体验 GlassFish 的特色功能

蒋健， 王昱
Sun Microsystems, Inc.

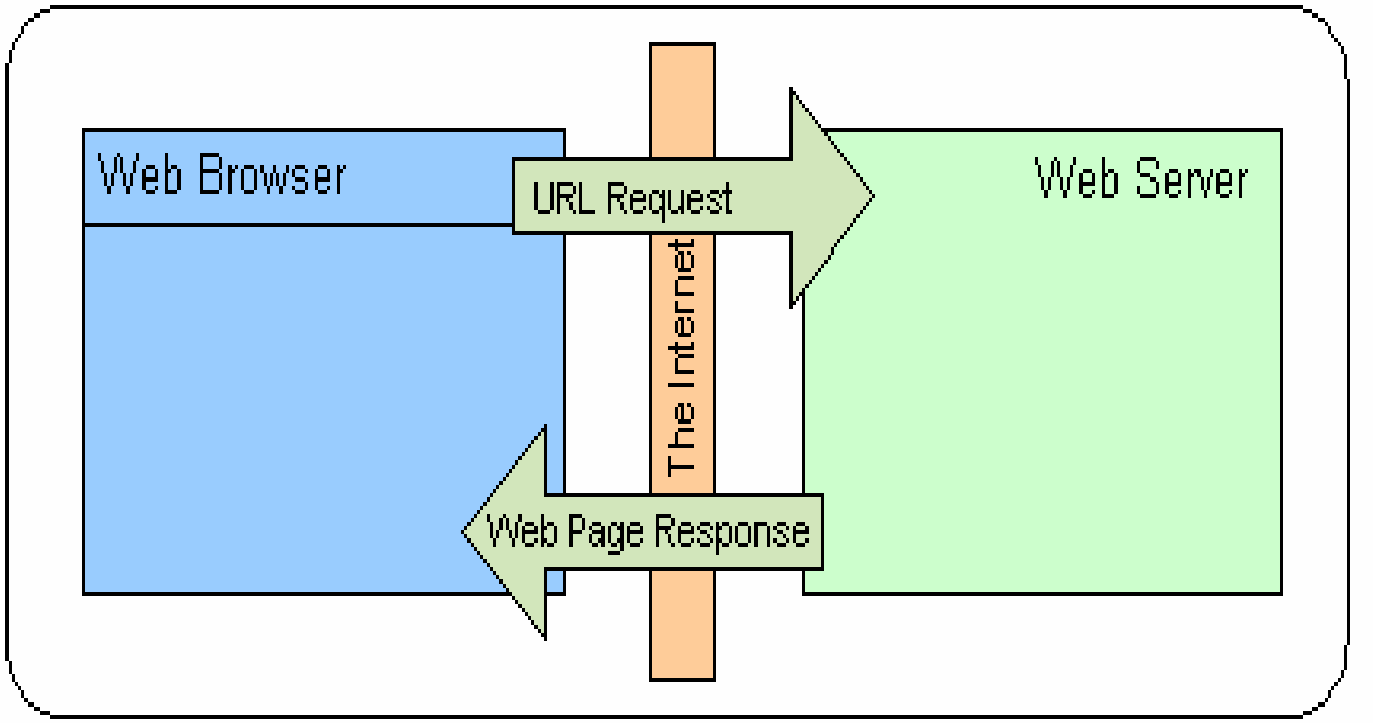
内容介绍

除了实现了 *JavaEE5* 的标准以外，*GlassFish* 还拥有许多非常出众的特点，这些特点在构建未来的企业应用非常有用，特别是对 *Web2.0* 的支持。在这个讲座中，我们来一一了解这些有趣的特点

