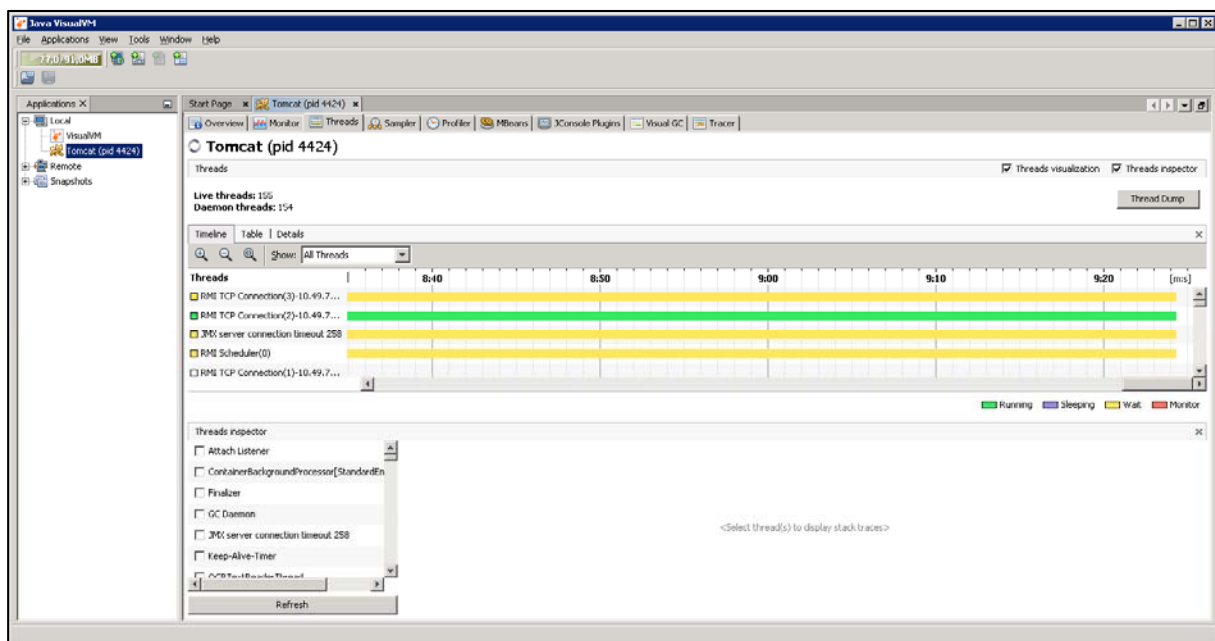
- 1) Anzeige der Startparameter und des Monitors.  
Im JDK Paket befindet sich das Programm „jvisualvm.exe“. Per default Installation liegt die Datei unter „C:\Program Files\Java\jdk1.7.0\_51\bin“. Die „jvisualvm.exe“ führt beim ersten Start eine Kalibrierung des Java Systems durch. Dies ist bei allen Messtechniken üblich. Wird die **ELO** Server Engine über die „catalina.bat“ gestartet, erkennt das Programm die „Tomcat PID“ und zeigt diese direkt an. Per Rechtsklick kann dann mit „Open“ der Monitor für diese Tomcat PID gestartet werden. Zunächst werden die System Informationen und die JVM arguments (Java Startparameter) angezeigt:



Im Monitor Bereich werden die CPU Verwendung, die Speicher Allokation, die Anzahl der geladenen Java Classes und die Anzahl der verwendeten Threads graphisch angezeigt:



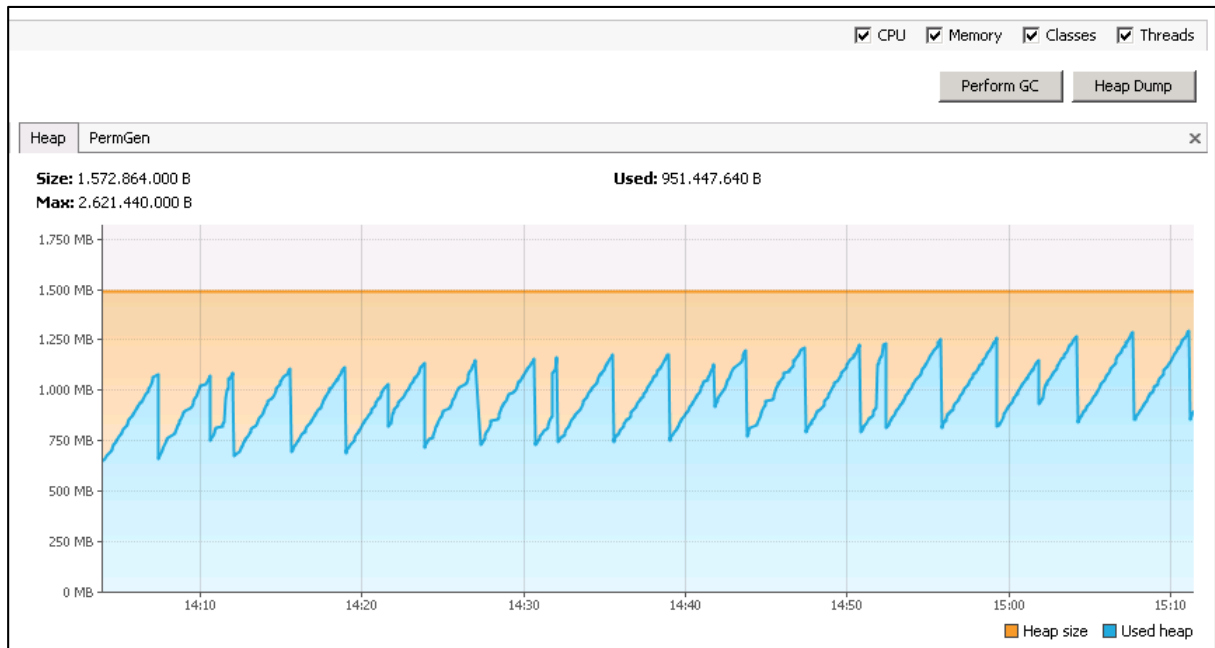
- 2) Erweitertes Monitoring (Threads, Profiler, MBeans, JConsole Plugins, Visual GC, Tracer)  
Die Details der Threads können über den Threads Bereich angezeigt werden:



