JMX
Java Management eXtension

Didier DONSEZ

Université Joseph Fourier –Grenoble 1

PolyTech'Grenoble LIG/ADELE

Didier.Donsez@imag.fr,

Didier.Donsez@ieee.org

Outline

- Motivation
- Architecture
- MBeans
  - Standard, Dynamic
  - Monitor
- Notification
- Utilities
- Remote Management
  - Connector & Adaptor
- API
- Tools
- JSR
- Bibliography

Under Translation fr->en

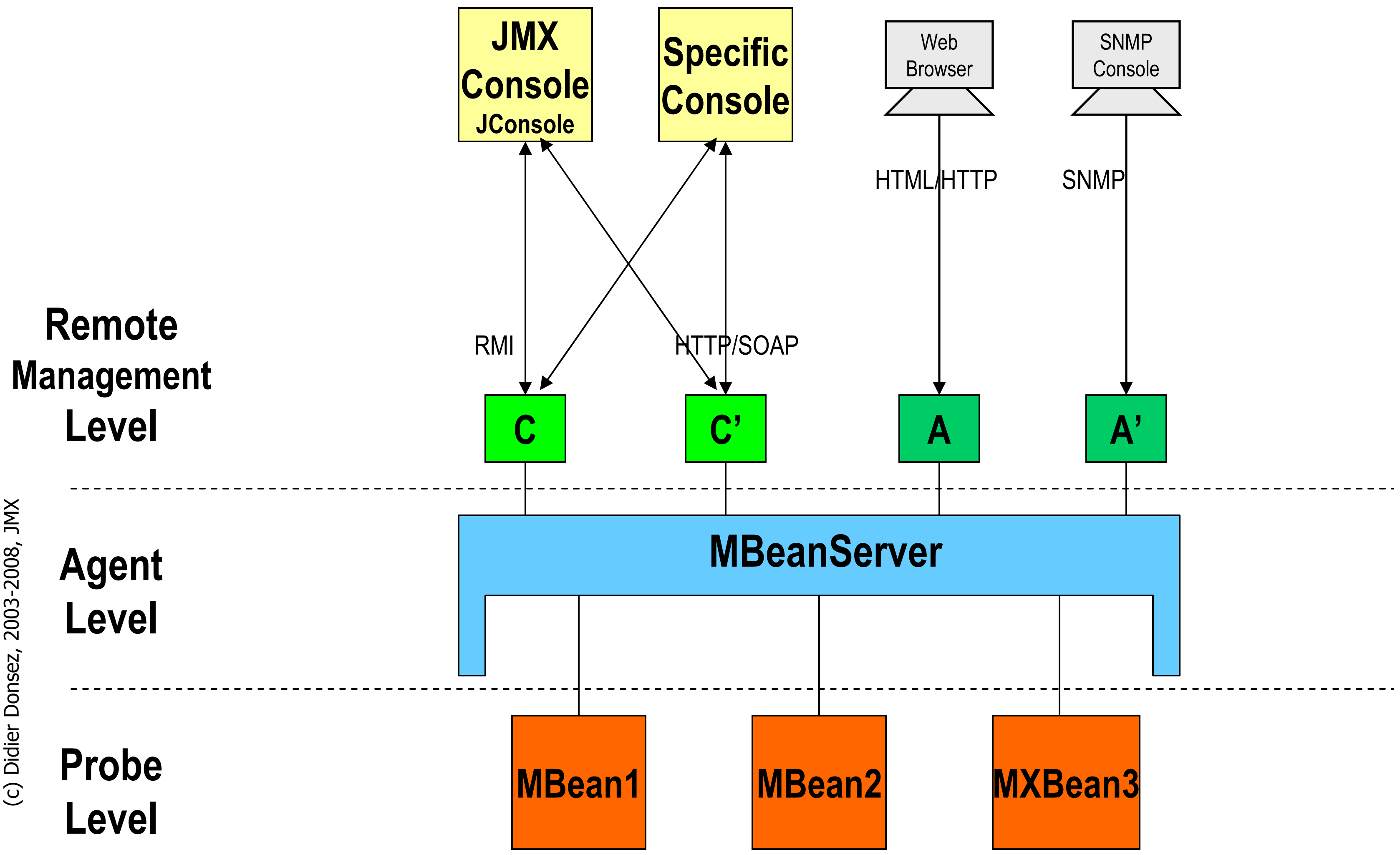
Motivations

- Service d'administration d'applications
  - Ressources matérielles accessibles depuis une JVM
  - Ressources logiciels s'exécutant sur une JVM
  - S'inspire de SNMP
- Instrumentations des ressources au moyen de composants appelés MBean (Manageable Bean)
- Adopté initialement par J2EE
  et maintenant J2SE 1.5

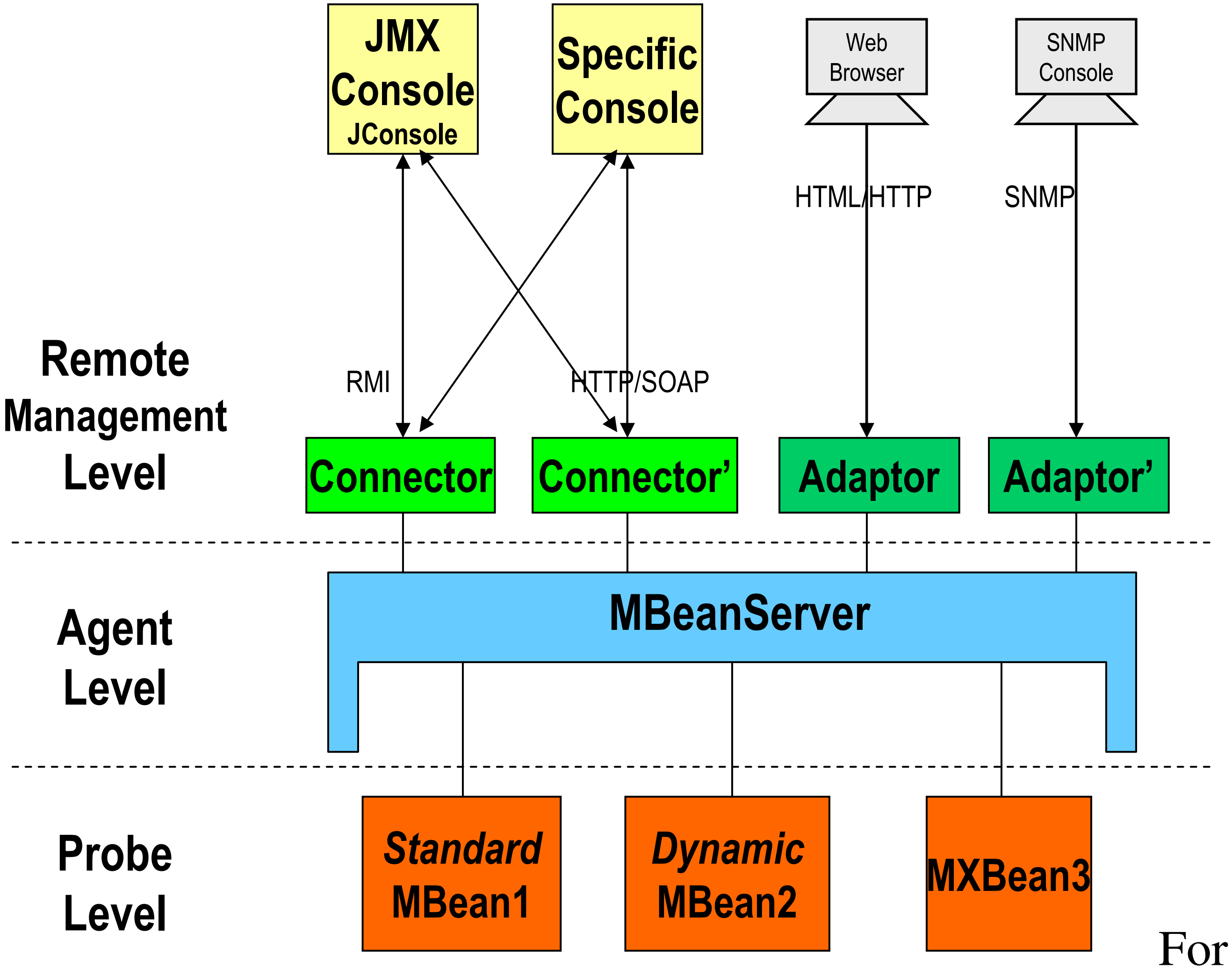
Usage

- Get and set applications configuration (Pull)
- Collect statistics (Pull)
  - Performance
  - Ressources usage
  - Problems
- Notify events (Push)
  - Faults
  - State changes