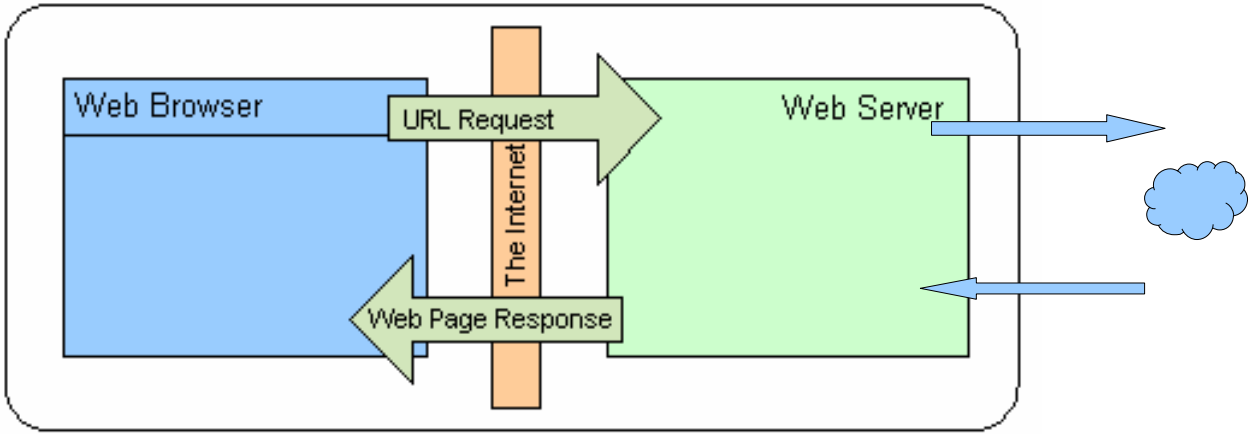
GlassFish 特色功能

- 异步请求处理
- 服务器推送技术
- 虚拟资源管理
- AMX
- 自管理
- CallFlow

HTTP同步请求处理


Web Browser
URL Request
Web Server
The Internet
Web Page Response

HTTP同步请求有时候满足不了...



- 请求处理需要调用外部服务
- 需要人工干预（经理批准）
- 例如：邮件显示

HTTP同步请求有时候满足不了...

- 通常的解决方案为 Polling
- 将业务逻辑拆分为两步
  - 业务执行
  - 状态查询
- Ajax 的应用（自动 polling ）
- Polling 的缺点
  - 白白消耗网络资源
  - 白白消耗 CPU 资源

GlassFish异步请求处理

- Polling 的解决方案的扩展性不好
- GlassFish通过NIO来实现ARP层
- ARP 允许暂时停止一个请求的处理，在某些条件满足后，重新执行
- 保持HTTP连接不断，但是并不是每个连接都会占用一个工作线程，来保证异步处理的性能和扩展性，这一点很重要。
- 例子：邮件显示
  - http://developers.sun.com/learning/javaoneonline/j1lab.jsp?lab=LAB-3360&yr=2007&track=3