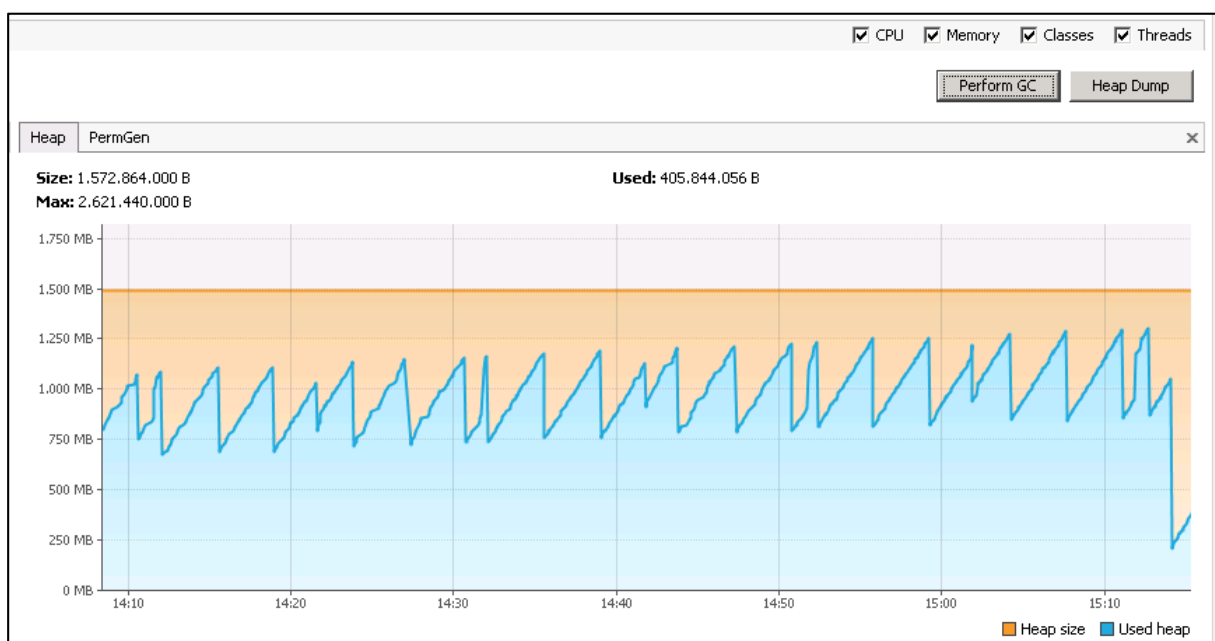
## 6 Basis Monitoring Optionen

Monitor Heap Memory

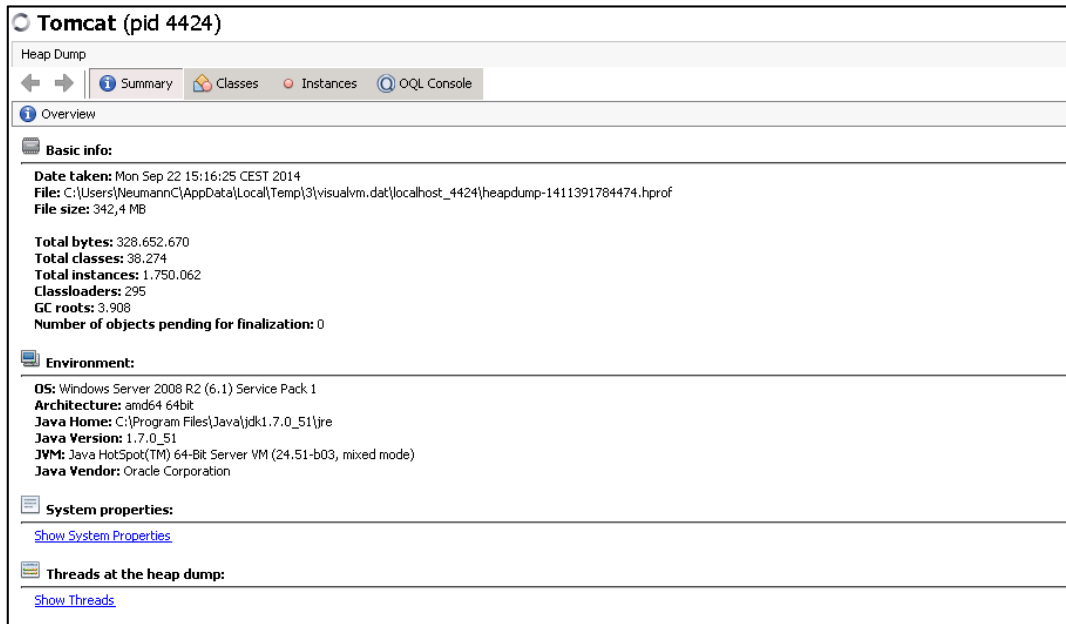
- 1) Tomcat verwendet die Java Garbage Collection. Entsprechend wird nach einer GC wieder Speicher frei und zur Verfügung gestellt. Deshalb sollte sich die graphische Darstellung nach mehreren GC Prozessen immer wie in folgendem Screenshot gezeigt darstellen:



- 2) Es stehen zwei Optionen zur Verfügung: Perform GC wird nicht mehr benötigten Speicher frei machen und entsprechend wird der „blau Graph“ nach Ausführung auf einen niedrigeren Wert fallen:



- 3) Heap Dump zeichnet die zu diesem Zeitpunkt aktuellen Speicherwerte in ein Dump File und lädt im Anschluss diese Heap Dump Aufzeichnung:



**Tomcat (pid 4424)**

Heap Dump

Summary Classes Instances OQL Console

Overview

**Basic info:**

Date taken: Mon Sep 22 15:16:25 CEST 2014  
 File: C:\Users\NeumannC\AppData\Local\Temp\3\visualvm.dat\localhost\_4424\heapdump-1411391784474.hprof  
 File size: 342,4 MB

Total bytes: 328.652.670  
 Total classes: 38.274  
 Total instances: 1.750.062  
 Classloaders: 295  
 GC roots: 3.908  
 Number of objects pending for finalization: 0

**Environment:**

OS: Windows Server 2008 R2 (6.1) Service Pack 1  
 Architecture: amd64 64bit  
 Java Home: C:\Program Files\Java\jdk1.7.0\_51\jre  
 Java Version: 1.7.0\_51  
 JVM: Java HotSpot(TM) 64-Bit Server VM (24.51-b03, mixed mode)  
 Java Vendor: Oracle Corporation

**System properties:**

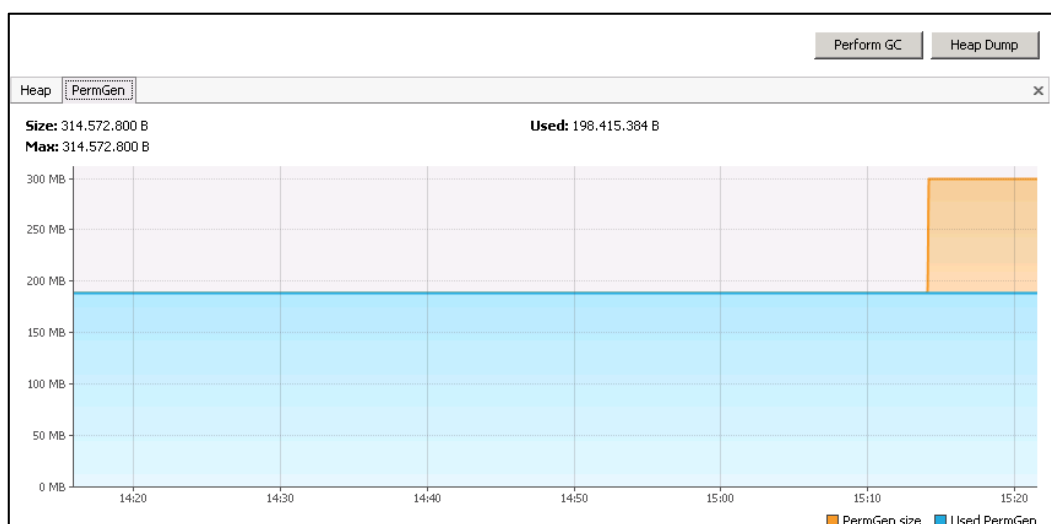
[Show System Properties](#)

**Threads at the heap dump:**

[Show Threads](#)

Über die einzelnen Optionen ist dann eine erweiterte Analyse des Speicherhaltens möglich.