JMX 3-Level Architecture

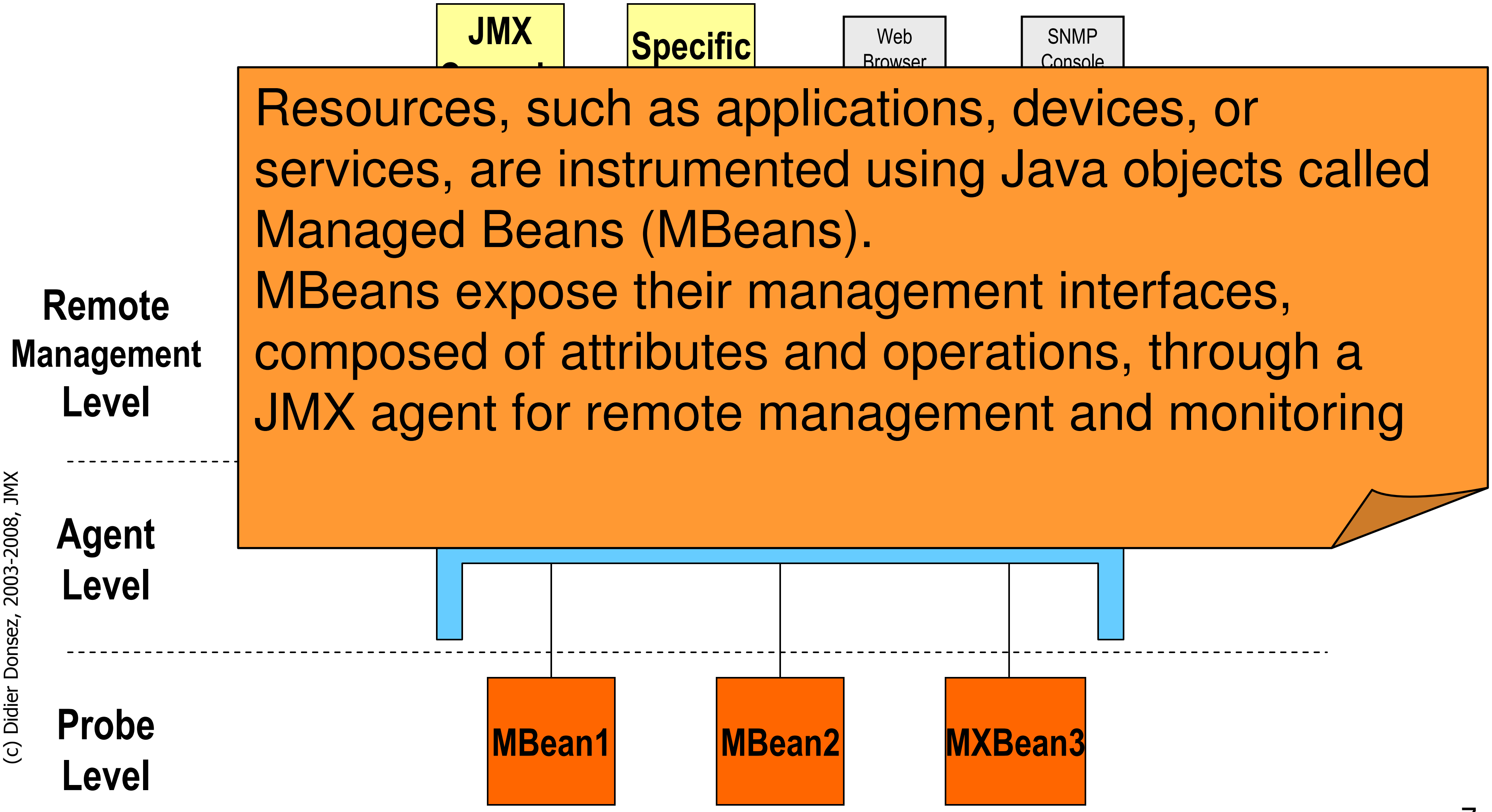


JMX 3-Level Architecture



For Wikipedia

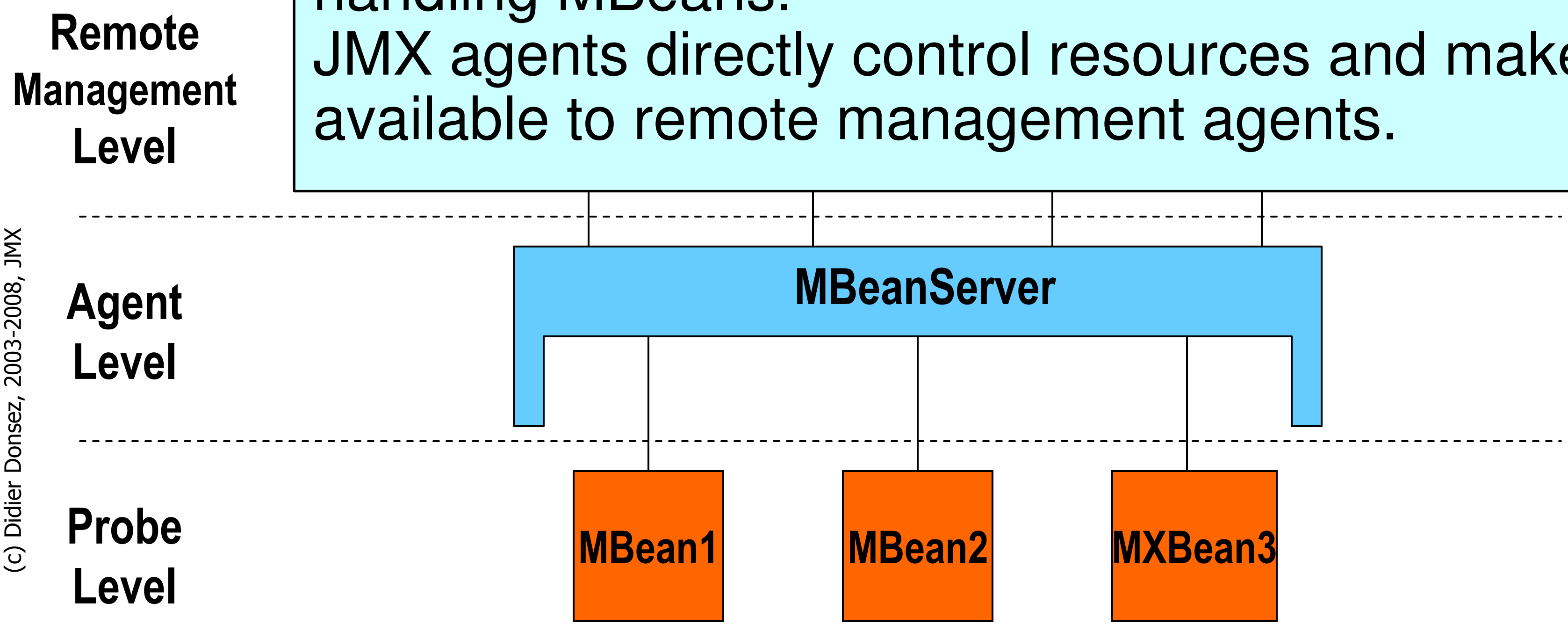
JMX 3-Level Architecture Probe level



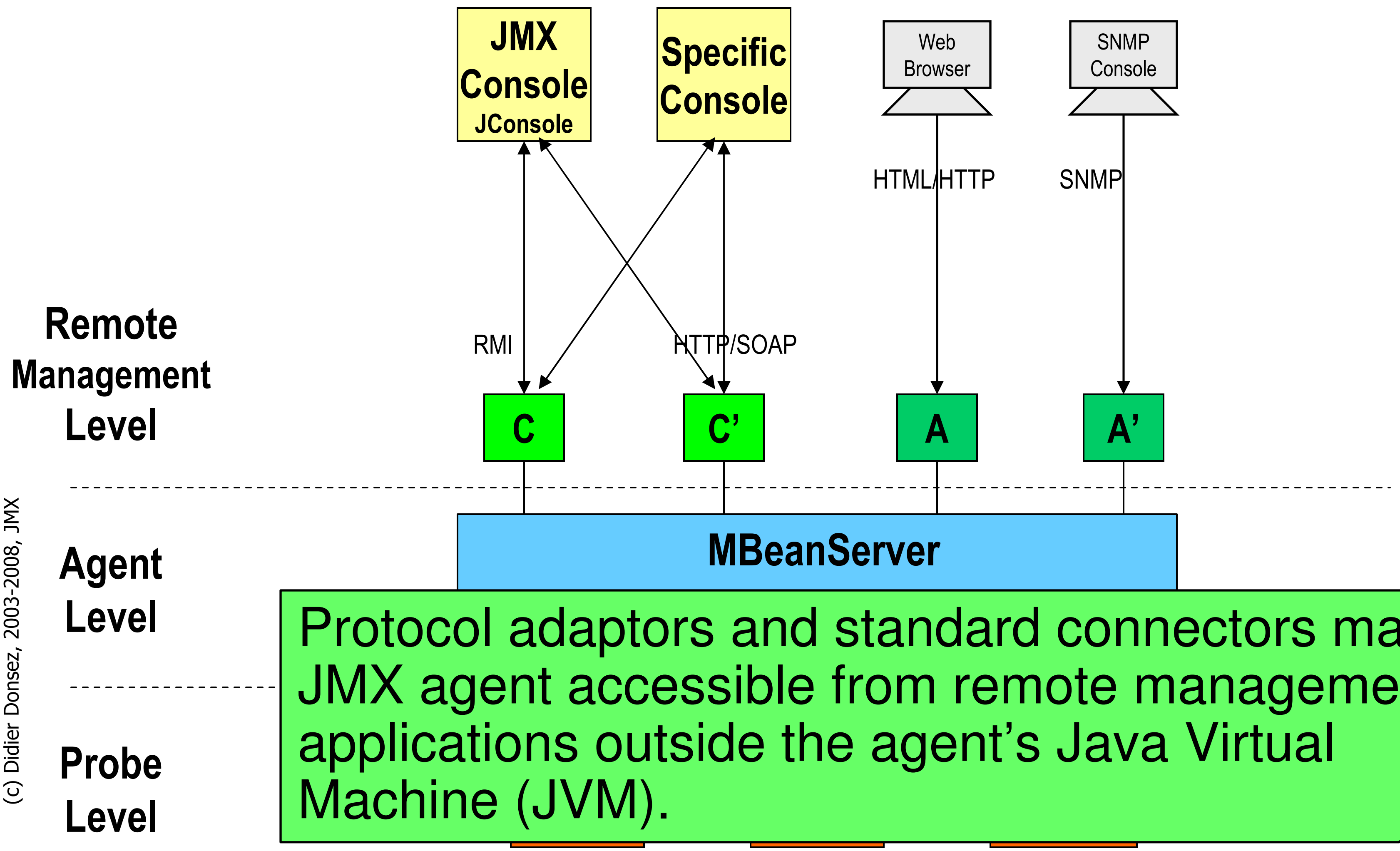
Resources, such as applications, devices, or services, are instrumented using Java objects called Managed Beans (MBeans).
MBeans expose their management interfaces, composed of attributes and operations, through a JMX agent for remote management and monitoring

JMX 3-Level Architecture
Agent level

The main component of a JMX agent is the MBean server.
This is a core managed object server in which MBeans are registered.
A JMX agent also includes a set of services for handling MBeans.
JMX agents directly control resources and make them available to remote management agents.



JMX 3-Level Architecture
Distribution level



Protocol adaptors and standard connectors make a JMX agent accessible from remote management applications outside the agent's Java Virtual Machine (JVM).

Probe Level
Mbean (M*anageable* Bean)

- represents any resource (device, application, …) that needs to be managed
- expose a management interface
  - a set of readable and/or writable attributes
  - a set of invokable operations, along with a self-description.
  - a set of
- 4 types of interface
  - standard MBeans
    - Class name' suffix must be « MBean »
    - setX(V)/getX() methods
  - dynamic MBeans
    - Interface javax.management.DynamicMBean
    - setAttribute("X" ,V)/getAttribute("X") methods
  - *open MBeans* and *model MBeans*

L’interface Mbean (Maneagable Bean)

Standard MBean

- statically defines its management interface through the names of the methods it contains.
- Example

```
public interface DiskMBean {
   public int getQuota();
   public void setQuota(int q);
   public int getUsageRatio();
   public void reset();
}
```

Dynamic MBean

- implements a specific Java interface
  and reveals its attributes and operations at runtime.
- Interface

```
interface DynamicMBean {
 Object getAttribute(String attribute);
         // Obtain the value of a specific attribute of the Dynamic MBean.
 AttributeList getAttributes(String[] attributes);
         // Get the values of several attributes of the Dynamic MBean.
 MBeanInfo getMBeanInfo();
       // Provides the exposed attributes and actions of the Dynamic MBean using an MBeanInfo object.
 Object invoke(String actionName, Object[] params, String[] signature);
         // Allows an action to be invoked on the Dynamic MBean.
 void setAttribute(Attribute attribute);
         // Set the value of a specific attribute of the Dynamic MBean.
 AttributeList setAttributes(AttributeList attributes);
         // Sets the values of several attributes of the Dynamic MBean.
}
```

Dynamic MBean Example

```
public class Disk
  extends NotificationBroadcasterSupport
  implements DynamicMBean {
  ...
  public Disk() {
    buildDynamicMBeanInfo();
   ...
  }
  public Object getAttribute(String attribute_name)
    throws AttributeNotFoundException, MBeanException, ReflectionException {
    if (attribute_name.equals("Quota")) { return getQuota();}
    if (attribute_name.equals("UsageRatio")) { return getUsageRatio(); }
    throw new AttributeNotFoundException("Cannot find " + attribute_name +" attribute in
   " + dClassName);
  }
  ...
```

JSR 255 Annotations

- Motivation
  - Annotated POJOs can be register directly as DynamicMBeans
  - Standard MBean interfaces are not more !
- javax.management
  - Description
  - DescriptorFields
  - DescriptorKey
  - ManagedAttribute
  - ManagedOperation
  - MBean
  - MXBean
  - NotificationInfo
  - NotificationInfos

JSR 255 Annotations Example

```
@Description("Cache configuration")
@MBean
@NotificationInfo(types={"com.example.notifs.create",
                         "com.example.notifs.destroy"})
public class Cache {

  @Description("Cache size in bytes")
  @ManagedAttribute
  int cacheSize;
  public int getCacheSize() { … }
  public void setCacheSize(int size) { … }

  @Description("Last time the configuration was changed in milliseconds since 1 Jan 1970")
  public long getLastChangedTime();

  @Description("Save the configuration to a file")
  @ManagedOperation(impact = Impact.ACTION)
  public void save(
        @Description("Optional name of the file, or null for the default name")
        String fileName) { … }
  …
}
```

MBean registration name

- ObjectName
  - MBeans must be registered on the MBeanServer with a registration name (class javax.management.ObjectName)
- Syntax
  - *domain*:*key-property-list*
  - Examples
    - com.sun.someapp:type=Whatsit,name=25
    - com.sun.someapp:type=Whatsit,name="25,26"
    - java.lang:type=GarbageCollector,name=MarkSweep
    - domain:type=Server,name=server5
    - domain:type=Server.Application,Server=server5,name=app1
    - domain:type=Server.Application.WebModule,Server=server5, Application=app1,name=module3
    - domain:type=Server.Application.WebModule.Servlet,Server=server5, Application=app1,WebModule=module3,name=default
- ObjectName pattern matching
  - MBean could be searched with pattern applied on their ObjectName
  - Exemple : *:type=GarbageCollector,*

ObjectName convention

- Remarks : Must follow some naming conventions
  - If the *domain* is empty, it implies the *default domain* for the MBean Server where the Object Name is used
  - Object Names are case sensitive and whitespace sensitive.
  - Every Object Name should contain a type= key property.
    - This property should be different for every object type in a given domain.
  - The most usual key property in addition to type is name.
  - The order of key properties is not significant.
    - no difference between type=Thread,name=DGC and name=DGC,type=Thread
  - The domain should not contain the character slash (/)
    - reserved for MBean Server hierarchies ("cascading").
  - managed objects can be logically contained in other managed objects
    - domain:type=Server,name=server5
    - domain:type=Server.Application,Server=server5,name=app1
    - domain:type=Server.Application.WebModule,Server=server5, Application=app1,name=module3
    - domain:type=Server.Application.WebModule.Servlet,Server=server5, Application=app1,WebModule=module3,name=default

Interface *MBeanRegistration*

- Un *MBean* peut récupérer la référence du *MBeanServer* qui l’enregistre en implémentant l’interface *MBeanRegistration* au moment de son enregistrement.

```
interface MBeanRegistration{
    void postDeregister()
    void postRegister(Boolean registrationDone)
    void preDeregister()
    ObjectName preRegister(MBeanServer server, ObjectName name)
}
```

- Remarque: Un *MBean* peut être enregistré sous plusieurs *ObjectNames* et par plusieurs *MBeanServers*
- Remarque: Les adaptateurs sont généralement des *MBeans* enregistrés qui implémentent cette interface

Change Notification (Push)

- Motivation
  - Push interaction
  - Publish-Subscribe for MBean internal changes
- Publisher
  - MBean implementing the *NotificationBroadcaster* interface
- Subscriber
  - For instance, remote consoles, …
- Notification
  - represents a notification object emitted by an MBean
  - Subclasses :
    - AttributeChangeNotification, JMXConnectionNotification, MBeanServerNotification, MonitorNotification, RelationNotification, TimerAlarmClockNotification, TimerNotification, …

Notification de changement (Push)

- Un *MBean* peut notifier des changements en implémentant l'interface *NotificationBroadcaster*.

```
interface NotificationBroadcaster {
    void addNotificationListener(NotificationListener listener,
                        NotificationFilter filter, Object handback)
    void removeNotificationListener(NotificationListener listener)
    MBeanNotificationInfo[] getNotificationInfo()
}
```

- Lors de l'enregistrement, le *MBeanServer* appelle la méthode d'ajout pour propager les notifications aux souscripteurs (qui peuvent être distant).

Monitors Package javax.management.monitor (J2SE5.0)

- Utility MBeans to monitor (observe) other MBean
  - CounterMonitorMBean, GaugeMonitorMBean, MonitorMBean, StringMonitorMBean, …
- Example

```
GaugeMonitor gm = new GaugeMonitor();
gm.setGranularityPeriod(5000);
gm.setDifferenceMode(false);
gm.setNotifyHigh(false);
gm.setNotifyLow(true);
gm.setThresholds(new Integer(50), new Integer(5));
gm.addObservedObject("system.disk:type=hda1");
gm.setObservedAttribute("freespace");
mbeanserver.registerMBean(gm, new ObjectName("Services:type=GaugeMonitor"));
gm.start();
```

OpenMBean

- TODO

- CompositeData
  - (interface) specifies the behavior of a specific type of complex *open data* objects which represent *composite data* structures.
- TabularData
  - (interface) specifies the behavior of a specific type of complex *open data* objects which represent *tabular data* structures.

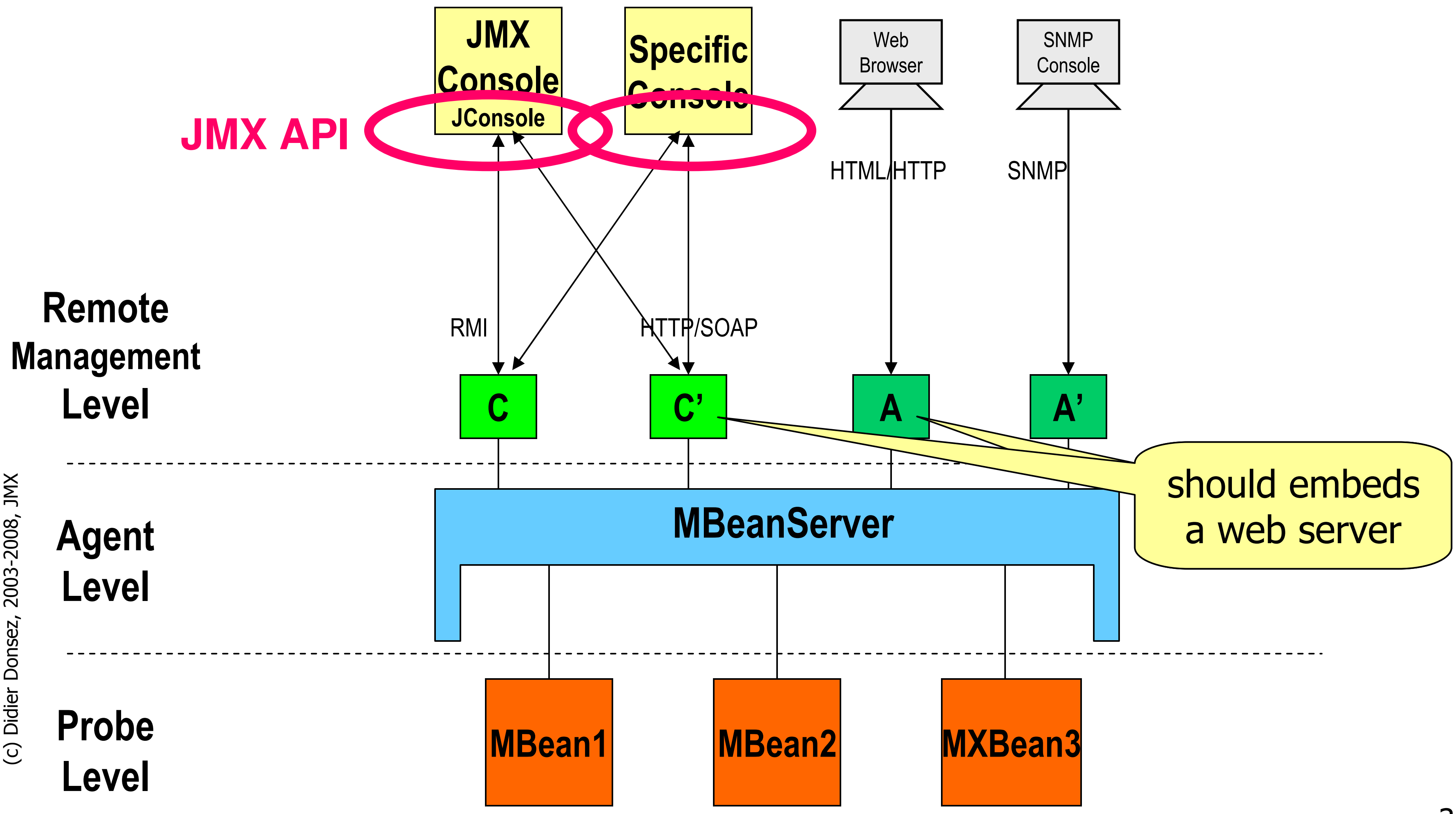
Virtual MBean

- TODO

Remote Management Level
Connectors and Adaptors

- Motivations
  - Protocol adaptors and standard connectors make a JMX agent accessible from remote management applications outside the agent’s JVM.
- Connectors
  - Provides the JMX API (remotly) with both pull and push interactions
- Adaptors
  - Adapts to a management protocol (SNMP, …)
  - Provides a man-oriented GUI (HTML, WML, …)