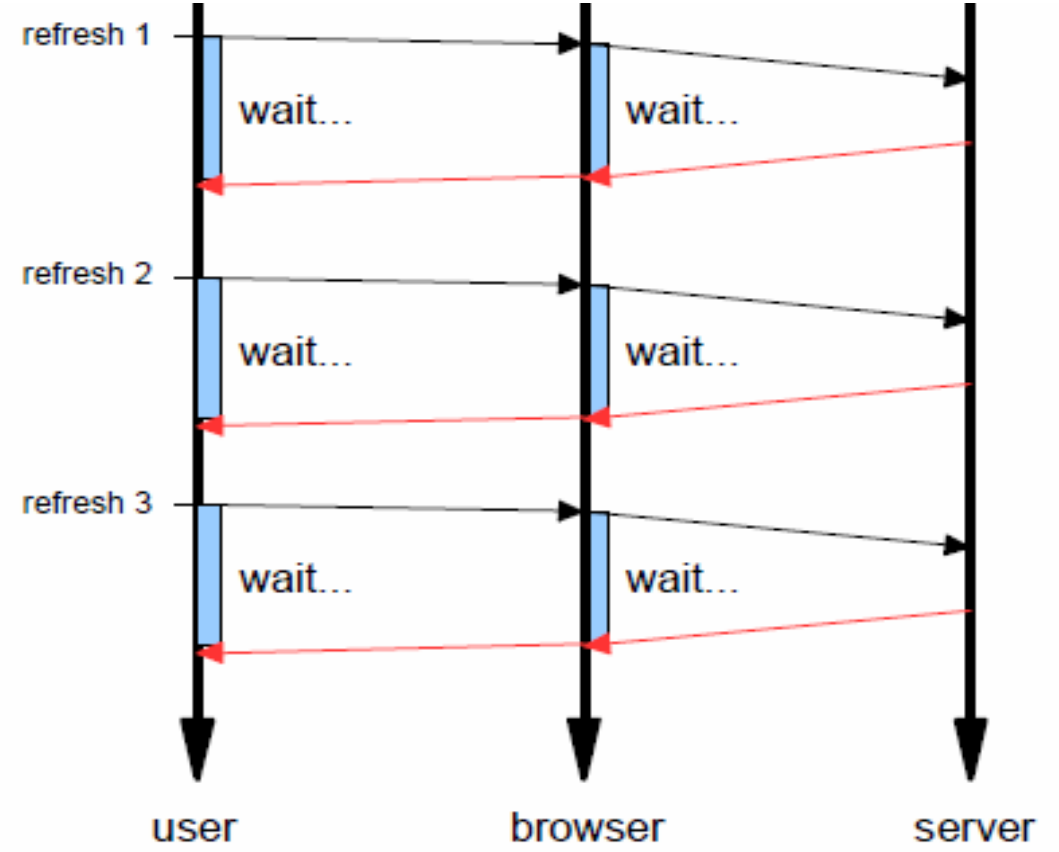
GlassFish 特色功能

- 异步请求处理
- 服务器推送技术
- 虚拟资源管理
- AMX
- 自管理
- CallFlow

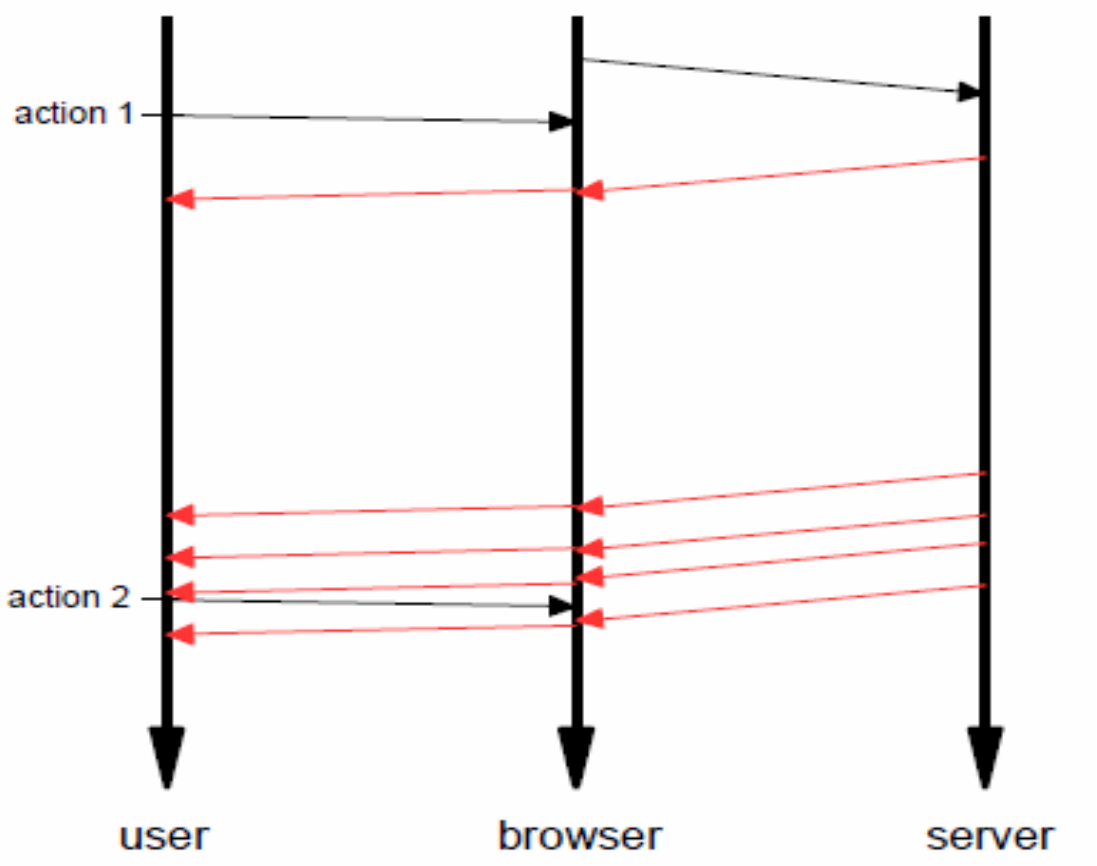
服务器推送技术 (Comet)