- 4) PermGen Bereich. In diesem werden den im Tomcat verwalteten Applikationen Klassen Speicher zur Verfügung gestellt. Beim Laden der Applikation werden die Klassen der jeweiligen Applikation in diesen Speicherbereich geschrieben. Da dies üblicherweise nur beim Start des Tomcat geschieht, ergibt sich ein statischer Verlauf des Graph:



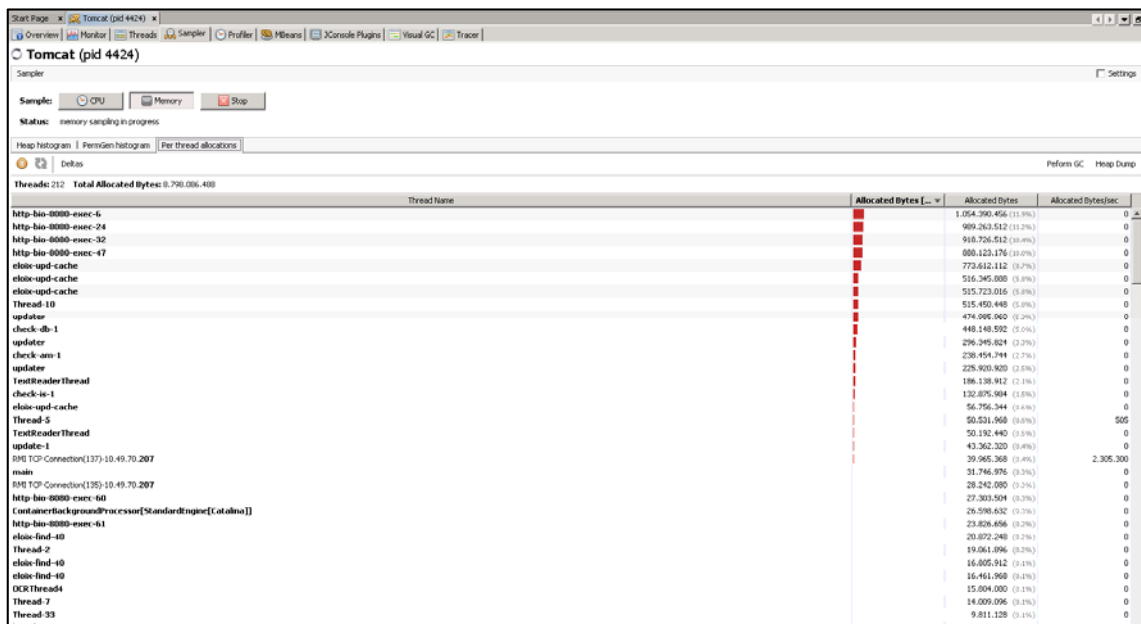
Die Grafik zeigt den gerade verwendeten Speicher in blauer und den maximalen Speicher (MaxPermSize Parameter) in oranger Farbe an. Werden nachträglich neue Applikation im laufenden Tomcat zur Verfügung gestellt, wird der Speicherbedarf, genau um den Wert der Klassen dieser Applikation zunehmen. Ein „Reload“ einer Applikation wird ebenso den Speicherbedarf um genau den Wert der Klassen dieser Applikation erhöhen.



## 7 Erweiterte Monitor Optionen

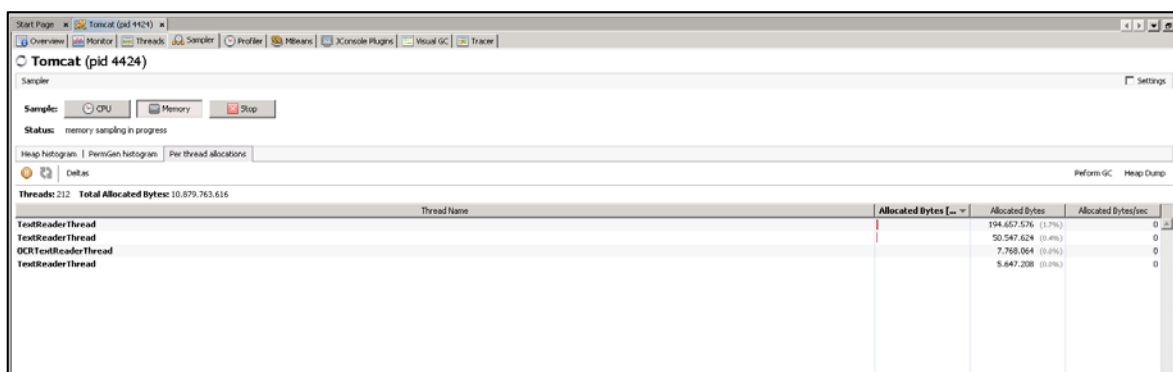
### 1) Sampler Analyse

Im Sampler Bereich können die CPU und der Memory genauer untersucht werden. Dazu wird der jeweilige Button verwendet. Im Memory Bereich werden drei weitere Betrachtungsweisen zur Verfügung gestellt. Häufig liefert die Option „Per thread allocations“ am schnellsten Hinweise darauf, welcher Thread tatsächlich den meisten Speicherbedarf hat:



Thread Name	Allocated Bytes	Allocated Bytes (sec)
http-bio-0000-exec-6	1,054,390,456 (11.3%)	0
http-bio-0000-exec-24	909,263,512 (11.3%)	0
http-bio-0000-exec-32	918,726,512 (11.4%)	0
http-bio-0000-exec-47	880,123,176 (11.0%)	0
elcix-upd-cache	773,612,112 (9.7%)	0
elcix-upd-cache	516,345,008 (5.8%)	0
elcix-upd-cache	515,723,016 (5.8%)	0
Thread-10	515,450,448 (5.8%)	0
updater	474,095,060 (5.3%)	0
check-db-1	448,148,592 (5.0%)	0
updater	296,345,624 (3.3%)	0
check-am-1	238,454,744 (2.7%)	0
updater	225,920,920 (2.6%)	0
TextReaderThread	186,138,912 (2.1%)	0
check-is-1	132,875,504 (1.5%)	0
elcix-upd-cache	56,756,344 (0.6%)	0
Thread-5	50,531,968 (0.6%)	505
TextReaderThread	50,192,440 (0.6%)	0
update-1	43,362,320 (0.5%)	0
RMI TCP Connection(137)-10.49.70.207	39,065,368 (0.4%)	2,305,300
main	31,766,976 (0.4%)	0
RMI TCP Connection(135)-10.49.70.207	28,242,080 (0.3%)	0
http-bio-0000-exec-60	27,303,504 (0.3%)	0
ContainerBackgroundProcessor[StandardEngine[Catalina]]	26,598,632 (0.3%)	0
http-bio-0000-exec-61	23,826,856 (0.3%)	0
Thread-40	20,872,240 (0.2%)	0
Thread-2	19,561,896 (0.2%)	0
elcix-find-40	16,605,912 (0.2%)	0
elcix-find-40	16,461,968 (0.2%)	0
OCRTThread4	15,804,000 (0.2%)	0
Thread-7	14,009,096 (0.2%)	0
Thread-33	9,811,128 (0.1%)	0
...	...	...