Remote Management Level Connectors and Adaptors



Remote Management Level Connectors and Adaptors

- Connectors
  - RMI and RMI/IIOP connectors
    - Standard in J2SE5.0 and mandatory
  - JMX Messaging Protocol (JMXMP) connector
    - based on TCP sockets. Defined by the JSR 160
  - Web Services (JSR 262)
  - XML/RPC, JMS (Lingo), XMPP …
- Adaptors
  - SNMP adaptor (provided by the J2SE5.0 for the JVM MIB (JSR163))
  - Web (HTML/HTTP) adaptor
  - Local GUI
  - CORBA Adaptor, AMI Adaptor for Tibco, …

Example

```
// Get the platform MBeanServer
    MBeanServer mbs = ManagementFactory.getPlatformMBeanServer();
// Unique identification of MBeans
    Hello helloBean = new Hello();
    ObjectName helloName = null;
// Uniquely identify the MBeans and register them with the MBeanServer
    helloName = new ObjectName("SimpleAgent:name=hellothere");
    mbs.registerMBean(helloBean, helloName);
// Create an RMI connector and start it
    JMXServiceURL url = new JMXServiceURL("service:jmx:rmi:///jndi/rmi://localhost:9999/server");
    JMXConnectorServer cs = JMXConnectorServerFactory.newJMXConnectorServer(url, null, mbs);
    cs.start();
// Register and start the HTML adaptor
    HtmlAdaptorServer adapter = new HtmlAdaptorServer();
    adapter.setPort(8000);
    ObjectName adapterName = new ObjectName("SimpleAgent:name=htmladapter,port=8000");
    mbs.registerMBean(adapter, adapterName);
    adapter.start();
```

Agent discovery

- Motivation
  - advertise and find JMX API agents (Connectors and Adaptors) in infrastructure/adhoc networks
    - *Remark : The Java Management Extensions (JMX) Remote API Specification describes how you can advertise and find JMX API agents by using existing discovery and lookup infrastructures. The specification does not define any discovery and lookup APIs specific to JMX technology. http://java.sun.com/j2se/1.5.0/docs/guide/jmx/overview/lookup.html#wp997349*
- Location protocols
  - Service Location Protocol (SLP)
  - Jini Network Technology
  - LDAP with JNDI
  - *DNS-SD, UPnP (not described by the specs)*

Agent discovery

- Motivation
  - advertise and find JMX API agents
    - *The Java Management Extensions (JMX) Remote API Specification describes how you can advertise and find JMX API agents by using existing discovery and lookup infrastructures.*
    - *The specification does not define any discovery and lookup APIs specific to JMX technology.*
- Discovery protocols
  - Service Location Protocol (SLP)
  - Jini Network Technology
  - LDAP with JNDI
  - *DNS-SD (not described by the specs)*

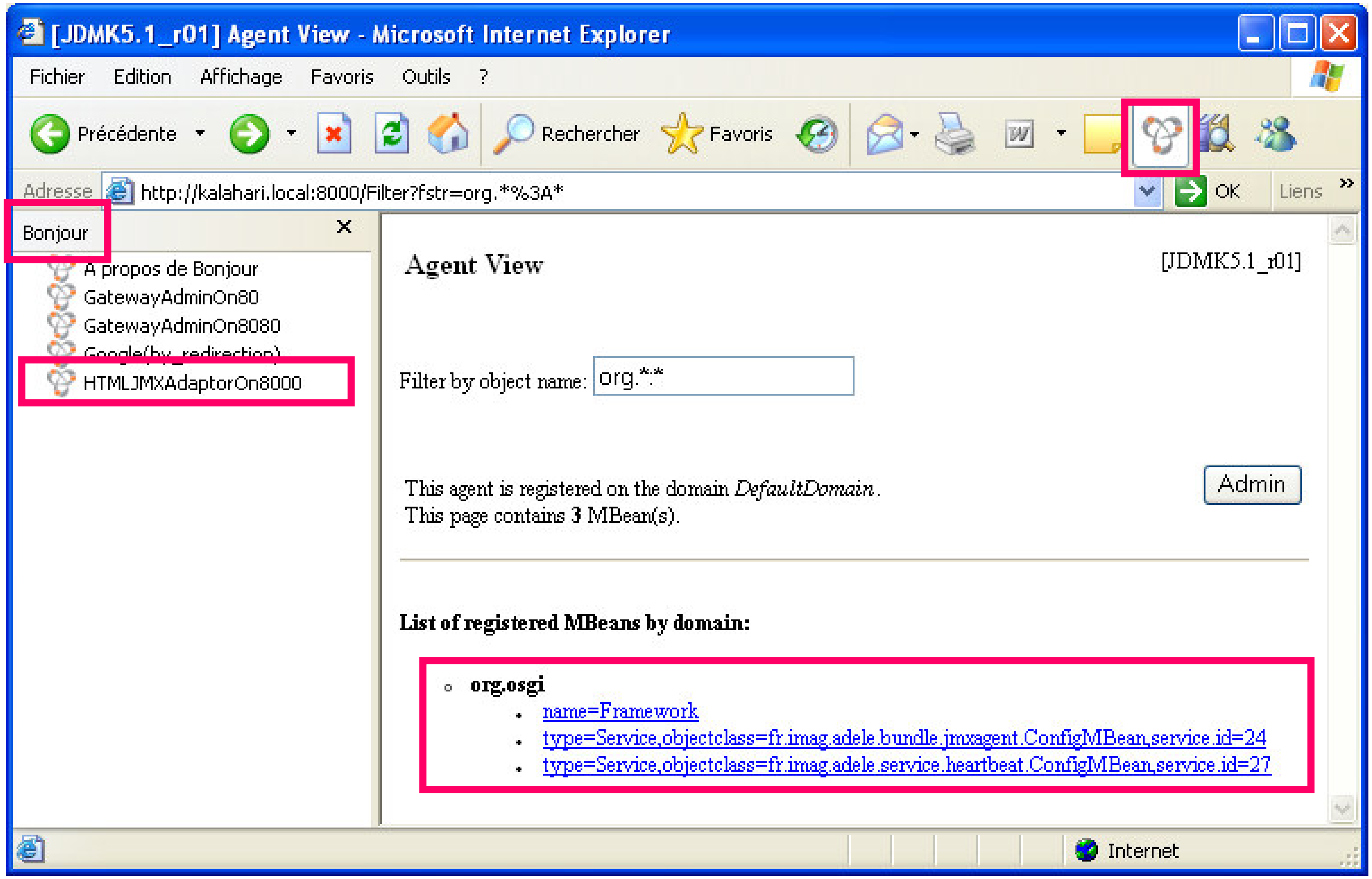
Agent discovery with Service Location Protocol (SLP)

- Recall
  - SLP is a lookup service for adhoc network
- Steps
  - *The following steps summarize the procedure defined in the JMX Remote API specification for using the SLP lookup service to advertise and find JMX agents:*

1. The agent creates one or more JMX connector servers.
2. For each connector to expose, the agent registers the address with the SLP lookup service, possibly giving additional attributes that qualify the agent and/or the connector, and can be used as filters.
3. The client queries the SLP lookup service, and retrieves one or more addresses that match the query.
4. Finally, the client obtains a connector that is connected with the server identified by a retrieved address.

Agent discovery with DNS-SD

- Direct for HTTP/HTML Adaptor



http://kalahari.local:8000/Filter?fstr=org.*%3A*

- A propos de Bonjour
- GatewayAdminOn80
- GatewayAdminOn8080
- Google(by_redirection)
- HTMLJMXAdaptorOn8000

Agent View [JDMK5.1_r01]

Filter by object name: org.*:*

This agent is registered on the domain *DefaultDomain*.
This page contains **3** MBean(s).

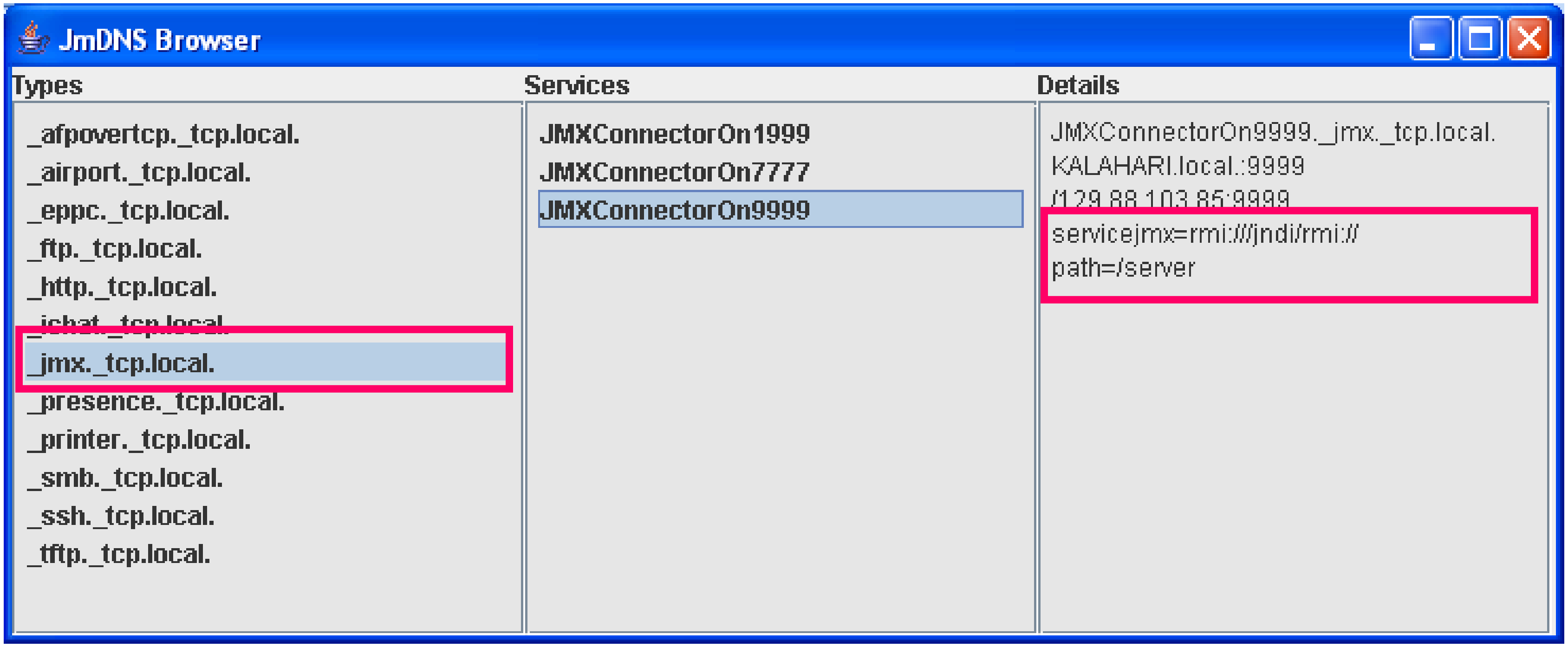
Admin

List of registered MBeans by domain:

- **org.osgi**
  - name=Framework
  - type=Service,objectclass=fr.imag.adele.bundle.jmxagent.ConfigMBean,service.id=24
  - type=Service,objectclass=fr.imag.adele.service.heartbeat.ConfigMBean,service.id=27

Agent discovery with DNS-SD

- Example with DNS-SD
  - New type for service:jmx:* URLs



```
_afpovertcp._tcp.local.
_airport._tcp.local.
_eppc._tcp.local.
_ftp._tcp.local.
_http._tcp.local.
_ichat._tcp.local.
_jmx._tcp.local.
_presence._tcp.local.
_printer._tcp.local.
_smb._tcp.local.
_ssh._tcp.local.
_tftp._tcp.local.
```

```
JMXConnectorOn1999
JMXConnectorOn7777
JMXConnectorOn9999
```

```
JMXConnectorOn9999._jmx._tcp.local.
KALAHARI.local.:9999
/129.88.103.85:9999
servicejmx=rmi:///jndi/rmi://
path=/server
```

- Usage : useful for JavaEE node discovery in a cluster

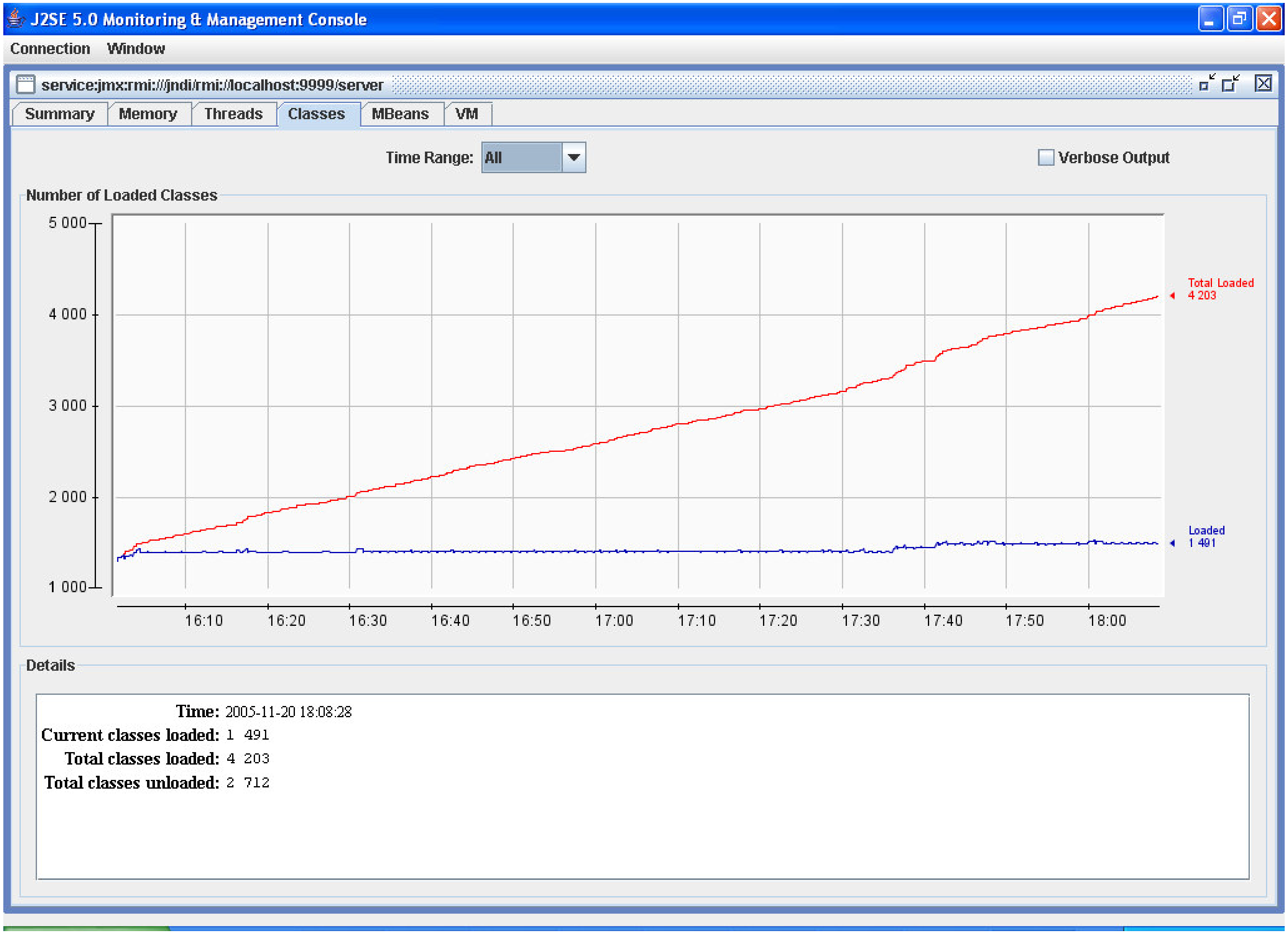
API JMX

- Initially adopted by the J2EE community
- Now in J2SE 5.0
- Packages
  - javax.management
  - javax.management.loading
  - javax.management.modelmbean
  - javax.management.monitor
  - javax.management.openmbean
  - javax.management.relation
  - javax.management.remote
  - javax.management.remote.rmi
  - javax.management.timer
  - *java.lang.management*