- B/S 应用逐步替代 C/S 应用
- Ajax 正在吞噬桌面应用的市场
  - 文字处理器
  - 邮件客户端
  - 相片管理及图形编辑
- 除了和硬件相关的 CAD 软件以及大型游戏
- Web 应用（ Ajax 应用）的一个致命的缺点：服务器端不能主动发消息
- 影响关键应用的响应能力，延迟时间长（股票系统）

传统 Web 的请求响应序列



Comet 请求的响应序列



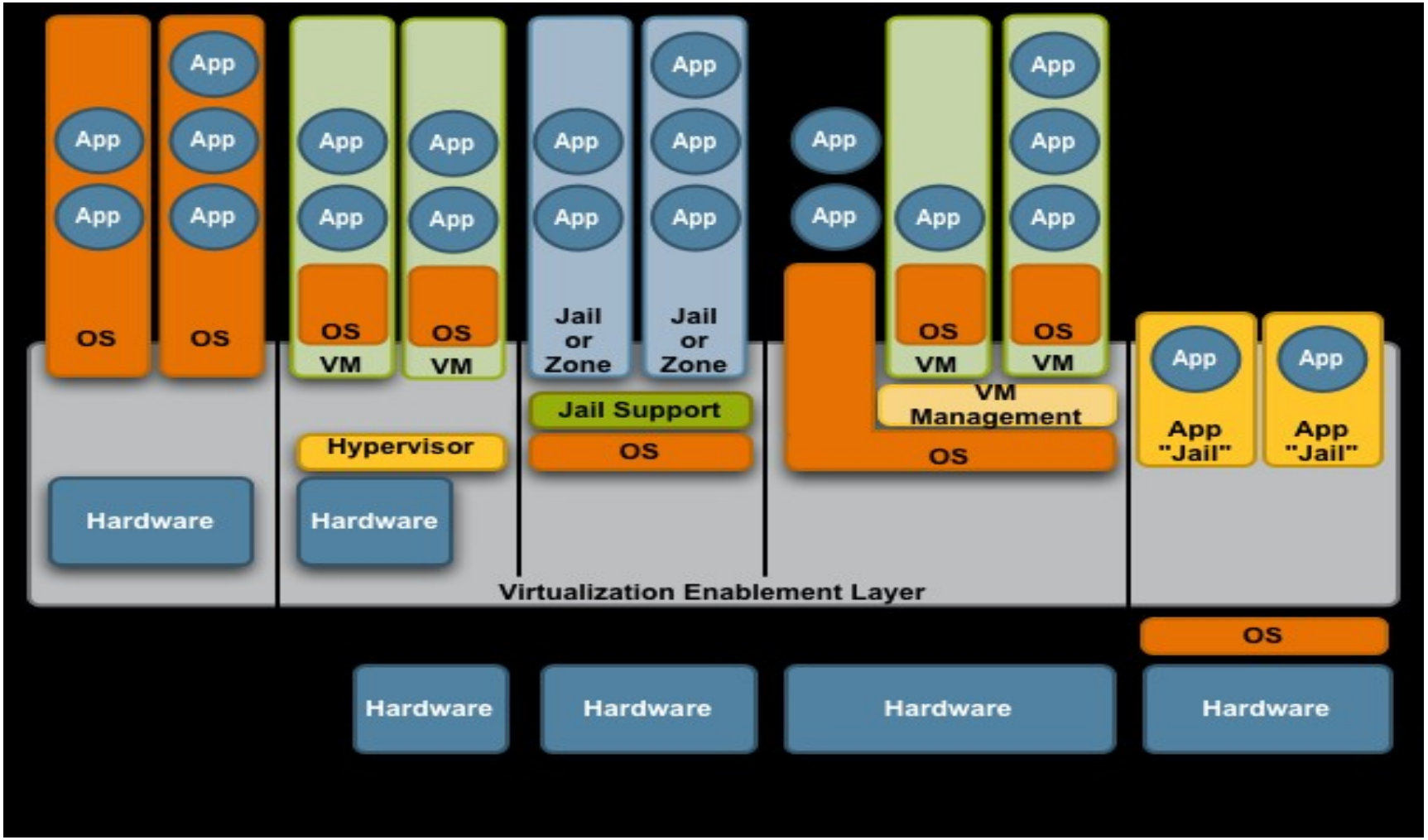
GlassFish数据推送技术

- 服务器端主动发送消息
- 没有使用Polling方法
- 使用了长时间保留的HTTP连接，减少延迟
- 基于 ARP（NIO）的高性能实现.
- 非常容易使用和扩展的API
- 兼容Cometd协议（http://cometd.com)

GlassFish 特色功能

- 异步请求处理
- 服务器推送技术
- 虚拟资源管理
- AMX
- 自管理
- CallFlow

资源虚拟化管理


App
OS
VM
Jail or Zone
Jail Support
Hypervisor
VM Management
App "Jail"
Hardware
Virtualization Enablement Layer

GlassFish的资源管理(RCM)

- GlassFish的 RCM与平台无关
- 应用层的资源分配的策略
- 比底层的虚拟技术更大的灵活性
- GlassFish目前支持两种规则：
  - 保留一定百分比的内存资源
  - 保留一定百分比的处理线程资源

GlassFish 特色功能

- 异步请求处理
- 服务器推送技术
- 虚拟资源管理
- AMX
- 自管理
- CallFlow

JMX 基础

- JMX(Java Management Extensions)
  - **管理** *可被 Java 抽象的各类应用、系统和网络*
  - **扩展** *管理和监控方案，使之模式化和标准化*
  - *MBean(Management Bean)：封装可管理资源*
  - *OjbectName:MBean 在 MBean 服务器中唯一标识*
- 好处
  - *标准化 — 条形码*
  - *轻量级 — 脚手架*
- 应用领域
