Monitoring Web Services with the J-OCM Monitoring System

Lechoslaw Trębacz[1], Włodzimierz Funika[2], Piotr Handzlik[3], Marcin Smętek[2]

[1] Department of Computer Methods in Metallurgy, AGH, Kraków, Poland

[2] Institute of Computer Science, AGH, Kraków, Poland

[3] Department of Physical Chemistry and Electrochemistry, AGH, Kraków, Poland

1 .Introduction

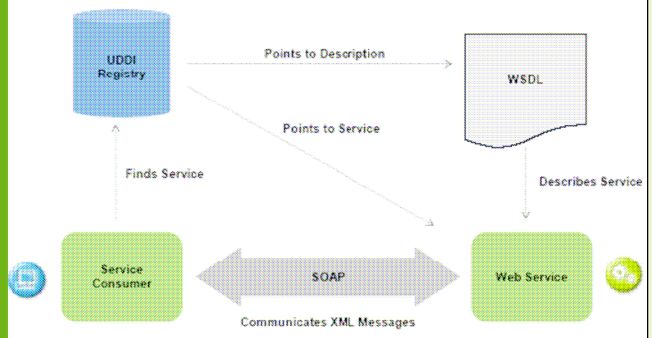
The *monitoring* of an application is aimed at observation, analysis and manipulation of the execution of an application.
This research focuses on a Java-related monitoring system for distributed applications which are oriented towards the use of web services. We focus on the issue of building a monitoring platform based on a well-defined interface between a monitoring system organized as middleware and tools, OMIS.

2. Goals

Provide an extension to JOCM system, which enables the monitoring of Web Services-based applications.
JOCM is a monitoring system which is used to monitor Java distributed applications. It complies with the OMIS interface.

3. Web Services

A Web Service is programmable application logic accessible via standard Internet protocols.
It uses **SOAP** (Simple Object Access Protocol), **WSDL** (Web Services Description Language), **UDDI** (Universal Description, Discovery and Integration).



4. Approach to the monitoring Web Services

The approach we use for the monitoring of Web Services in the J-OCM relies on:
• dynamic instrumentation of AXIS classes. At the start and the end of each stage of processing SOAP messages there are placed sensors (in AXIS) which generate events and convey them to the agent,
• JVM Tool Interface and JNI to obtain information about the state of Java Virtual Machine (for example: information about a class of Web Services)

5. Identification of request / reply

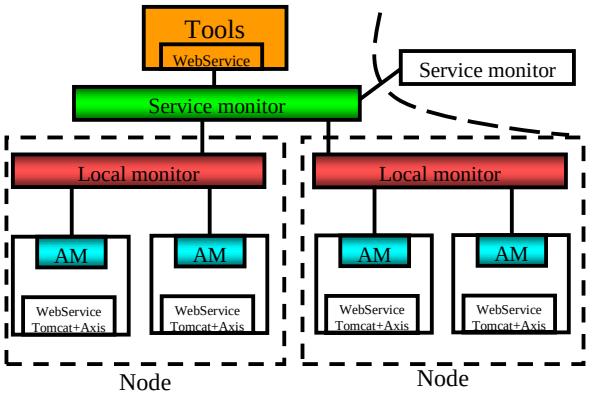
The problem of associating the events raised by the client and those raised by the server can be solved by sending additional information (the token of Web Service, additional request options / reply data) from the client to the server.

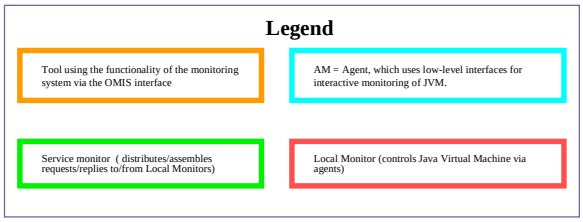
6. Role of tokens

Each monitored object (node, jvm, Web Services, operations of WS) is identified by a token – object's name as well as a platform independent way of addressing, e.g. the *web service* token jvm_ws_2_j_1_n_1 allows to find out where it is running (on jvm 1 on node 1).

7. Architecture

Features:
•Use of J-OMIS Interface,
•Extension to J-OCM,
•Web Service monitoring,
•Use of Java 1.5 and JVM Tool Interface,
•Underlied by Tomcat and AXIS as platform





8. Start-up procedure and query in tool

In order to start the monitoring of an Web Service application we must execute the following commands:

```
java –jar agentlib:jvmlm bootstrap.jar
```

where `bootstrap.jar` is a Jar with Axis classes

A query from a tool to the monitoring system looks like:

:**jvm_ws_get_tokens**([jvm_j_1_n_1]) – request for all Web Services tokens on 1 JVM,

jvm_ws_event_operationStart([jvm_j_1_n_1]):print(["Operation Start at",$time, $wsStr, $ws, $opStr, $op]) – a request for the echo of events (with time, name and token WS, and name and token operation of WS), when a WS starts an operation on JVM 1 (begin of execution of a method of WS)

9. References

1. Ludwig T., Wismueller R., Sunderam V., Bode A.: OMIS – Online Monitoring Interface Specification (Version 2.0). Shaker Verlag, Aachen, vol. 9, LRRTUM Research Report Series, 1997
2. M. Bubak, W. Funika, M.Smetek, Z. Kilianski, and R. Wismüller: Architecture of Monitoring System for Distributed Java Applications. In: Dongarra, J., Laforenza, D., Orlando, S. (Eds.), Proc. Euro PVM/MPI 2003, Venice, Italy, Sept. 29 - Oct. 2 2003, LNCS 2840, pp. 447-454, Springer, 2003
3. http://ws.apache.org/axis/java/architecture-guide.html