MXBean (Platform MBean) Package java.lang.management (J2SE5.0)

- Provide MBean to monitor the platform (JVM and underlying OS)
- Interfaces
  - ClassLoadingMXBean java.lang:type=ClassLoading
  - CompilationMXBean java.lang:type=Compilation
  - GarbageCollectorMXBean
    java.lang:type=GarbageCollector,name=collector's name
  - MemoryManagerMXBean
  - MemoryMXBean
  - MemoryPoolMXBean
  - OperatingSystemMXBean
  - RuntimeMXBean
  - ThreadMXBean
- Custom tabs for the JConsole 6

MXBean (Platform MBean) *ClassLoadingMXBean* Tab in JConsole



Time: 2005-11-20 18:08:28
Current classes loaded: 1 491
Total classes loaded: 4 203
Total classes unloaded: 2 712

Timer
Package javax.management.timer

- Utility MBean
  to manage a list of dated timer notifications.

```
interface TimerMBean {
    …
    Integer addNotification(String type, String message,
                            Object userData, Date date,
                            long period, long nbOccurences, boolean fixedRate);
    …
}
```

- Subscribers are notified when a timer expires
  - TimerNotification(String type, Object source,
    long sequenceNumber, long timeStamp, String msg, Integer id)

MLet (Management applet)

- Utility to load, instanciate and register MBeans in the MBeanServer from a (« XML ») description
  - Class javax.management.loading.MLet
  - Description:

```
<MLET CODE = class | OBJECT = serfile
      ARCHIVE = archiveList
      [CODEBASE = codebaseURL]
      [NAME = mbeanname]
      [VERSION = version]
      >[arglist]</MLET>
```

- registered as MBean

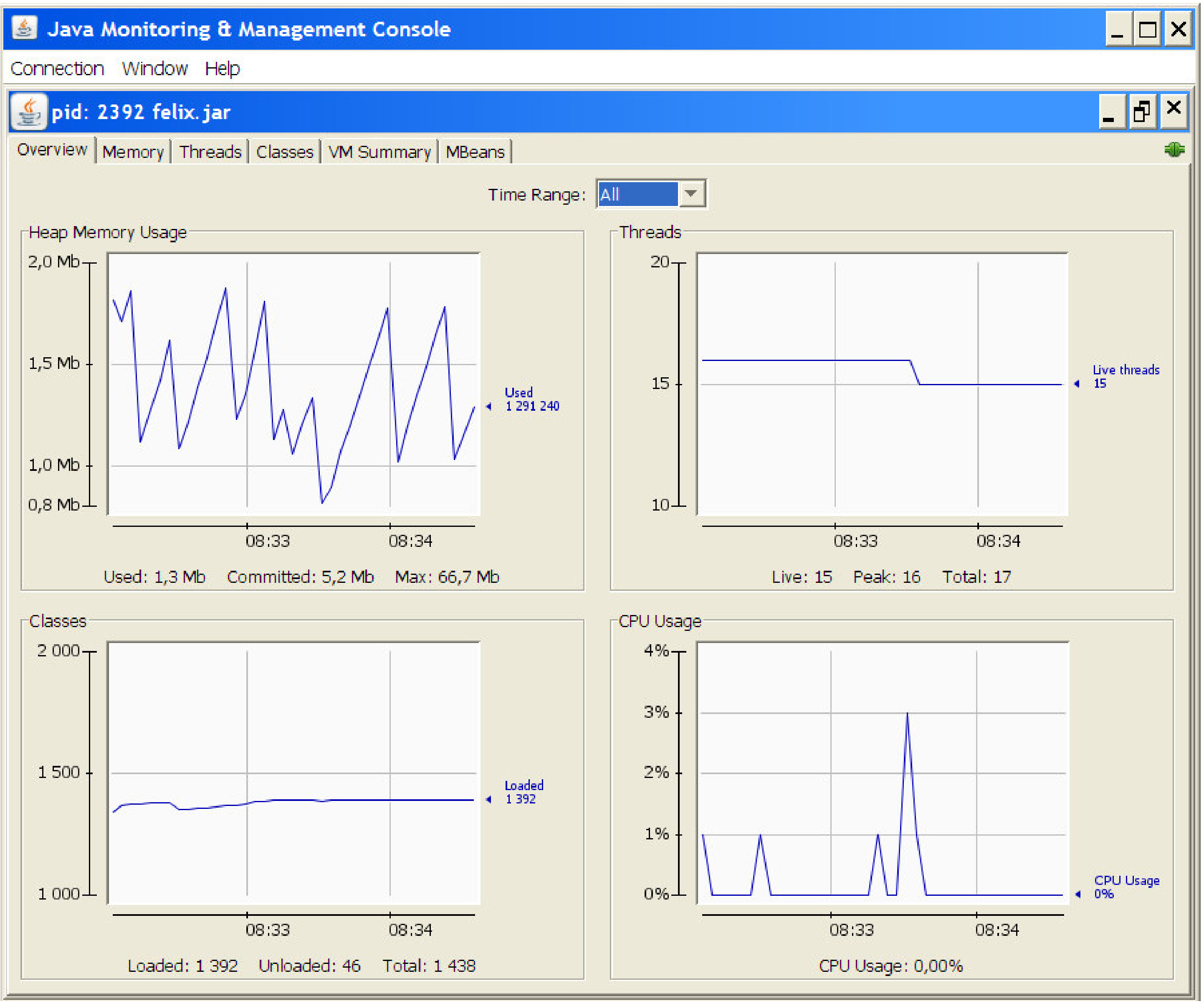
```
mbeanserver.registerMBean(new MLet(), new ObjectName("Services:type=MLet"));
```

- However
  - MLet extends URClassLoader so
    - No class unloading
    - No package dependencies resolution and No package versioning

Consoles

- User agents to monitor (remotly) a server
  - Set/Get attributes
  - Invoke operations
  - Subscribe to notification
  - Keep histories
- Consoles (general-purpose)
  - JConsole
  - MC4J
  - XtremJ
  - jManage http://www.jmanage.org/
  - …
  - Remarks
    - Custom « tabs »/plugins can be added !

JConsole (provided with J2SE since v 5.0)



Time Range: All

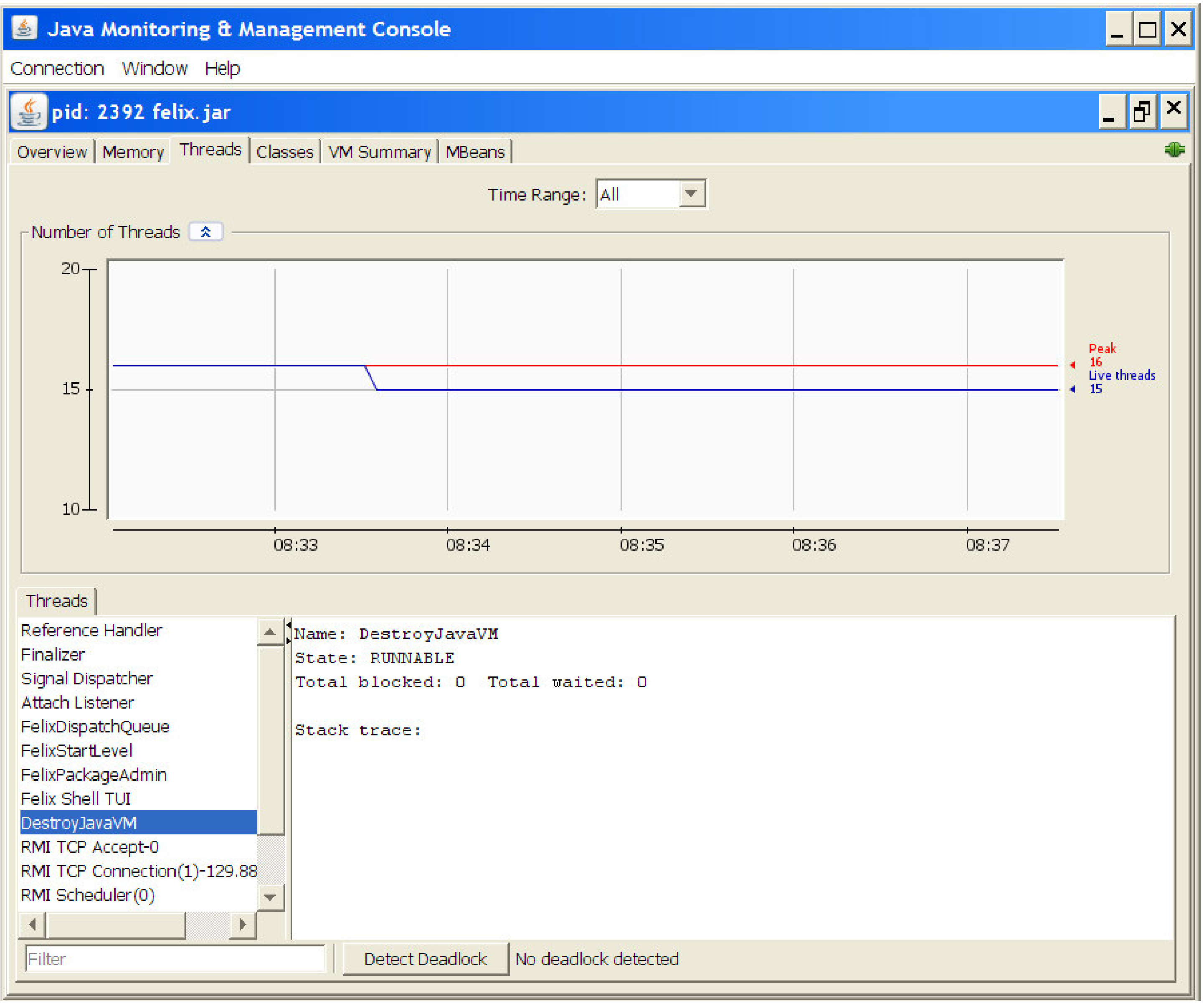
Used: 1,3 Mb   Committed: 5,2 Mb   Max: 66,7 Mb

Live: 15   Peak: 16   Total: 17

Loaded: 1 392   Unloaded: 46   Total: 1 438

CPU Usage: 0,00%

JConsole 6 Example



Reference Handler
Finalizer
Signal Dispatcher
Attach Listener
FelixDispatchQueue
FelixStartLevel
FelixPackageAdmin
Felix Shell TUI
DestroyJavaVM
RMI TCP Accept-0
RMI TCP Connection(1)-129.88
RMI Scheduler(0)

```
Name: DestroyJavaVM
State: RUNNABLE
Total blocked: 0  Total waited: 0

Stack trace:
```