  - 配置管理，监控和统计

GlassFish 对 JMX 的应用和扩展

- JMX 是 GlassFish 管理架构的基础
- 多种方式的支持
  - 管理控制台
  - 命令行工具
    - Dotted Name： com.sun.appserv:type=jdbc-connection-pool,name=mypool,category=config
    - ObjectName:server.resources.jdbc-connection-pool.mypool
  - 第三方工具
  - 编程方式
    - JMX
    - AMX(*AppServer Management Extensions*)

标准 JMX 方式 vs AMX 方式

JMX标准方式

```
//create the connection
JMXServiceURL url = new JMXServiceURL(
   "service:jmx:rmi:///jndi/rmi://localhost:8686/jmxrmi");
java.util.Map env = new java.util.Hashtable();
String[] creds = {"admin","adminadmin"};
env.put(JMXConnector.CREDENTIALS,creds);
JMXConnector connector =
   JMXConnectorFactory.connect(url,env);
MBeanServerConnection mbsc =
   connector.getMBeanServerConnection();

//locate the mbean
ObjectName mbeanName =new
   ObjectName("com.sun.appserv:type=jdbc-
   connection-pool,
   name=mypool,category=config");


//print mbean attribute value
System.out.println(mbsc.getAttribute(mbeanName,"idl
   e-timeout-in-seconds"));
```