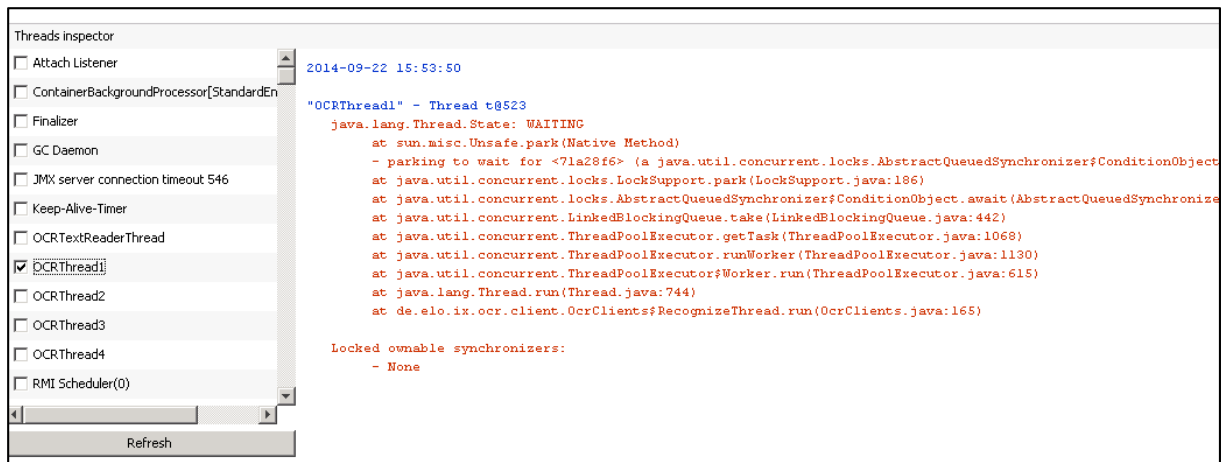
Die tabellarische Darstellung listet in absteigender Reihenfolge den Speicherbedarf der Threads auf. Die Abbildung oben zeigt, dass die http connector Threads den größten Bedarf bei dieser Aufzeichnung hatten. So stellt sich üblicherweise das „normale Verhalten“ dar. Weiter unten sind die **ELO** spezifischen Threads gelistet. Insbesondere fallen die „elcix-upd-cache“ Threads auf, die für update und cache der ELOix Funktionen verantwortlich sind. Weitere **ELO** Threads sind „check-db-1“ und „check-am-1“. Diese überwachen die Datenbank Verbindungen und starten diese bei Bedarf neu. Auch der **ELO** „TextReaderThread“ und der „OCRTThread4“ sind in der Liste eingetragen. Haben Threads einen besonders hohen Speicherbedarf, werden diese in dieser Überwachung sehr schnell durch ihr hohes „Ranking“ auffallen. Über die untere „Filterzeile“ können schnell die unter Verdacht stehenden Threads herausgefiltert werden:



Thread Name	Allocated Bytes	Allocated Bytes (sec)
TextReaderThread	194,667,576 (1.7%)	0
TextReaderThread	50,547,624 (0.6%)	0
OCRTThread4	7,768,064 (0.0%)	0
TextReaderThread	5,647,208 (0.0%)	0

**2)** Thread inspector zur weiteren Prüfung

Über diese Sampler Analyse können meist auch schnell ungünstig formulierte **ELOas** Regeln identifiziert werden. Wenn der Thread Name bekannt ist, kann über das zusätzliche Plugin „Thread inspector“ auch der Zustand des Threads angezeigt werden. Hier ein Beispiel für den „OCRThread1“, welches analog auch für **ELOas** Threads gilt:

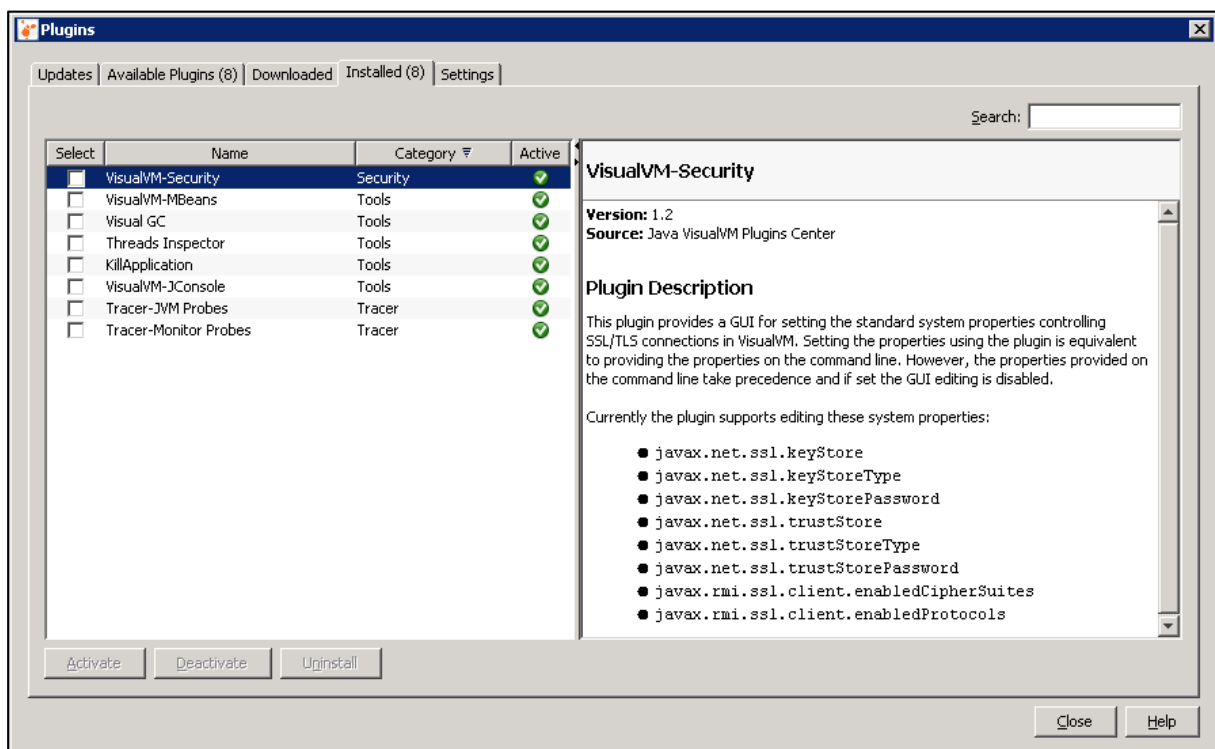


## 8 Plugins installieren

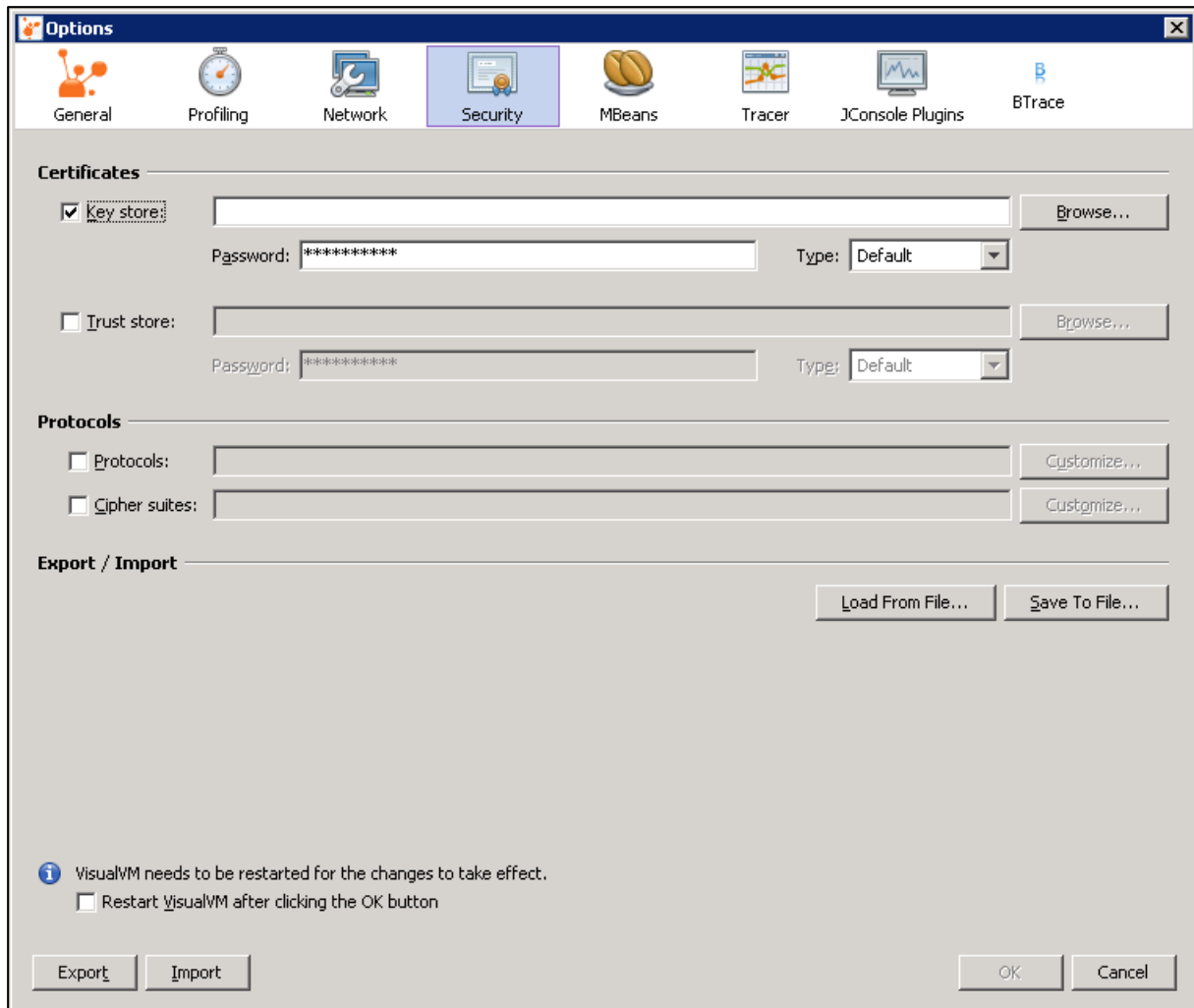
Zusätzliche Plugins für erweitertes Monitoring. Zusätzlich zu den Standard Anzeigen können weitere Plugins eingebunden werden. Unter Menü „Tools\Plugins“ werden die verfügbaren Plugins angezeigt. Weitere Plugins können per Download und Install in der JVisualVM verwendet werden. Diese Plugins sind häufig hilfreich:

Über die Standard Plugins sind per Download weitere Plugins verfügbar. Folgende Plugins liefern weitere wichtige Informationen:

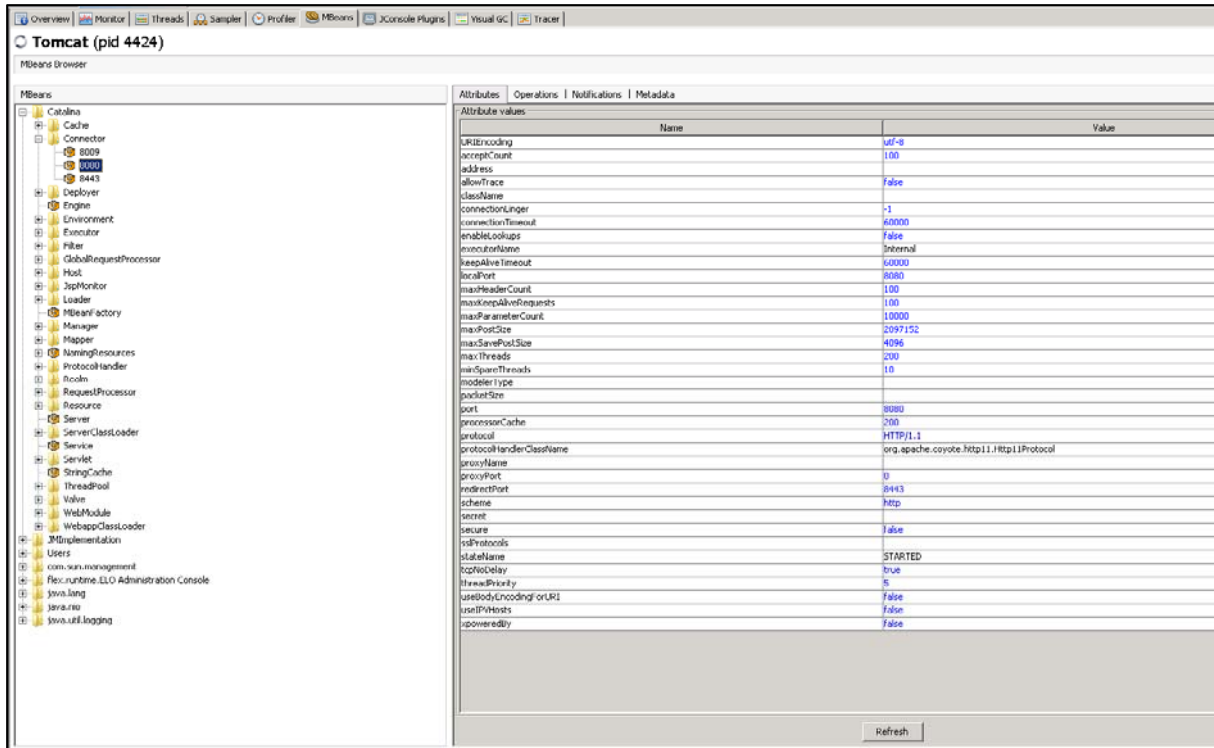
- VisualVM-Security zur Administration einer SSL Anbindung
- VisualVM-Beans zur Anzeige aller Tomcat Parameter
- Thread Inspector zur Anzeige der Thread Aufrufe
- KillApplication, zum „harten Stopp“ des Tomcats
- VisualVM-JConsole z.B. zum Anzeigen der Top CPU Anwendungen
- Visual GC zeigt die Garbage Collection in den Bereichen an
- Tracer-JVM Probes zur längerfristigen Überwachung spezieller Parameter
- Tracer-Monitor Probes zur längerfristigen Überwachung spezieller Parameter



- 1) VisualVM-Security dient dazu, SSL Verbindungen aufzusetzen und die Passwörter für den Key Store zu verwalten. Weiterhin stehen hier Export / Import Funktionen zur Verfügung:

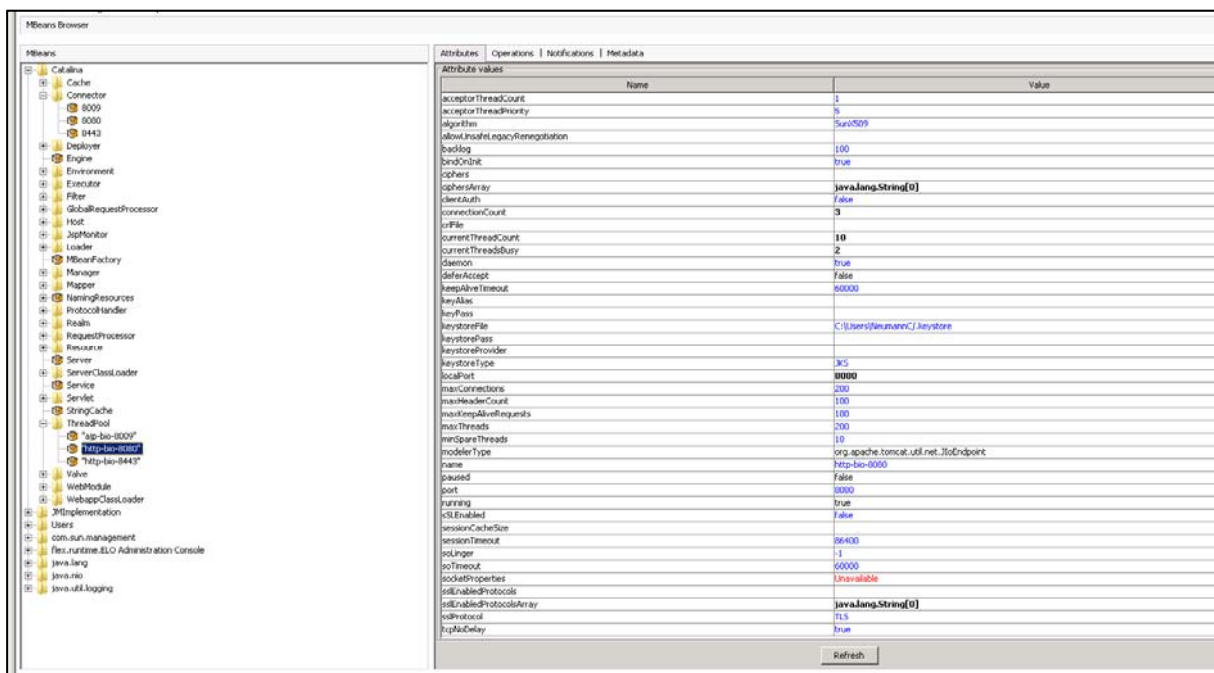


- 2) VisualVM-Beans zeigt alle Tomcat- und Java Parameter an. Insbesondere sind für das Netzwerk, vielfältige Einstellungen möglich. Allein für die Konfiguration des „Connector“ Ports, sind diese Settings verfügbar:



Name	Value
URLEncoding	utf-8
acceptCount	100
address	
allowTrace	false
className	
connectionLinger	-1
connectionTimeout	60000
enableLookups	false
executorName	Internal
keepAliveTimeout	60000
localPort	8080
maxHeaderCount	100
maxKeepAliveRequests	100
maxParameterCount	10000
maxPostSize	2097152
maxSavePostSize	4096
maxThreads	200
maxSwingThreads	10
modellerType	
packetSize	
port	8080
processorCache	200
protocol	HTTP/1.1
protocolHandlerClassName	org.apache.coyote.http11.Http11Protocol
proxyName	
proxyPort	0
redirectPort	8443
scheme	http
secure	false
sslProtocols	
stateName	STARTED
tcpNoDelay	true
threadPriority	5
useBodyEncodingForURI	false
useIPV6	false
upgradedBy	false

Ebenso können die Thread-Pools vielfältig konfiguriert werden:



Name	Value
acceptorThreadCount	1
acceptorThreadPriority	5
algorithm	Surrogate
allowProtocolUpgradeRenegotiation	
backlog	100
bindOnInit	true
ciphers	
ciphersArray	javax.lang.String(0)
clientAuth	false
connectionCount	3
crFile	
currentThreadCount	10
currentThreadBusy	2
daemon	true
deferAccept	false
keepAliveTimeout	60000
keyAlias	
keyPass	
keystoreFile	C:\Users\jbaumann\keystore
keystorePass	
keystoreProvider	
keystoreType	JKS
localPort	8080
maxConnections	200
maxHeaderCount	100
maxKeepAliveRequests	100
maxThreads	200
minSpareThreads	10
modellerType	org.apache.tomcat.util.net.NioEndpoint
name	http-bio-8080
paused	false
port	8080
running	true
sslEnabled	false
sessionCacheSize	26400
sessionTimeout	30
socket	
socketTimeout	60000
socketProperties	Unavailable
sslEnabledProtocols	
sslEnabledProtocolsArray	javax.lang.String(0)
sslProtocol	TLS
tcpNoDelay	true

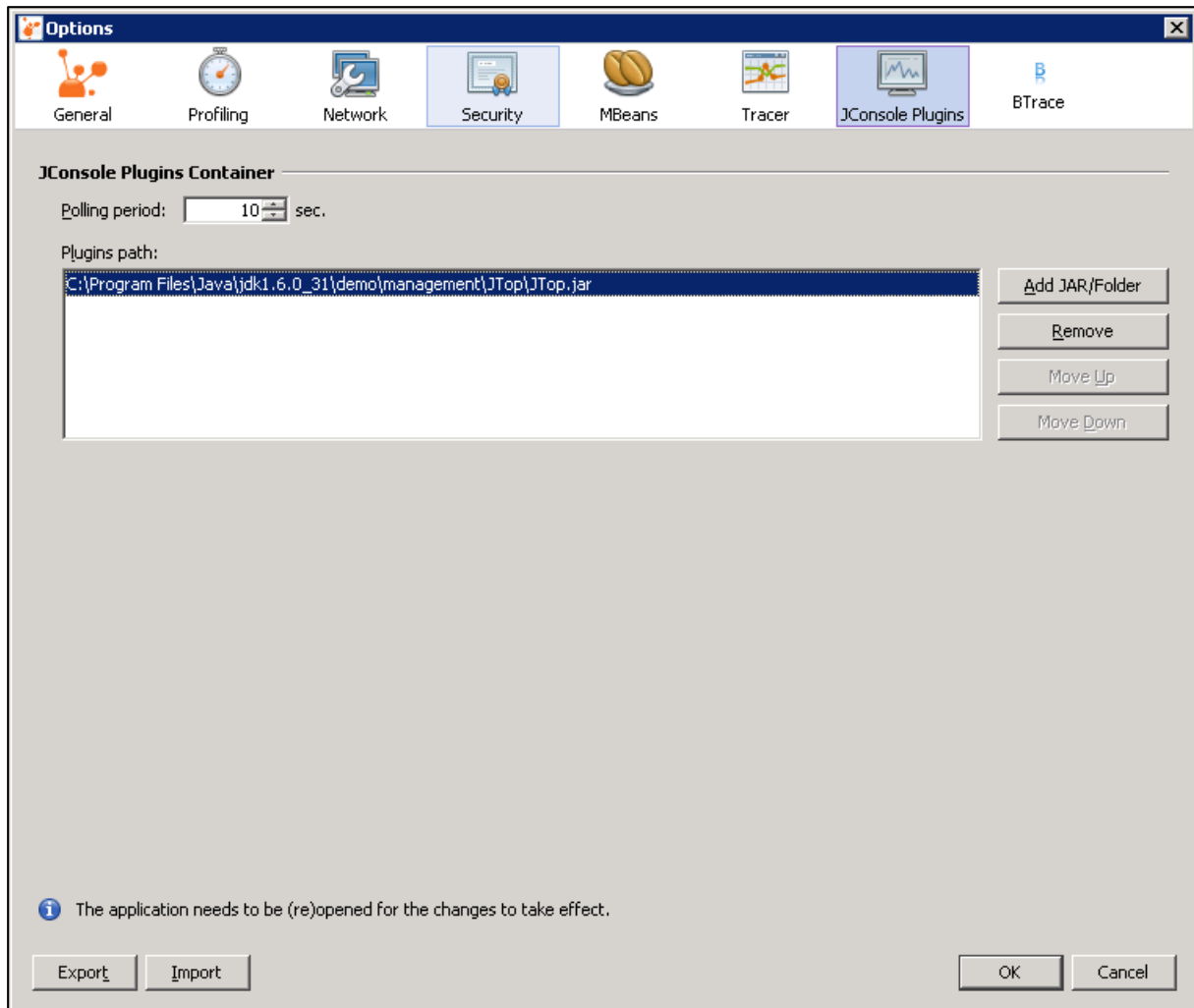
Bisher waren insbesondere die Parameter „maxThreads“ und „maxConnections“ in größeren **ELO** Umgebungen nach oben anzupassen.