No deadlock detected

JConsole 6 Example

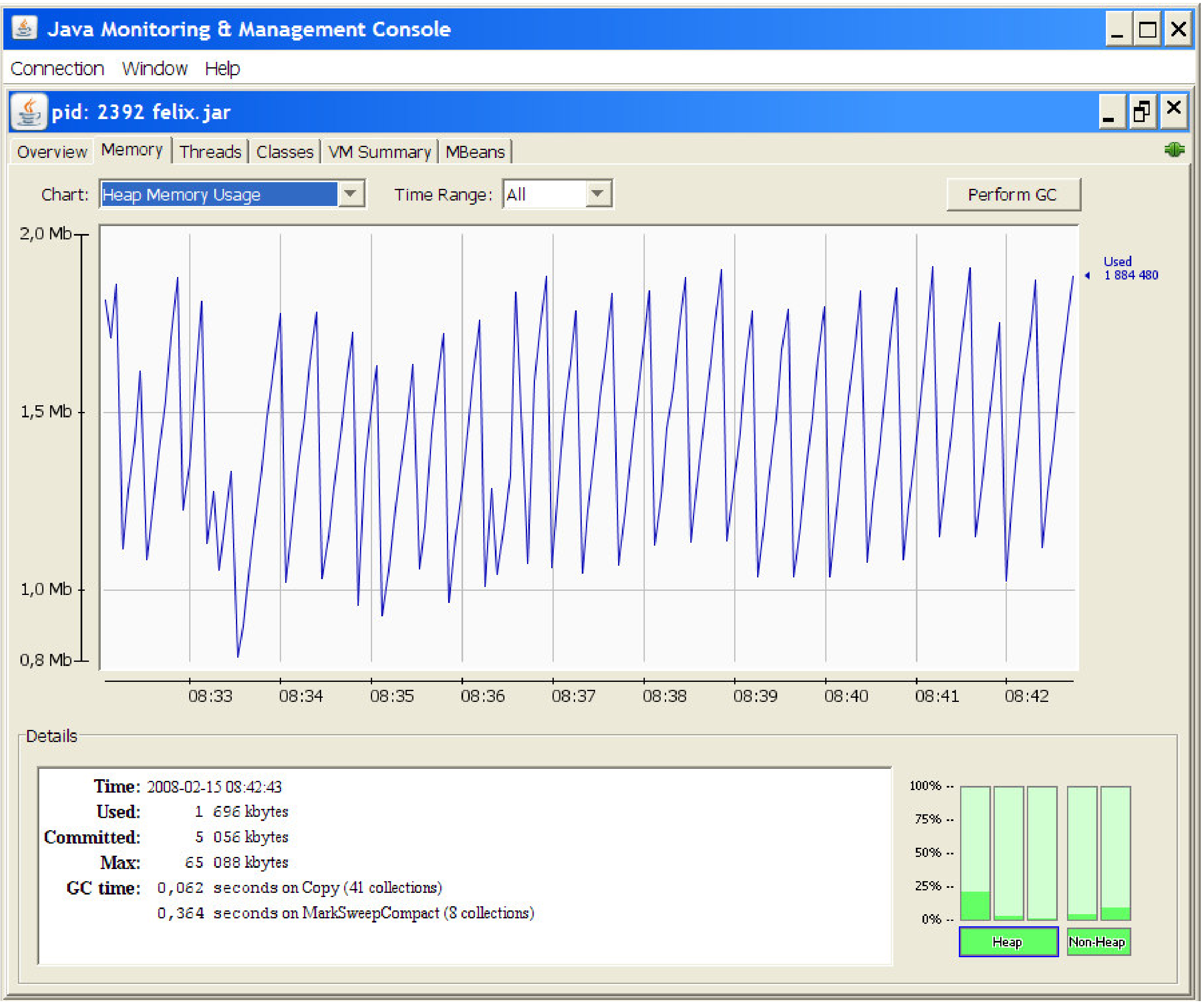


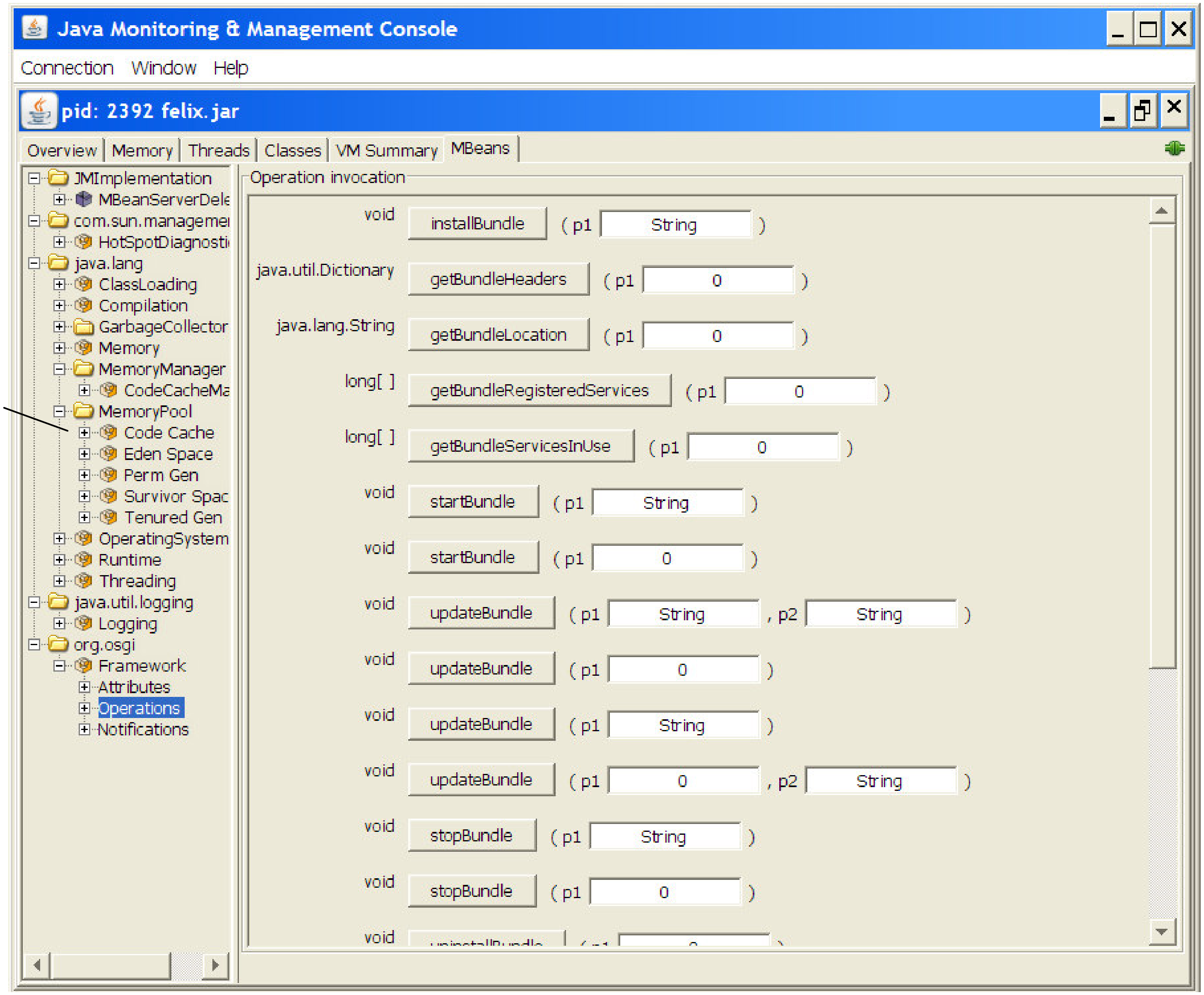
Chart: Heap Memory Usage

Time Range: All

Time: 2008-02-15 08:42:43
Used: 1 696 kbytes
Committed: 5 056 kbytes
Max: 65 088 kbytes
GC time: 0,062 seconds on Copy (41 collections)
0,364 seconds on MarkSweepCompact (8 collections)

JConsole 6 Example

JVM specific



Java Monitoring & Management Console

Connection Window Help

pid: 2392 felix.jar

Overview | Memory | Threads | Classes | VM Summary | MBeans

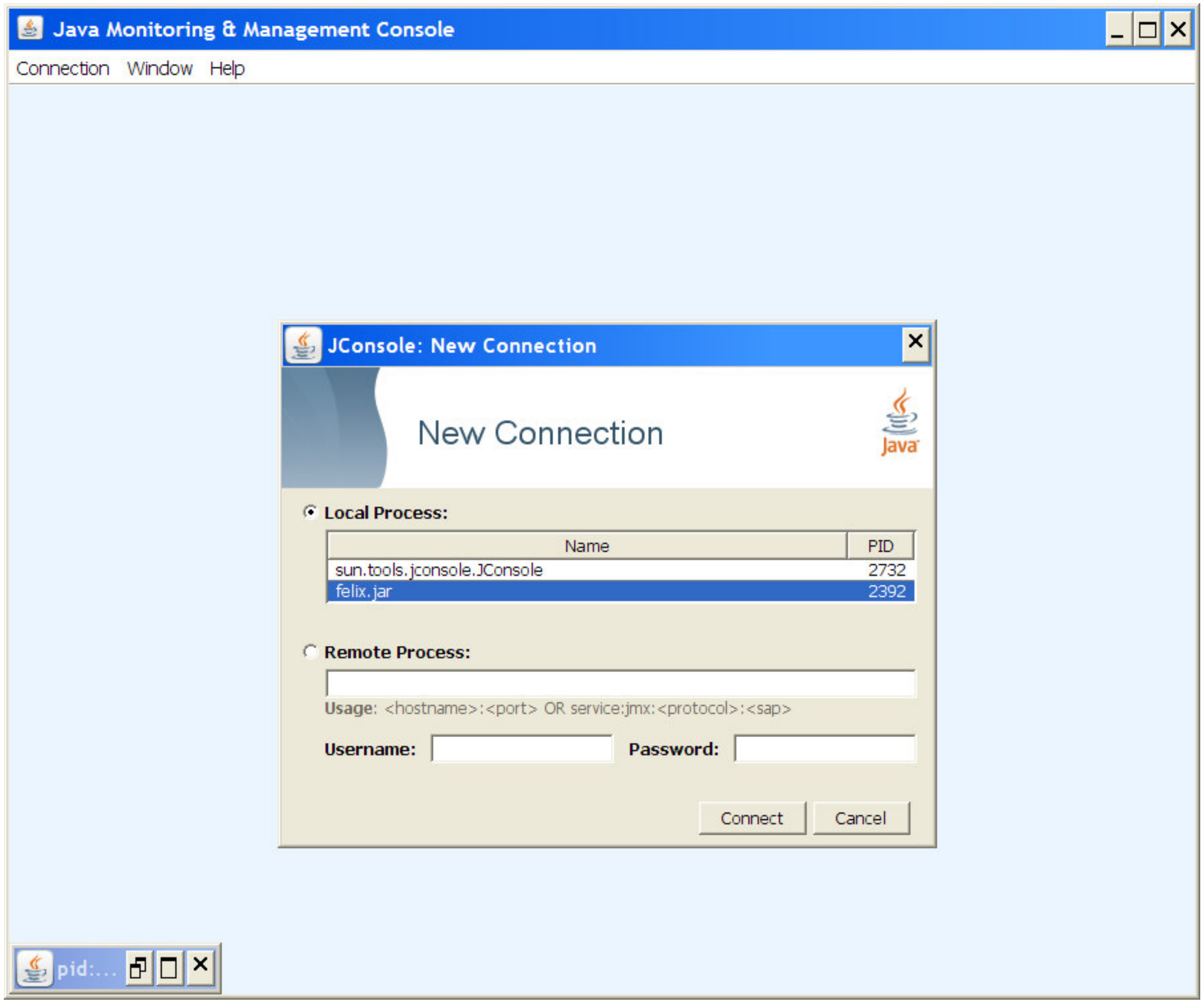
```
⊟ JMImplementation
   ⊞ MBeanServerDele
⊟ com.sun.manageme
   ⊞ HotSpotDiagnosti
⊟ java.lang
   ⊞ ClassLoading
   ⊞ Compilation
   ⊞ GarbageCollector
   ⊞ Memory
   ⊟ MemoryManager
      ⊞ CodeCacheMa
   ⊟ MemoryPool
      ⊞ Code Cache
      ⊞ Eden Space
      ⊞ Perm Gen
      ⊞ Survivor Spac
      ⊞ Tenured Gen
   ⊞ OperatingSystem
   ⊞ Runtime
   ⊞ Threading
⊟ java.util.logging
   ⊞ Logging
⊟ org.osgi
   ⊟ Framework
      ⊞ Attributes
      ⊞ Operations
      ⊞ Notifications
```

Operation invocation

```
void                 installBundle                ( p1 String )
java.util.Dictionary getBundleHeaders             ( p1 0 )
java.lang.String     getBundleLocation            ( p1 0 )
long[ ]              getBundleRegisteredServices  ( p1 0 )
long[ ]              getBundleServicesInUse       ( p1 0 )
void                 startBundle                  ( p1 String )
void                 startBundle                  ( p1 0 )
void                 updateBundle                 ( p1 String , p2 String )
void                 updateBundle                 ( p1 0 )
void                 updateBundle                 ( p1 String )
void                 updateBundle                 ( p1 0 , p2 String )
void                 stopBundle                   ( p1 String )
void                 stopBundle                   ( p1 0 )
void
```

JConsole 6 Connection

- Local JVM (Java process) or Service URL



● Local Process:

Name	PID
sun.tools.jconsole.JConsole	2732
felix.jar	2392

○ Remote Process:

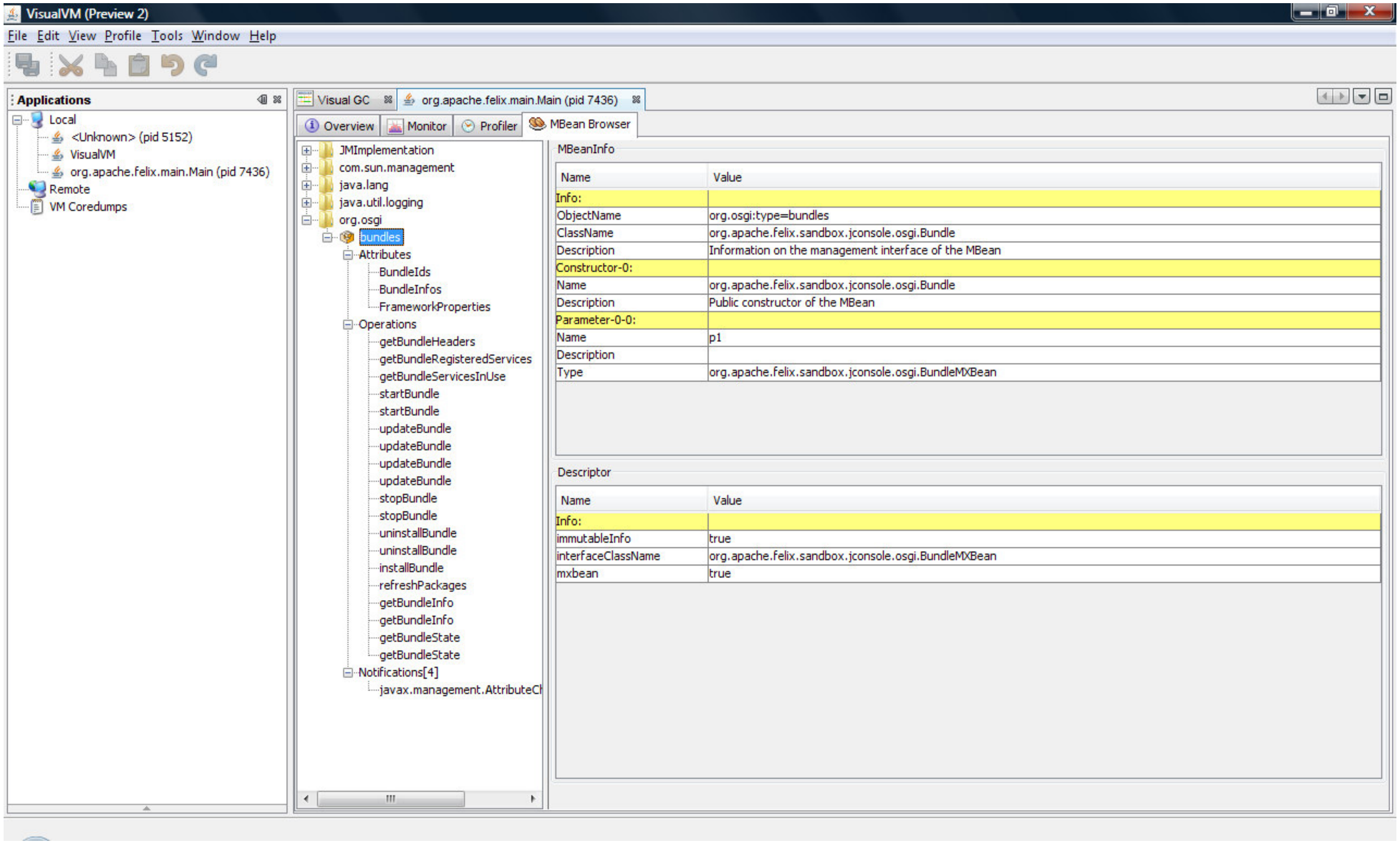
Usage: <hostname>:<port> OR service:jmx:<protocol>:<sap>

Username: Password:

VisualVM

https://visualvm.dev.java.net/

- « *VisualVM is a visual tool that integrates several existing JDK software tools and lightweight memory and CPU profiling capabilities. This tool is designed for both production and development time use and further enhances the capability of monitoring and performance analysis for the Java SE platform.»*
- **VisualVM includes the JConsole.**



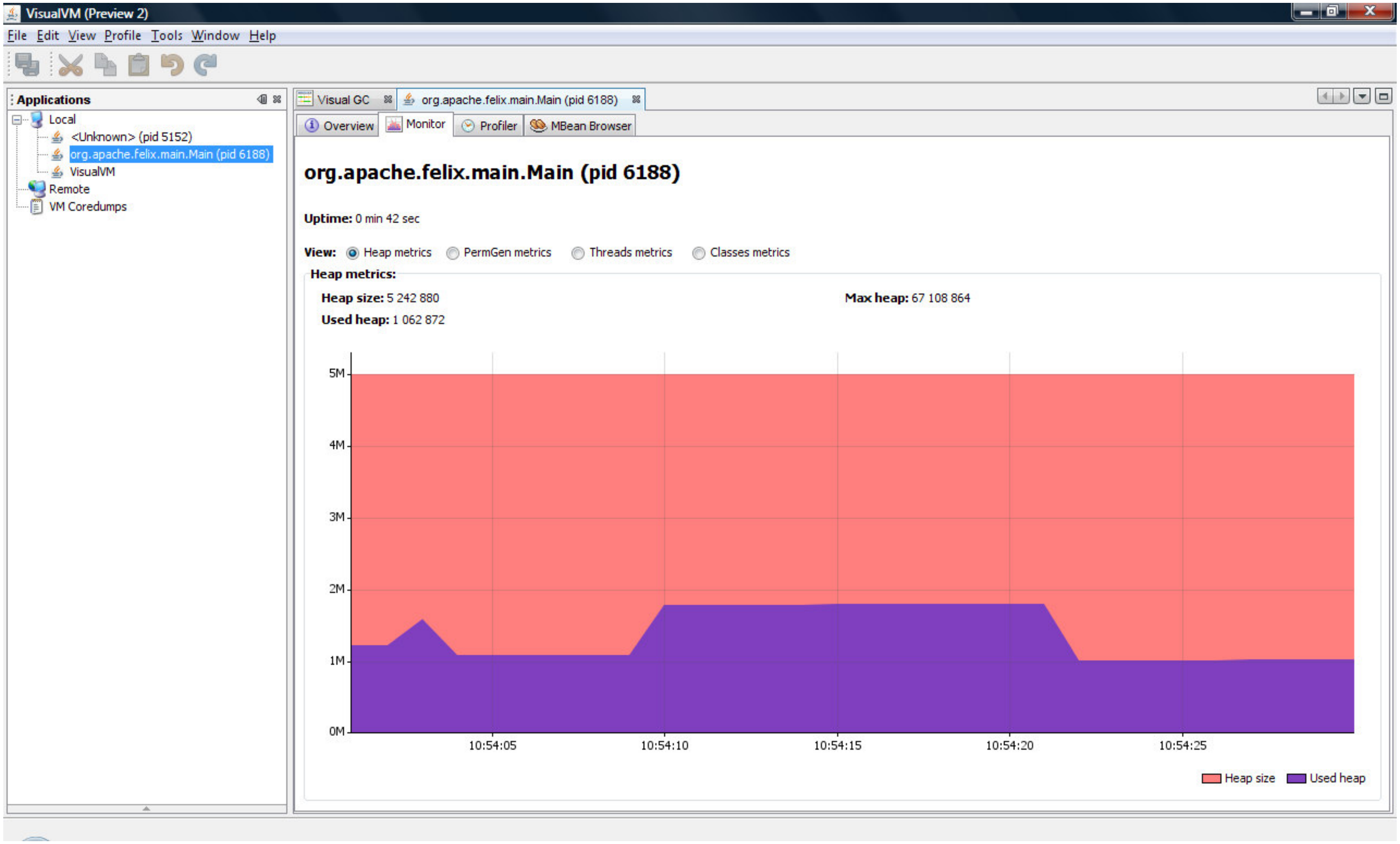
MBeanInfo

Name	Value
Info:	
ObjectName	org.osgi:type=bundles
ClassName	org.apache.felix.sandbox.jconsole.osgi.Bundle
Description	Information on the management interface of the MBean
Constructor-0:	
Name	org.apache.felix.sandbox.jconsole.osgi.Bundle
Description	Public constructor of the MBean
Parameter-0-0:	
Name	p1
Description	
Type	org.apache.felix.sandbox.jconsole.osgi.BundleMXBean