AMX的DCP方式

```
//create the connection
AppserverConnectionSource conn = new
   AppserverConnectionSource(
   AppserverConnectionSource.PROTOCOL_RMI, host,
   port, user, password, tlsParams, null);
    conn.getJMXConnector( true );


//locate the mbean
DomainRoot  domainRoot = conn.getDomainRoot();
Map<String,JDBCConnectionPoolConfig> pools =
domainRoot.getDomainConfig().getJDBCConnectionP
   oolConfigMap();
 JDBCConnectionPoolConfig mypool =
   (JDBCConnectionPoolConfig)pools.get("mypool");


//print mbean attribute value
System.out.println(mypool.getIdleTimeoutInSeconds());
```

特性展示—查看管理GlassFish 组件

- 通过命令行工具 asadmin
- 通过管理控制台
- 通过第三方工具 JConsole
- 通过 JMX 代码
- 通过 AMX 代码

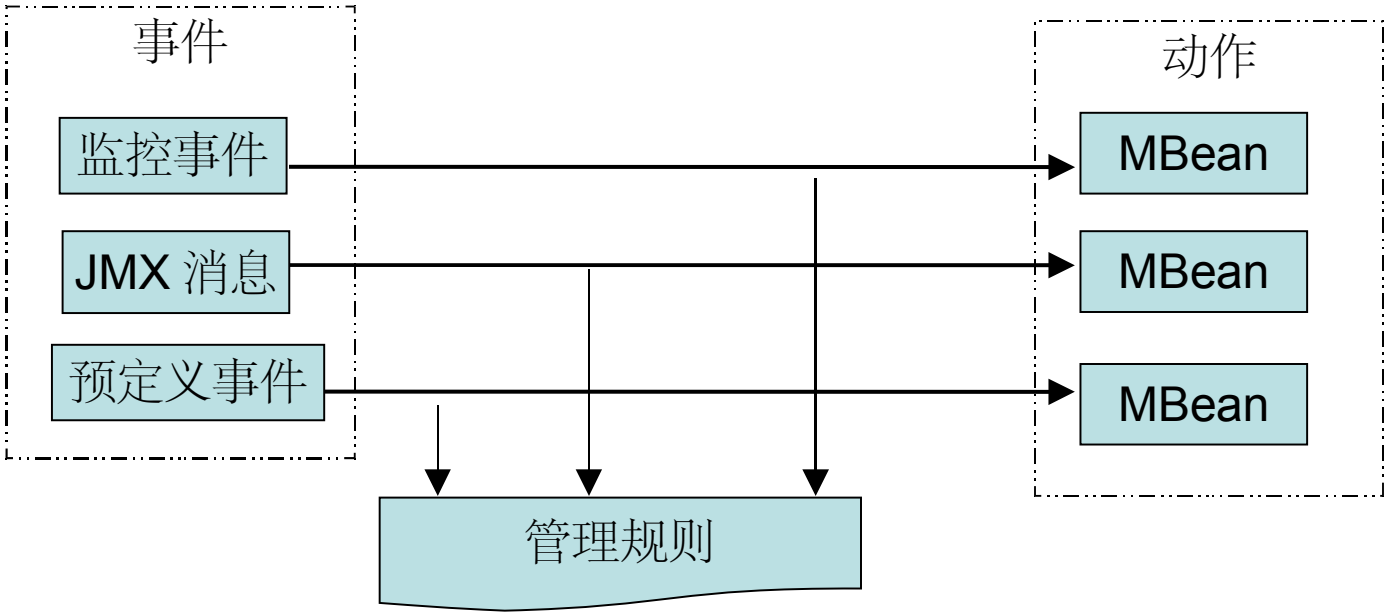
GlassFish 特色功能

- 异步请求处理
- 服务器推送技术
- 虚拟资源管理
- AMX
- 自管理
- CallFlow

自管理

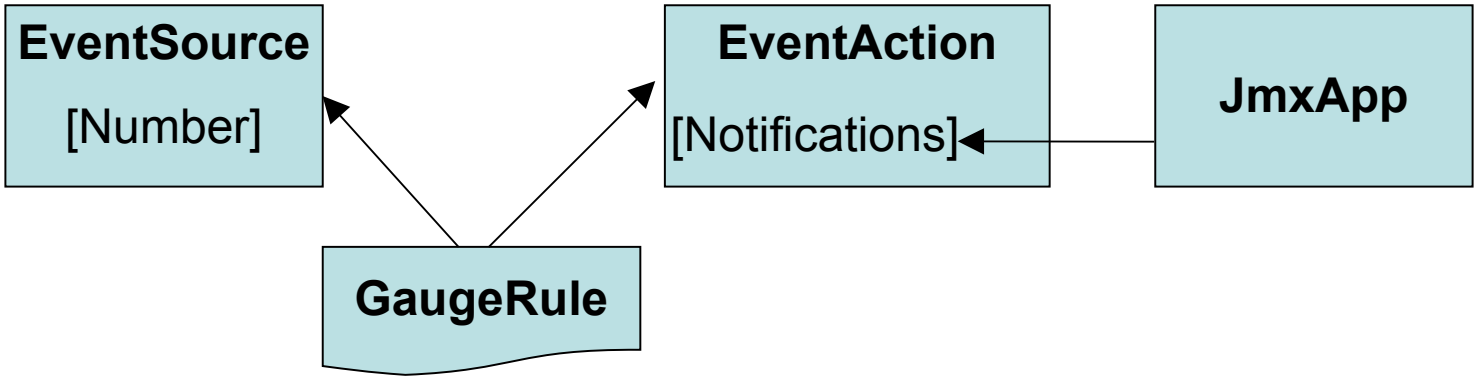
- JMX 事件机制的扩展应用
- 用于自调整，自配置，自恢复
- 好处
  - 定制简单
  - 扩展方便
- 管理规则的定制和运行

管理规则=事件+动作




```
<management-rule enabled="false" name="gaugerule">