Die VisualVM-Beans Anzeigen stellen die Informationen zur Verfügung, wie der Tomcat im Detail eingestellt werden kann und sind deshalb eine wichtige Informationsquelle zum Erstellen von Optimierungsansätzen. Diese Informationen können als **ELO** Tomcat Basis Dokumentation genutzt werden.

- 3) VisualVM-Jconsole Plugins. In Verbindung mit den Java Demo Libraries können weitere Plugins verwendet werden. Für einen ersten Überblick über die CPU Nutzer der **ELO** Anwendungen ist das Tool „JTop“ hilfreich. Dieses zeigt in einer Tabelle die CPU (sec) der Anwendungen in einer Ranking Liste an:

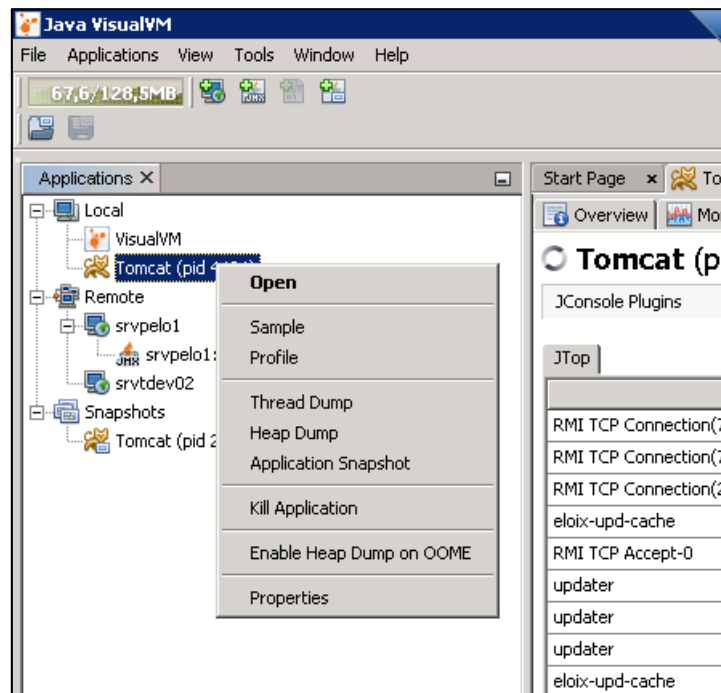
Tomcat (pid 4424)			
JConsole Plugins			
JTop			
ThreadName	CPU(sec)		State
RMI TCP Connection(712)-10.49.70.207	65,0000		RUNNABLE
RMI TCP Connection(267)-10.49.70.207	44,0000		TIMED_WAITING
RMI TCP Connection(709)-10.49.70.207	40,0000		RUNNABLE
eloi-upd-cache	21,0000		TIMED_WAITING
RMI TCP Accept-0	14,0000		RUNNABLE
updater	11,0000		TIMED_WAITING
updater	10,0000		TIMED_WAITING
updater	10,0000		TIMED_WAITING
eloi-upd-cache	9,0000		TIMED_WAITING
TextReaderThread	7,0000		TIMED_WAITING
http-bio-8080-Acceptor-0	6,0000		RUNNABLE
check-db-2	6,0000		TIMED_WAITING
Thread-153	4,0000		TIMED_WAITING
eloi-find-40	3,0000		WAITING
TextReaderThread	2,0000		TIMED_WAITING
eloi-upd-cache	2,0000		TIMED_WAITING
Thread-33	1,0000		TIMED_WAITING
TextReaderThread	1,0000		TIMED_WAITING
OCRTextReaderThread	1,0000		TIMED_WAITING
Thread-5	1,0000		TIMED_WAITING
check-is-2	1,0000		TIMED_WAITING
ContainerBackgroundProcessor[StandardEngine[Catalina]]	1,0000		TIMED_WAITING
Finalizer	1,0000		WAITING
main	1,0000		RUNNABLE
http-bio-8080-exec-265	1,0000		TIMED_WAITING
http-bio-8080-exec-162	1,0000		TIMED_WAITING
http-bio-8080-exec-213	1,0000		RUNNABLE
http-bio-8080-exec-246	1,0000		TIMED_WAITING
http-bio-8080-exec-131	1,0000		RUNNABLE
http-bio-8080-exec-214	1,0000		TIMED_WAITING
http-bio-8080-exec-257	1,0000		TIMED_WAITING
http-bio-8080-exec-256	0,0000		TIMED_WAITING
Thread-34	0,0000		TIMED_WAITING
eloi-alive	0,0000		TIMED_WAITING
http-bio-8080-exec-258	0,0000		TIMED_WAITING
eloi-alive	0,0000		TIMED_WAITING
OCRThread4	0,0000		WAITING

Damit die JVisualVM dies darstellen kann, ist die **ELO** Server Engine über die catalina.bat zu starten und die zu verwendende Java Demo Library anzugeben:

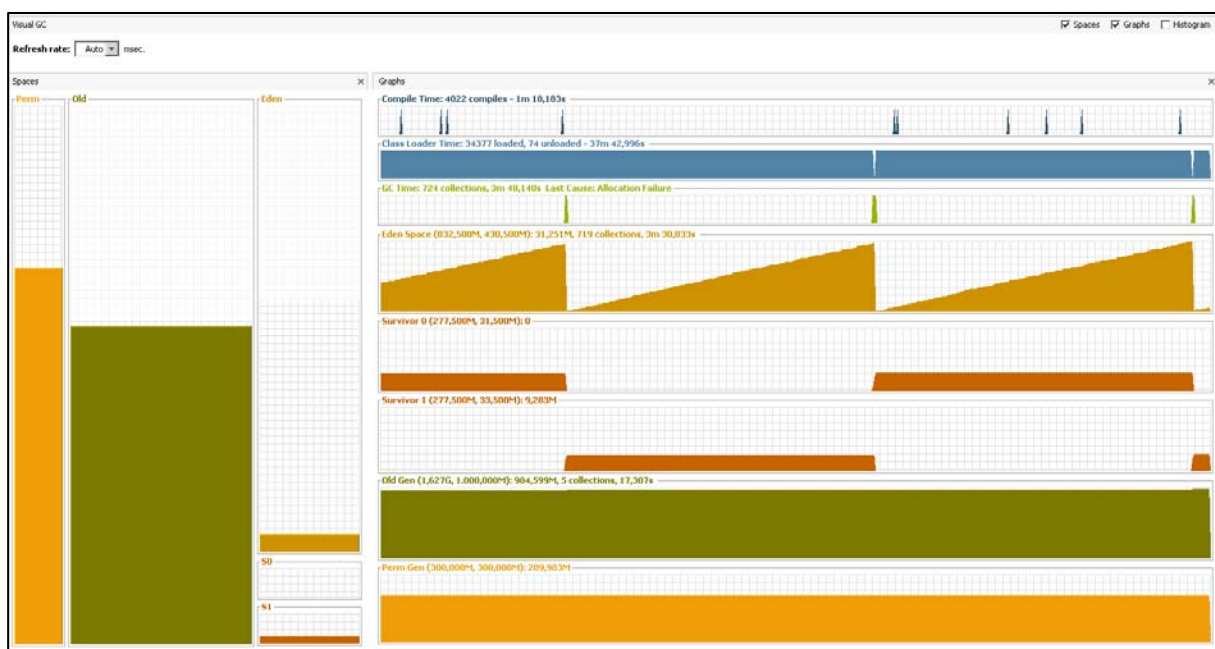


- 4) Thread inspector zeigt die Thread Aufrufe an. Das Tool wurde hier bereits unter 5.2 dargestellt.

- 5) Kill Application beendet den **ELO** Server Engine Prozess hart, wenn dies über das Beenden des Dienstes nicht mehr möglich ist. Die Funktion kann im Kontextmenü aufgerufen werden:

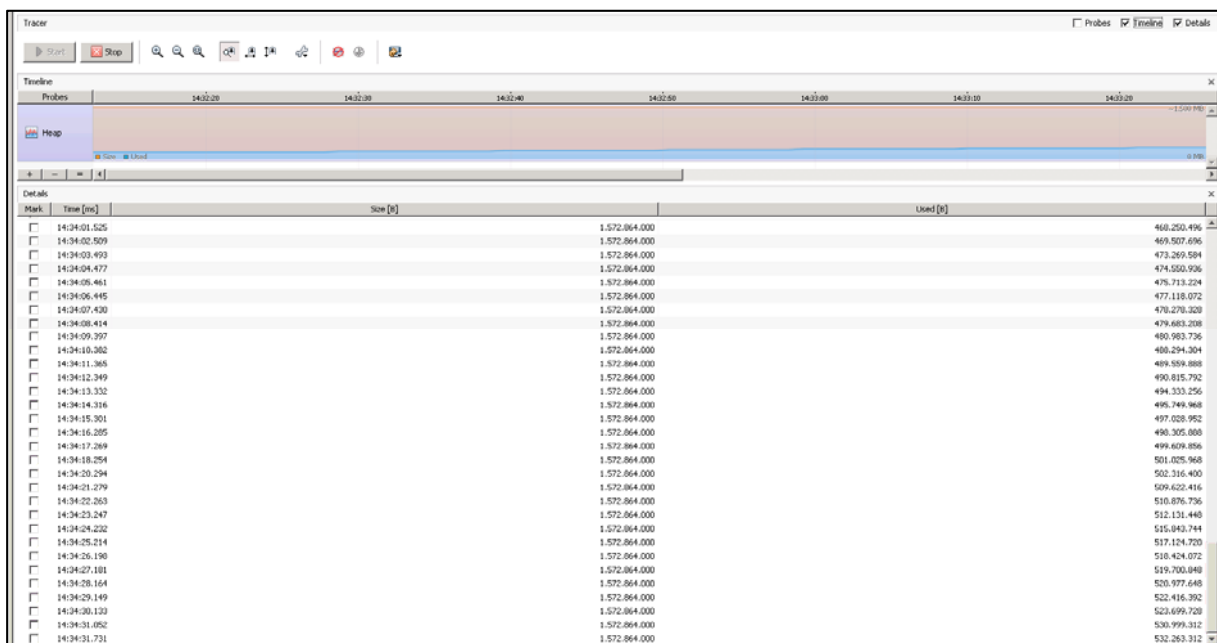


- 6) Visual GC zeigt detailliert an, wie der Prozess der Garbage Collection durchgeführt wird. Die Garbage Collection durchläuft mehrere Bereiche. Insbesondere die Eden-Perm- und Old Spaces können von Bedeutung sein. Damit kann schnell das Speicherverhalten geprüft werden:





- 7) Tracer-JVM Probes zeichnen die Daten für CPU&GC, Heap, PermGen, Classes und Threads auf. Meist sind die Heap Werte besonders wichtig. Die Messung wird über den Start Button begonnen und läuft dann bis zum Beenden über den Stop Button. Die Ergebnisse werden graphisch und tabellarisch angezeigt und können nach der Messung in ein CSV File exportiert werden. Damit ist es auch möglich, Messungen über einen längeren Zeitraum auszuführen und nachträglich auszuwerten:



## 9 Fazit

Die Visualisierung der Operationen in der **ELO** Server Engine sind eine große Hilfe bei der Analyse von Problemen der Basisinfrastrukturen Java und Tomcat. Damit können sowohl die CPU Last als auch die Memory Allokation detailliert visualisiert werden. Anwendungen mit ungewöhnlich hoher Nutzung der CPU oder des Memory können so meist sehr schnell identifiziert werden. Weiterhin kann schnell ermittelt werden, welche Werte für CPU- und Memory-Nutzung üblich sind und entsprechend die Zuweisungen für besondere Aufgaben vorgeplant und dann entsprechend angepasst, durchgeführt werden.

Zudem stellen insbesondere die MBeans Anzeigen wichtige Informationen über die Einstellungsmöglichkeiten zur Verfügung.

Grundsätzlich sollte jedes **ELO** System mit einem Monitor Tool überwacht werden, um im Problemfall schnell die Ursachen identifizieren zu können. Die JVisualVM ist dabei ein sehr effizientes Tool, dass für solche Fälle wertvolle Informationen zur Verfügung stellt und darüber hinaus Informationen liefert, um das **ELO** System noch performanter bezüglich der Basisinfrastruktur einzustellen.

**Copyright**

ELO Digital Office GmbH • Tübinger Straße 43 • 70178 Stuttgart

Alle Rechte, auch die des Nachdrucks, der Vervielfältigung oder der Verwertung bzw. Mitteilung des Inhalts dieses Dokuments oder von Teilen daraus, behalten wir uns vor. Kein Teil darf ohne schriftliche Genehmigung der ELO Digital Office GmbH in irgendeiner Form reproduziert, an Dritte weitergegeben oder insbesondere unter Verwendung elektronischer Systeme verarbeitet, vervielfältigt oder zur öffentlichen Wiedergabe benutzt werden. Wir behalten uns das Recht vor, Inhalte zu aktualisieren oder zu modifizieren.

**Warenzeichen**

ELOenterprise, ELOprofessional und ELOoffice sind eingetragene Warenzeichen der ELO Digital Office GmbH. SAP®, R/2, R/3, ABAP/4, SAP ArchiveLink, SAP Business Workflow, das SAP-Logo und das R/3-Logo sind eingetragene Marken der SAP AG. Microsoft Windows, Microsoft Office, Microsoft Word, Microsoft Excel, Microsoft PowerPoint, Microsoft Outlook und Microsoft SQL Server sind eingetragene Warenzeichen. Andere Produktnamen werden nur zur Identifikation der Produkte verwendet und können eingetragene Warenzeichen / Marken der jeweiligen Hersteller sein.