Descriptor

Name	Value
Info:	
immutableInfo	true
interfaceClassName	org.apache.felix.sandbox.jconsole.osgi.BundleMXBean
mxbean	true

VisualVM



org.apache.felix.main.Main (pid 6188)

Uptime: 0 min 42 sec

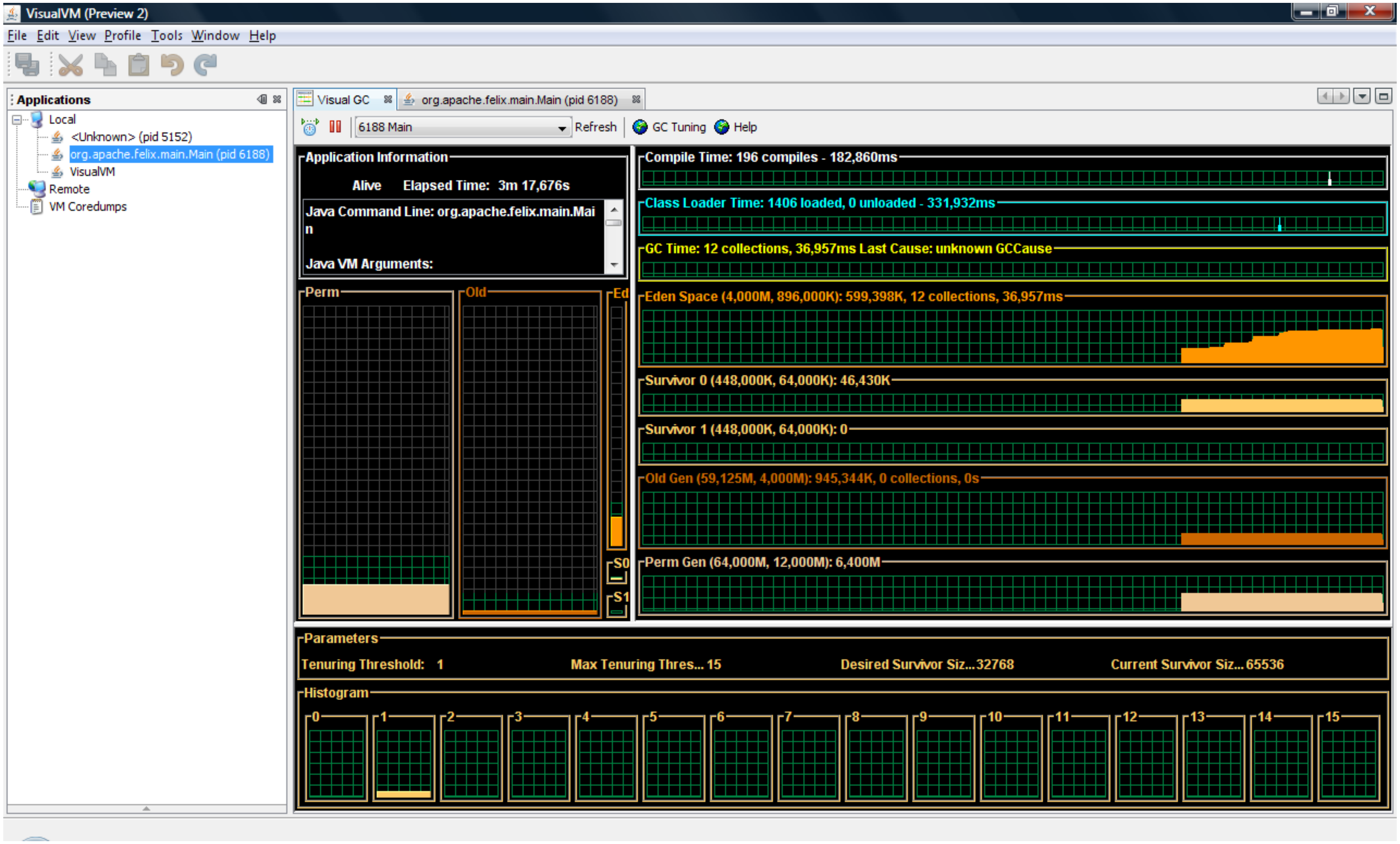
View: ● Heap metrics ○ PermGen metrics ○ Threads metrics ○ Classes metrics

Heap metrics:

Heap size: 5 242 880 **Max heap:** 67 108 864

Used heap: 1 062 872

VisualVM



Application Information

Alive Elapsed Time: 3m 17,676s

Java Command Line: org.apache.felix.main.Main

Java VM Arguments:

Compile Time: 196 compiles - 182,860ms

Class Loader Time: 1406 loaded, 0 unloaded - 331,932ms

GC Time: 12 collections, 36,957ms Last Cause: unknown GCCause

Eden Space (4,000M, 896,000K): 599,398K, 12 collections, 36,957ms

Survivor 0 (448,000K, 64,000K): 46,430K

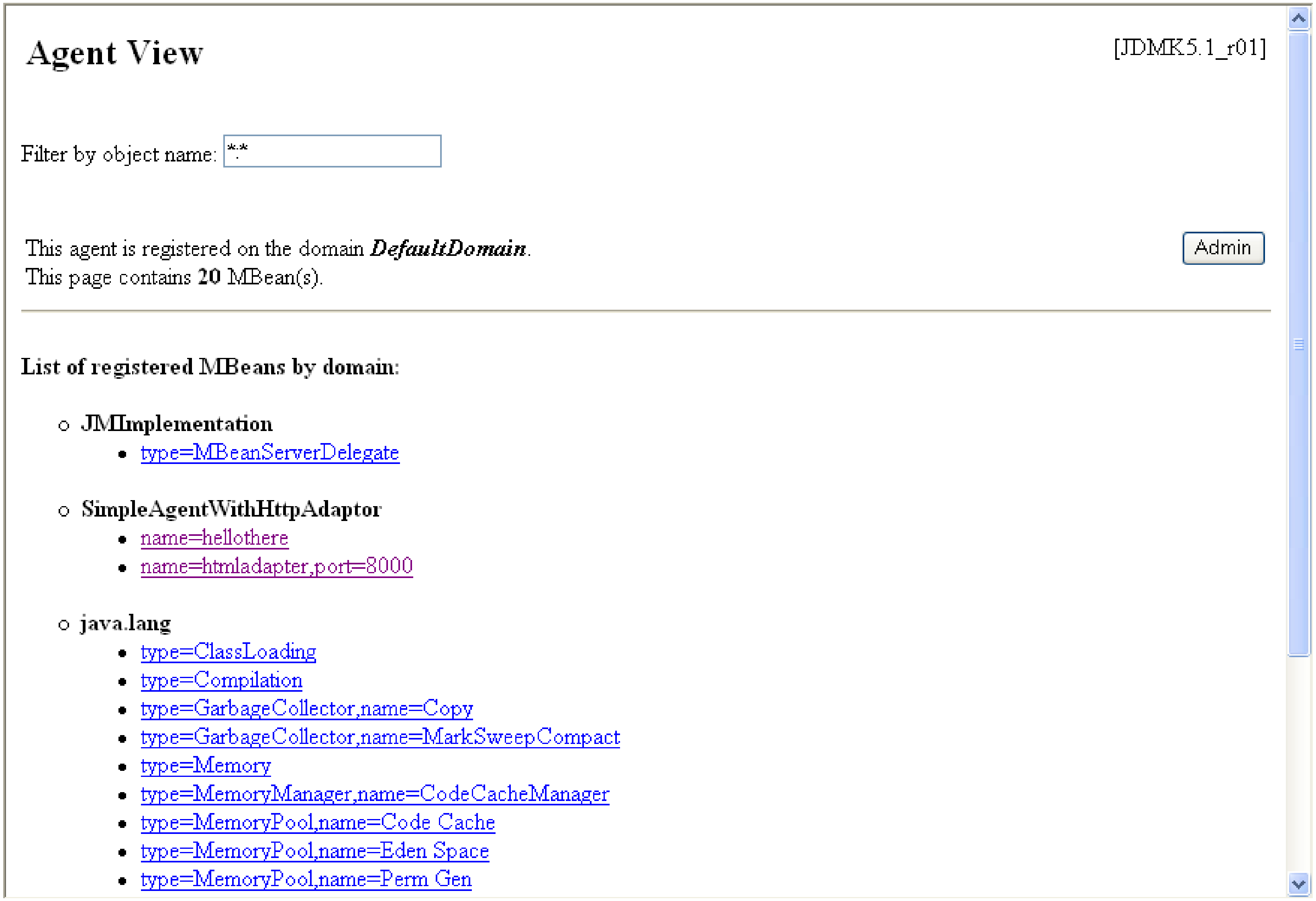
Survivor 1 (448,000K, 64,000K): 0

Old Gen (59,125M, 4,000M): 945,344K, 0 collections, 0s

Perm Gen (64,000M, 12,000M): 6,400M

Tenuring Threshold: 1 Max Tenuring Thres... 15 Desired Survivor Siz...32768 Current Survivor Siz... 65536

JMX RI 1.2
l'adaptateur http (i)



Agent View

[JDMK5.1_r01]

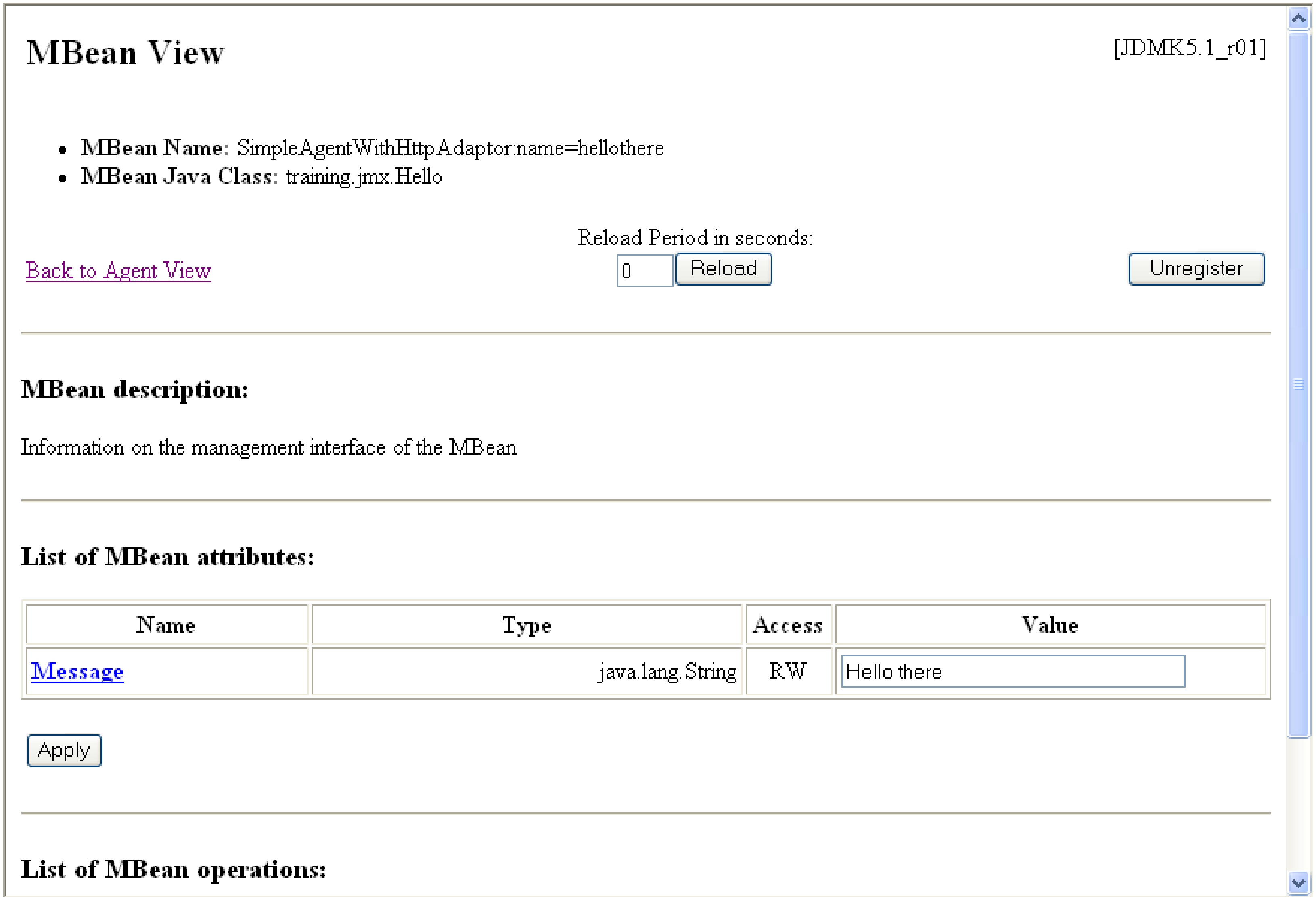
Filter by object name: *:*

This agent is registered on the domain ***DefaultDomain***.
This page contains **20** MBean(s).