   <event level="INFO" record-event="true" type="monitor">
      <property name="observedobject" value="user:impl-class-
 name=EventSource,name=EventSource,server=server"/>
      <property name="observedattribute" value="Number"/>
      <property name="monitortype" value="gaugemonitor"/>
...
      </event>
      <action action-mbean-name="EventAction"/>
</management-rule>
```

特性展示—自管理应用



1. 部署 MBean’EventSource’
2. 部署 MBean ‘EventAction’
3. 创建数值型监控器的管理规则 ‘GaugeRule’
4. 运行客户端监控程序 ‘JmxApp’
5. 通过 JConsole 触发管理规则

GlassFish 特色功能

- 异步请求处理
- 服务器推送技术
- 虚拟资源管理
- AMX
- 自管理
- CallFlow

CallFlow —内置的性能探查器

- 收集细致到方法级别的运行运行程序信息，用于性能调优和程序调试
- 特点
  - 对系统性能的影响轻微
  - 可以监控到其它 Profiler 无法访问到的信息
  - 提供 AMX 开发接口，方便数据挖掘

特性展示— CallFlow 监测 Web 应用

- 信息收集
  - 各类容器中消耗的时间
  - 被调用的方法及其响应堆栈
  - 关联的应用程序名和模块名
  - 异常出处
- 图形化的结果显示
- 数据查看和过滤

资源

- 书《GlassFish —开源的 Java EE 应用服务器》
- JavaOne 2007 上的动手实验室 3360 “Taste Special Features of Glassfish ”
- 博客：

http://developers.sun.com.cn/blog/java/