List of registered MBeans by domain:

- **JMImplementation**
  - type=MBeanServerDelegate
- **SimpleAgentWithHttpAdaptor**
  - name=hellothere
  - name=htmladapter,port=8000
- **java.lang**
  - type=ClassLoading
  - type=Compilation
  - type=GarbageCollector,name=Copy
  - type=GarbageCollector,name=MarkSweepCompact
  - type=Memory
  - type=MemoryManager,name=CodeCacheManager
  - type=MemoryPool,name=Code Cache
  - type=MemoryPool,name=Eden Space
  - type=MemoryPool,name=Perm Gen

JMX RI 1.2
l'adaptateur http (ii)



MBean View

[JDMK5.1_r01]

- **MBean Name:** SimpleAgentWithHttpAdaptor:name=hellothere
- **MBean Java Class:** training.jmx.Hello

Reload Period in seconds:

Back to Agent View

MBean description:

Information on the management interface of the MBean

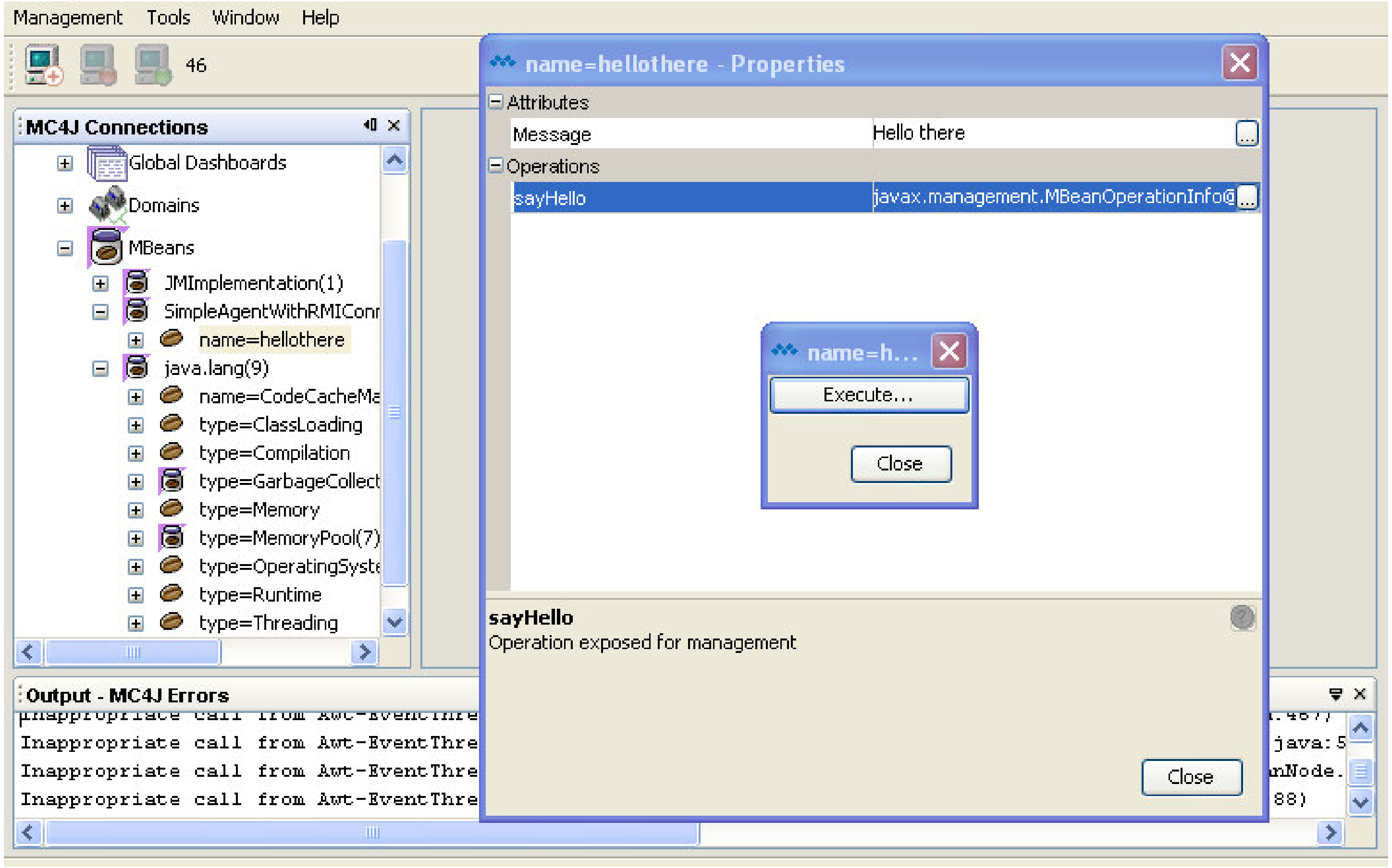
List of MBean attributes:

Name	Type	Access	Value
Message	java.lang.String	RW	Hello there

List of MBean operations:

MC4J *(http://mc4j.sf.net)*

- Console JMX open-source
  - Profiles de connexion pour la plupart des serveurs JavaEE
  - Tableaux de bord personnalisables (configuration XML)



Message	Hello there
sayHello	javax.management.MBeanOperationInfo@

sayHello
Operation exposed for management

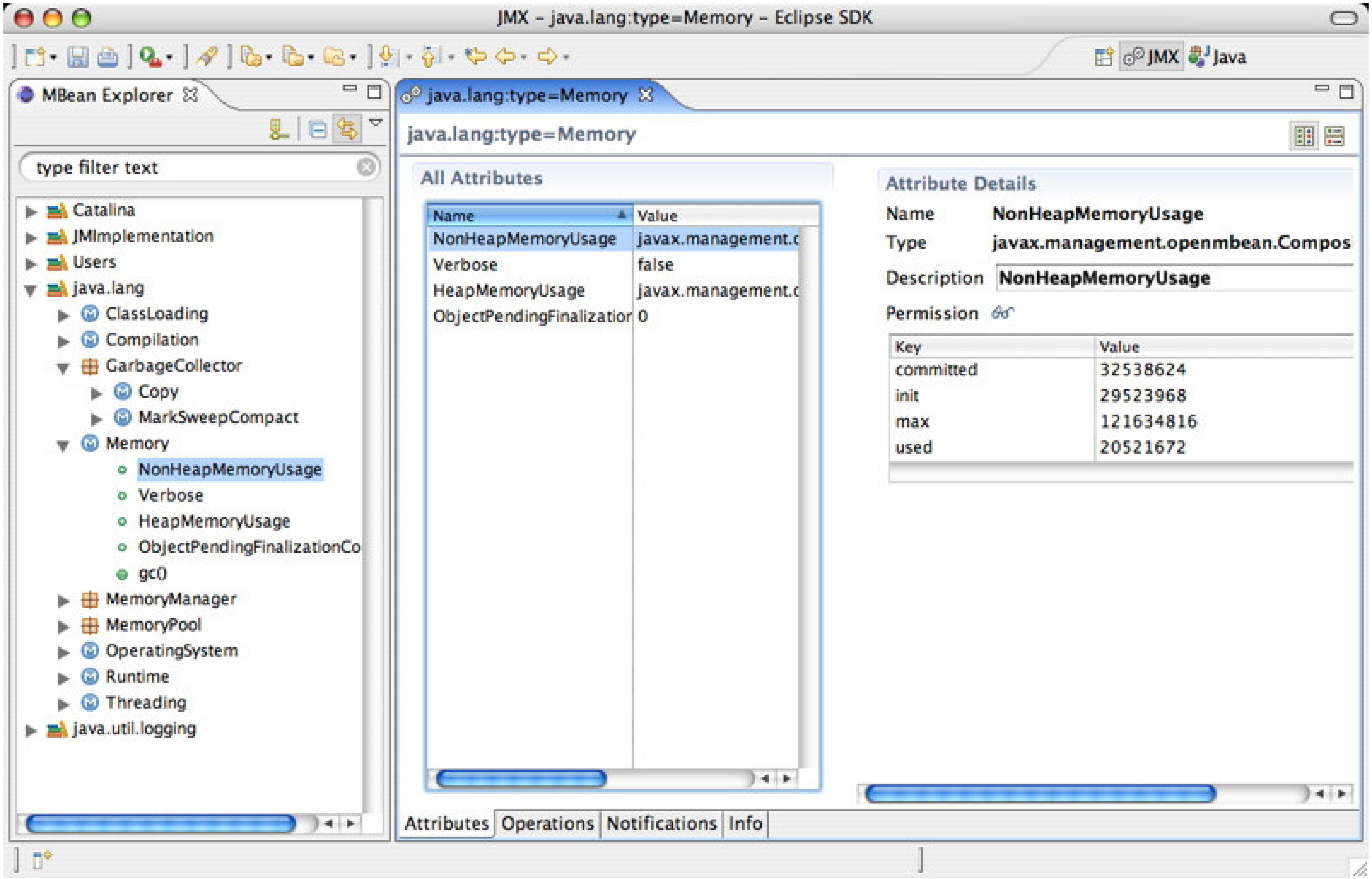
```
Inappropriate call from Awt-EventThre
Inappropriate call from Awt-EventThre
Inappropriate call from Awt-EventThre
Inappropriate call from Awt-EventThre
```

```
:467)
java:5
nNode.
88)
```

Eclipse JMX

http://code.google.com/p/eclipse-jmx/

- JMX console for Eclipse IDE



JMX – java.lang:type=Memory – Eclipse SDK

java.lang:type=Memory

All Attributes

Name	Value
NonHeapMemoryUsage	javax.management.c
Verbose	false
HeapMemoryUsage	javax.management.c
ObjectPendingFinalizatior	0

Attribute Details

Name NonHeapMemoryUsage

Type javax.management.openmbean.Compos

Description NonHeapMemoryUsage

Permission

Key	Value
committed	32538624
init	29523968
max	121634816
used	20521672

(c) Didier Donsez, 2003-2008, JMX

JMX-related JSR

- JSR 3 - Java Management Extensions (JMXTM) Specification
- JSR 160 - Java Management Extensions (JMX) Remote API 1.0
- JSR 255 - Java Management Extensions (JMX) Specification, v 2.0
  - updates the JMX and JMX Remote APIs for J2SE 1.6
  - Annotated MBeans, …
- JSR 262 - Web Services Connector for JavaTM Management Extensions (JMXTM) Agents

- JSR 18 - JAIN OAM API Specification
- JSR 22 - JAIN SLEE API Specification
- JSR 77 - J2EE Management Specification
- JSR 138 - Performance Metric Instrumentation
- JSR 151 - J2EE 1.4 Specification
- JSR 174 - Management and Monitoring Specification for the JVM
- JSR 176 - J2SE 1.5 (now called J2SE 5) Release Content

- JSR 163 JVM MIB for SNMP

JMX and ANT

- JMX4Ant http://jmx4ant.sourceforge.net/
  - *provides the ability to interact with JMX MBeans from Ant. Supports several popular JMX implementations and J2EE servers including JBoss and BEA WebLogic Server*
  - *Seems inactive*
- Jakarta TomCat JMX remote Ant (catalina-ant.jar)
  - Supports JSR 160

JMX and Apache TomCat

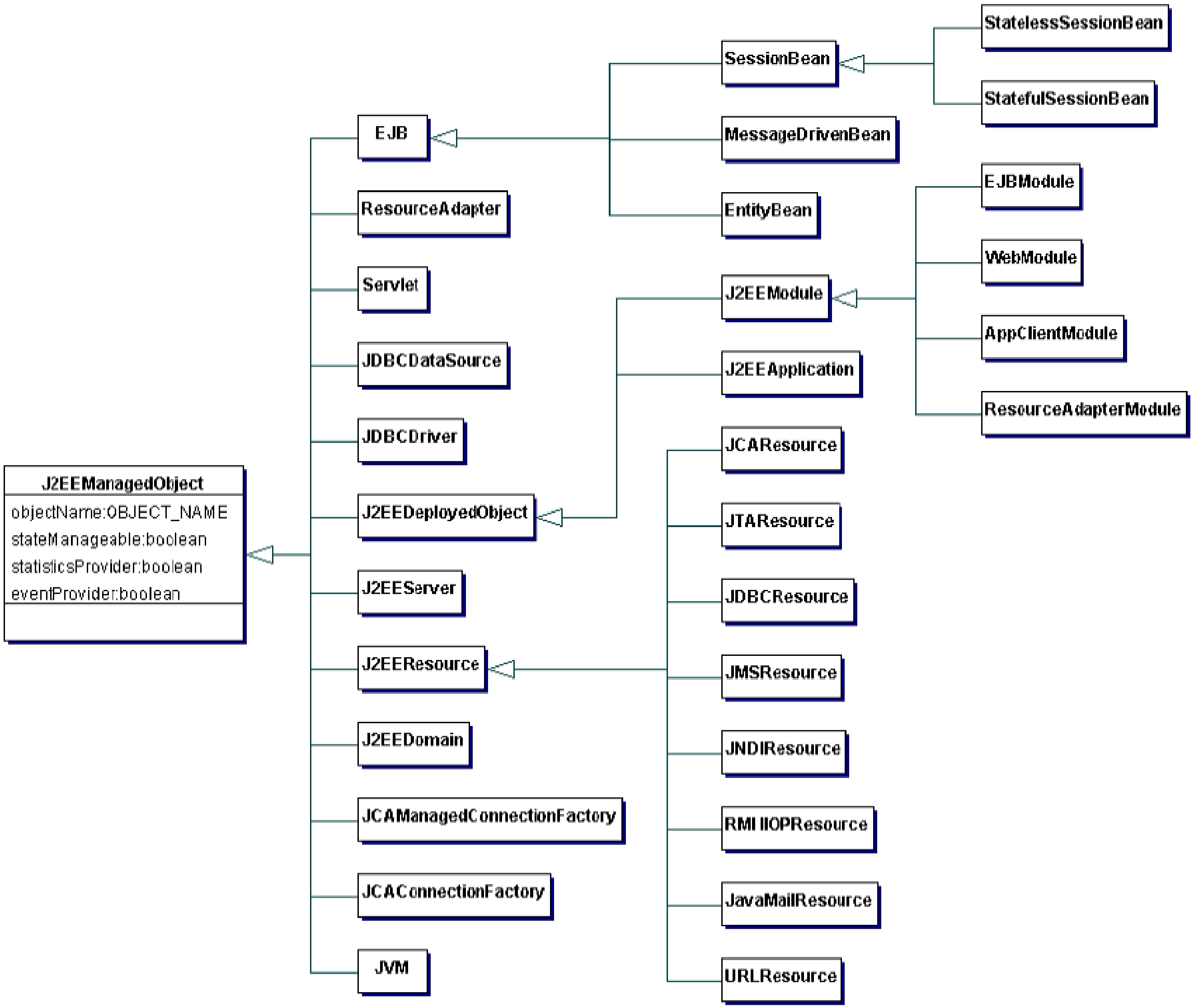
- Motivation
  - grab some statistic data or reconfigure some aspects
- Tools
  - JMXProxyServlet (simple)
    org.apache.catalina.ant.JMX*Task
  - JSR 160 (RMI Connector) Tasks
    org.apache.catalina.ant.jmx.*
  - JMX remote Ant (catalina-ant.jar)
- See
  - **Catalina Functional Specifications - Tomcat MBean Names**
    - %TOMCAT_HOME%**catalina\funcspecs\mbean-names.html**
  - **Monitoring and Managing Tomcat**
    - %TOMCAT_HOME%\webapps\tomcat-docs\monitoring.html

JSR77
J2EE Management Specification

Under Construction

- Motivations
  - Managed Objects
  - Event notification
  - State manageable object (SMO)
    - SMO states: starting, running, failed, stopping, stopped
    - recursive starting and stopping
    - Event notifications on state transition
  - Performance monitoring
    - Statitic
- J2EE Management EJB Component (MEJB) (JSR77.7)
- J2EE Management SNMP (JSR77.8)
- J2EE Management CIM (JSR77.9)

JSR77 J2EE Managed Objects hierarchy



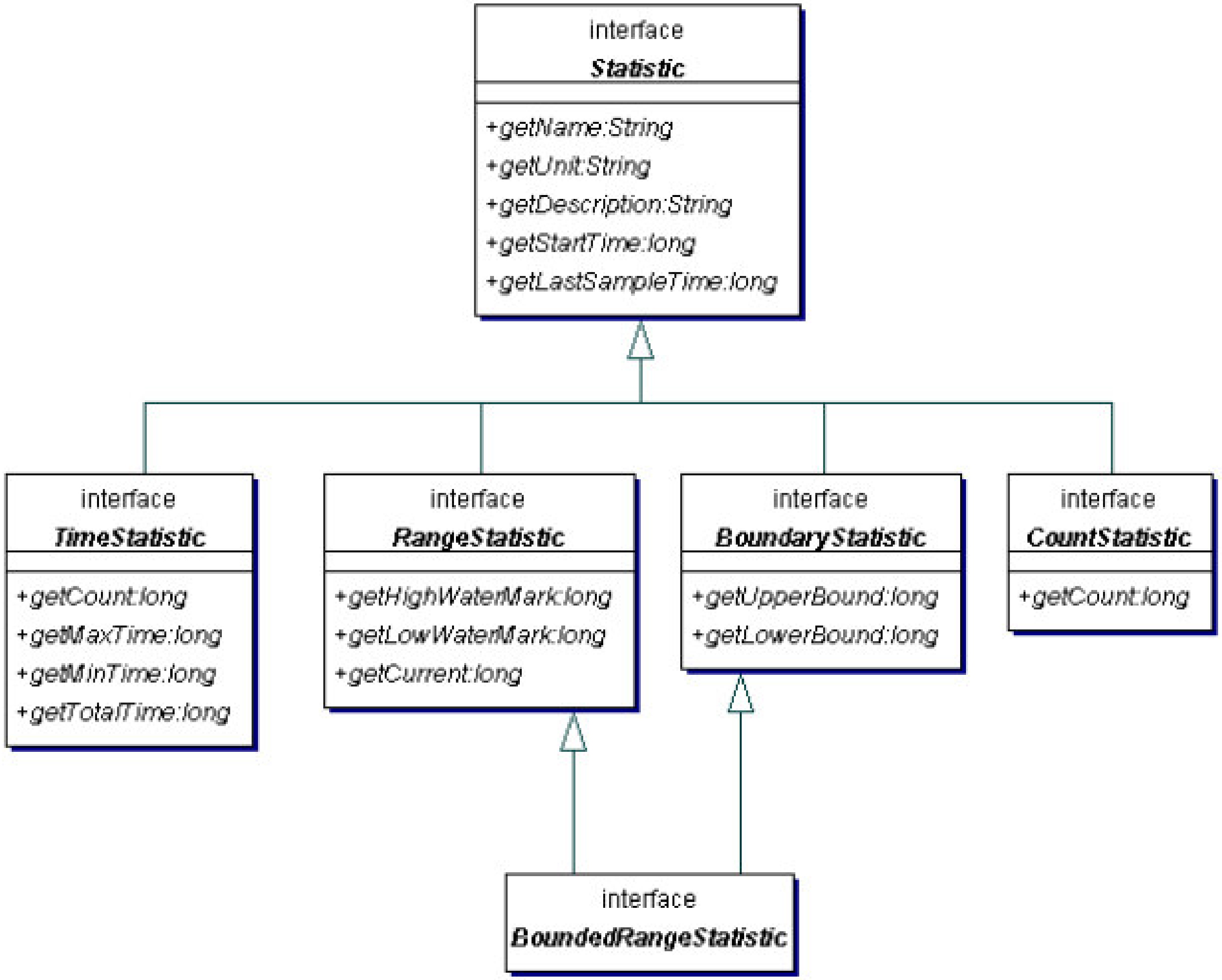
JSR77 J2EE Managed Objects ObjectNames

- Syntax
  - [domainName]:j2eeType=value,name=value,<parent-j2eeType>[,property=value]*
- Examples
  - FirstEverBank:j2eeType=J2EEDomain,name=FirstEverBank
  - FirstEverBank:j2eeType=J2EEServer,name=BankServer1
  - FirstEverBank:j2eeType=J2EEApplication,name=AccountsController,J2EEServer=BankServer1
  - FirstEverBank:j2eeType=EJBModule,name=BankAccount,J2EEApplication=AccountsController,J2EEServer=BankServer1
  - FirstEverBank:j2eeType=EntityBean,name=Account,EJBModule=BankAccount,J2EEApplication=AccountsController,J2EEServer=BankServer1

JSR77 Statitic

J2EEManagedObject	Required Stats interface
EJB	EJBStats
EntityBean	EntityBeanStats
JavaMail	JavaMailStats
JCAResource	JCAStats
JDBCResource	JDBCStats
JMSResource	JMSStats
JTAResource	JTAStats
JVM	JVMStats
MessageDrivenBean	MessageDrivenBeanStats
Servlet	ServletStats
SessionBean	SessionBeanStats
StatefulSessionBean	StatefulSessionBeanStats
StatelessSessionBean	StatelessSessionBeanStats
URLResource	URLStats

JSR77
Statitic (javax.management.j2ee.statistics)



JSR77 Management EJB Component (MEJB)

- Motivation
  - provides access to the managed object instances of all the available managed objects in one or more management domains
  - for all J2EE components (EJB, Servlet/JSP)
- Interfaces
  - Remote : javax.management.j2ee.Management
  - Home : javax.management.j2ee.ManagementHome
  - *+ class javax.management.j2ee.ListenerRegistration*
- JNDI subcontext
  - ejb/mgmt

JSR77 MEJB Management EJB Component

- Example
- Creating a MEJB session object from a J2EE application

```
Context ic = new InitialContext();
java.lang.Object objref = ic.lookup("java:comp/env/ejb/mgmt/MEJB");
ManagementHome home =
(ManagementHome)PortableRemoteObject.narrow(objref,ManagementHome.class);
Management mejb = home.create();
```

- Find all J2EEApplication managed objects on this system

```
ObjectName searchpattern = new ObjectName("*:j2eeType=J2EEApplication,*");
Set managed_object_set = mejb.queryNames(searchpattern, null);
System.out.println("found " + managed_object_set.size() + " matching objects.");
```

JSR77 MEJB Management EJB Component

- **retrieving attribute information from a set of managed objects**

```
Iterator managed_objects = managed_object_set.iterator();
while (managed_objects.hasNext()) {
ObjectName objectname = (ObjectName)managed_objects.next();
System.out.println(objectname);
/* get MBeanInfo and print the info */
MBeanInfo moi = mejb.getMBeanInfo(objectname);
MBeanAttributeInfo[] atts = moi.getAttributes();
for (int i = 0; i < atts.length; i++) {
  System.out.println("**** Attribute Info ****");
  System.out.println("Name " + atts[i].getName());
  System.out.println("Type " + atts[i].getType());
  System.out.println("isIs " + atts[i].isIs());
  System.out.println("isReadable " + atts[i].isReadable());
  System.out.println("isWritable " + atts[i].isWritable());
  System.out.println("value " + mejb.getAttribute(objectname, atts[i].getName()));
 }
}
```

JSR77
MEJB Management EJB Component

- invoke the method with the signature `String test(String,int)`

```
String sig1 = String.class.getName();
String sig2 = Integer.TYPE.getName();
String [] signature = new String [] {sig1, sig2};
Object [] params = new Object [] {"This is a Test", new Integer(77)};
String s = (String) mejb.invoke(appname, "test", params, signature);
```

- register an event listener

```
objectname = new ObjectName(mejb.getDefaultDomain()
    +":type=J2EEDomain,name=Hans’ J2EE Domain");
System.out.println("addNotificationListener( " + listener + " ) to " + objectname);
ListenerRegistration lr = mejb.getListenerRegistry();
lr.addNotificationListener(objectname, listener, null, "MEJBTester");
```

JSR77
J2EE Management SNMP (JSR77.8)

- MIB for J2EE servers
  - TODO

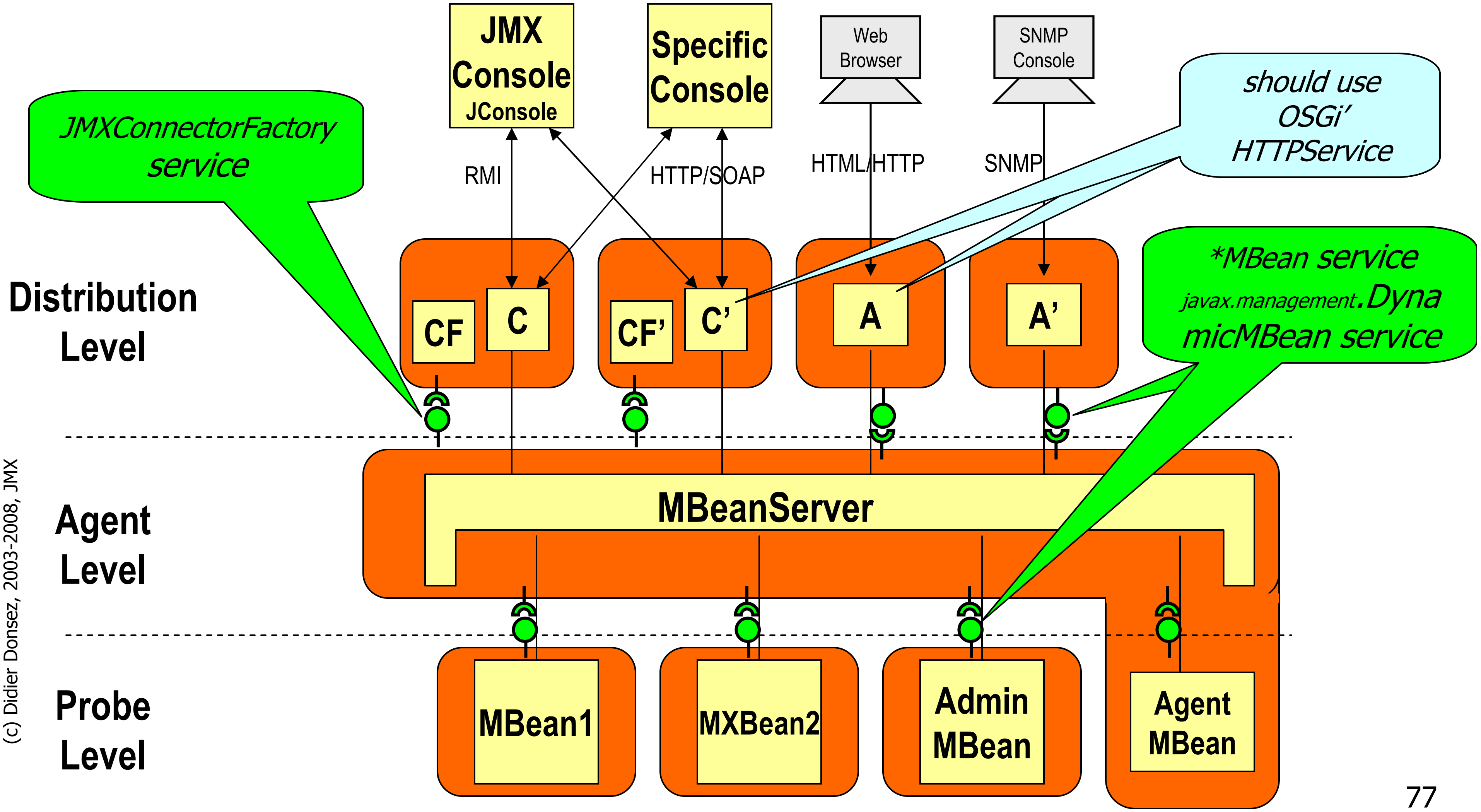
Non traité pour l'instant

- JMX avec WS-Manangement
  - En relation avec le JSR 262
- JSR 255
  - Cascading
  - Namespace

JMX and OSGi

- Principe
  - Le bundle JMXAgent démarre un MBeanServer
  - Chaque bundle enregistre un ou plusieurs services MBean
  - Le bundle JMXAgent récupère tout nouveau service MBean
    - filter="objectClass=*MBean"
  - Le bundle JMXAgent récupère tout service de connection ou d'adaptation
- Voir MOSGi du projet Apache Felix
  - http://felix.apache.org/site/mosgi-managed-osgi-framework.html

JMX and OSGi


JMX Console
JConsole
Specific Console
Web Browser
SNMP Console
JMXConnectorFactory service
should use OSGi' HTTPService
RMI
HTTP/SOAP
HTML/HTTP
SNMP
Distribution Level
CF
C
CF'
C'
A
A'
*MBean service javax.management.DynamicMBean service
Agent Level
MBeanServer
Probe Level
MBean1
MXBean2
Admin MBean
Agent MBean

JMX XDoclet (i)

- (JavaDoc) Doclet Annotations for JMX
- XDoclet ANT tasks and *subtasks*
  - <mbeaninterface> Generates a managed operation interface for standard MBeans
  - <mlet> Generates an mlet descriptor file
  - <jbossxmbean> Generates a deployment descriptor for JBossMX model MBeans (XMBeans)
  - <jbossxmldoc> Generates documentation for an MBean in DocBook format
  - <jbossxmlservicetemplate> Generates JBossMX service deployment descriptors
  - <mx4jdescription> Generates an MBean description class for deployment in MX4J
- See
  - http://xdoclet.sourceforge.net/xdoclet/index.html
  - http://xdoclet.sourceforge.net/xdoclet/tags/jmx-tags.html
  - http://www.manning-source.com/books/walls/walls_ch09.pdf

JMX XDoclet (ii)

- Example

```
package com.xdocletbook.jmx;
/**
* @jmx.mbean name="Memory:name=memoryMonitor"
* @jmx.mlet-entry archive="MemoryMonitorMBean.jar"
*/
public class MemoryMonitor implements MemoryMonitorMBean{
  public MemoryMonitor() {
  }
/**
* @jmx.managed-attribute description="Available memory"
*/
public long getAvailableMemory() {
  return Runtime.getRuntime().freeMemory();
}
/**
* @jmx.managed-attribute
*/
public long getTotalMemory() {
  return Runtime.getRuntime().totalMemory();
}
/**
* @jmx.managed-operation description="Garbage collection"
*/
public long gc() {
  return Runtime.getRuntime().gc();
}
}
```

JMX and JBoss

- Services in JBoss are (extended MBeans)
- Deployed by the JBoss MicroContainer
- Deployment descriptor (XML syntax)

Conclusion

Bibliography

- JCP
  - *JSR 3, Java Management Extensions Instrumentation and Agent Specification*
  - *JSR 160, Java Management Extensions Remote API*
- State of the art chapter in Noël de Palma' PhD thesis
  - http://sardes.inrialpes.fr/papers/files/01-DePalma-PhD.pdf
- Tutorial JMX of J2SE1.5
  - %JAVA_HOME%\docs\guide\jmx\tutorial\tutorialTOC.html
  - http://www.xmojo.org/products/xmojo/tutorials/docs/index.html
- Presentation
  - Eamonn McManus, « JMX », JavaOne 2003, Session 2074
- JMX Best Practices
  - http://java.sun.com/products/JavaManagement/best-practices.html
  - http://www.ftponline.com/javapro/2005_01/magazine/features/cpeltz/
- Wikipedia entry
  - http://en.wikipedia.org/wiki/Java_Management_Extensions

Bibliography

- Books
  - Marc Fleury et al, Jmx: Managing J2ee Applications with Java Management Extensions, Publ. SAMS; 1st edition (January 31, 2002), ISBN: 0672322889, 408 pages
  - Mike Jasnowski, JMX Programming, Publ: John Wiley & Sons; 1 edition (July 16, 2002), ISBN: 076454957X, 528 pages
  - Ben G. Sullins, Mark Whipple, Benjamin G. Sullins, Mark B. Whipple, JMX in Action, Publ. Manning Publications Company; (October 2002) , ISBN: 1930110561, 424 pages
  - J. Steven Perry, Java Management Extensions, Publisher: O'Reilly & Associates; 1st edition (June 15, 2002), ISBN: 0596002459, 312 pages
  - Heather Kreger, Ward K. Harold, Leigh Williamson, Java and JMX: Building Manageable Systems, Publ. Addison-Wesley Pub Co; 1st edition (December 30, 2002), ISBN: 0672324083, 592 pages
  - Benjamin G. Sullins and Mark B. Whipple, JMX in Action (Greenwich, CT: Manning, 2002).

Pour démarrer rapidement

- Avec le J2SE 1.5
  - http://java.sun.com/developer/technicalArticles/J2SE/jmx.html
- Charger JMX Reference Implementation
  et tester les exemples
  - http://java.sun.com/products/JavaManagement/download.html
- Puis pour appronfondir, le Tutoriel
  - http://java.sun.com/j2se/1.5.0/docs/guide/jmx/tutorial/tutorialTOC.html

Pour démarrer un peu moins rapidement

- MC4J
  - Console JMX GUI très complète
    - Personnalisé pour les principaux serveurs J2EE
  - Open-Source
  - http://mc4j.sf.net
- MX4J
  - Implémentation d'agent + adapter HTTP + … pour Java1.4 et -
  - Open-Source
  - http://mx4j.sf.net
- Une autre console
  - XtremeJ Management Console (plugin Eclipse) http://www.xtremej.com
- Une autre implémentation JMX open source
  - http://www.xmojo.org/