Dynamo-AOP user manual

Domenico Bianculli
domenico.bianculli@lu.unisi.ch

Carlo Ghezzi
ghezzi@elet.polimi.it

July 31, 2007

Dynamo-AOP[1] is a framework for monitoring functional properties of external services which a BPEL process interacts with, to realize a composite service. Its architecture is based on the dynamic aspectization of the BPEL engine executing the monitored service compositions, achieved by using AspectJ as an AOP language.

1 Installation and usage

1.1 Installation

The following components need to be installed in order to run Dynamo-AOP:

- the Apache Tomcat servlet container (assumed to be installed in the directory `$TOMCAT-DIR` and running on port `7080`), available at http://tomcat.apache.org/
- the JBoss application server (assumed to be installed in the directory `$JBOSS-DIR` and running on port `8080`), available at http://www.jboss.org/
- the ActiveBPEL BPEL engine, available at http://www.active-endpoints.com
- (optional) a mail server, to support the `notify` recovery strategy.

The current version of Dynamo-AOP has been tested using Apache Tomcat ver. 5.5.23, JBoss Application Server ver 4.2, ActiveBPEL ver 2.1.
The following libraries are also required:

- ANTLRv2, available at http://www.antlr2.org/
- Apache Axis, available at http://ws.apache.org/axis/
- Apache XMLBeans, available at http://xmlbeans.apache.org/

[1] Part of this work has been developed by Lorenzo D'Ercole and Luca Gallupi, as part of their master theses, under the supervision of Luciano Baresi and Sam Guinea.

- Apache Xerces 2, available at http://xerces.apache.org/xerces2-j/index.html
- Castor, available at http://www.castor.org
- Jakarta Commons Discovery, available at http://commons.apache.org/discovery/
- Jakarta Commons Logging, available at http://commons.apache.org/logging/
- CommonJ Timer and Work Manager for Application Servers, available at http://dev2dev.bea.com/wlplatform/commonj/twm.html
- Javamail, available at http://java.sun.com/products/javamail/
- JAXP, available at https://jaxb.dev.java.net/
- Jaxen, available at http://jaxen.org/
- JAX-RPC, available at https://jax-rpc.dev.java.net/
- JBoss EJB3, available at http://labs.jboss.com/jbossejb3/
- JBoss Web Services, available at http://labs.jboss.com/jbosssws/
- Saxon XSLT processor, available at http://saxon.sourceforge.net/
- StAX, available at http://stax.codehaus.org/
- Sun Java Streaming XML Parser (JSR 173), available at http://java.sun.com/webservices/docs/1.5/sjsxp/ReleaseNotes.html
- WSDL for Java API, available at http://sourceforge.net/projects/wsdl4j
- XML Pull Parser (XPP), available at http://www.extreme.indiana.edu/xgws/xsoap/xpp/
- XSUL, available at http://www.extreme.indiana.edu/xgws/xsul/

To actually install the Dynamo-AOP monitoring framework you have to:

- copy from the Dynamo-AOP distribution `ConfigurationManager.jar`, `HistoricalVariable.jar`, `MonitorLogger.jar` in the directory `$JBOSS-DIR/server/default/deploy`.
- copy from the Dynamo-AOP distribution `ae_rtbpel.jar`, in the directory `$TOMCAT-DIR/shared/libs`, overwriting the original ActiveBPEL file.
- copy from the Dynamo-AOP distribution `invoker.jar` in the directory `$TOMCAT-DIR/common/libs`.
- install a copy of `antlr-2.7.6.jar`, `antlrdebug_1.0.0.jar`, `jaxb-api.jar`, `jaxb-impl.jar`, `jaxb-xjc.jar`, `jaxb1-impl.jar`, `mail.jar`, `jsr173_api.jar`, `saxon8-dom.jar`, `saxon8-jdom.jar`, `saxon8-sql.jar`, `saxon8-xom.jar`, `saxon8-xpath.jar`, `saxon8.jar`, `xbean_path.jar`, `xbean.jar`, `xmlpublic.jar`, `aspectjrt.jar`, `xpp3-1.1.3.4.M.jar`, `xsul-2.0.9_2.jar`, `stax-1.1.1.jar` in the directory `$TOMCAT-DIR/common/libs`.

- install a copy of `axis.jar`, `commons-logging.jar`, `commons-discovery-0.2.jar`, `jaxrpc.jar`, `saaj.jar`, `wsdl4j-1.5.1.jar`, `castor-0.9.6-xml.jar`, `commonj-twm.jar`, `jaxen-1.1-beta-8.jar`, `ssaj-api.jar`, `xercesImpl.jar`, `saxon8-dom.jar` in the directory `$TOMCAT-DIR/shared/libs`.

1.2 Usage

1. launch the JBoss application server using the command `$JBOSS-DIR/bin/run.sh` and wait for the completion of the starting phase; check that JBoss is running by typing in your browser `http://localhost:8080`: you should see the start page of the application server.

2. launch the Apache Tomcat servlet container using the command `$TOMCAT-DIR/bin/catalina.sh run` and wait until the message "ActiveBPEL In-Memory Configuration Started" is displayed on the console. Check that Tomcat is running by typing in your browser `http://localhost:7080`. To check that ActiveBPEL is running, visit `http://localhost:7080/BpelAdmin`: you should see the ActiveBPEL administration tool. On it, click on Configuration and uncheck the box labelled "Validate Input/Output messages against schema".

3. deploy your BPEL process, as illustrated in ActiveBPEL user guide.

4. configure monitoring for the deployed process. This step assumes an interaction with the ConfigurationManager, which — in the first prototype of Dynamo-AOP — is done by directly accessing the API of the component, as shown in the "Dynamo Supervision Manager" application, also distributed within the framework (see next section).

2 Example

In the software distribution, you will find a complete web application that can be used to see how the monitoring framework works.

The application contains a BPEL process, `PizzaDeliveryCompany`, which you should deploy in the BPEL engine; at the end of the deployment you should see the process listed in the Deployed Processes section of the ActiveBPEL control panel, as shown in Figure 1. Before using the demo application, you should deploy some web services in the application server, by copying all the `*.jar` file from the `demo` directory to `$JBOSS-DIR/server/default/deploy`. In the same directory, you should also copy `dynamo.war`, which contains the "Dynamo Supervision Manager", a web application developed to to control the behavior of the monitoring framework; it can be reached at `http://localhost:8080/dynamo`.

The main page of this application contains two links: one to set the properties of the monitored processes and the other to access the Dynamo-AOP demo, also available directly at `http://localhost:8080/dynamo/DemoManager.jsp`. This page contains the various steps through which you can interact with the BPEL process. The execution of step #1 will attach some monitoring rules to the process, as shown in Figure 2. The execution of the monitor process can then be monitored on the output console of Tomcat, as shown in Figure 3. The

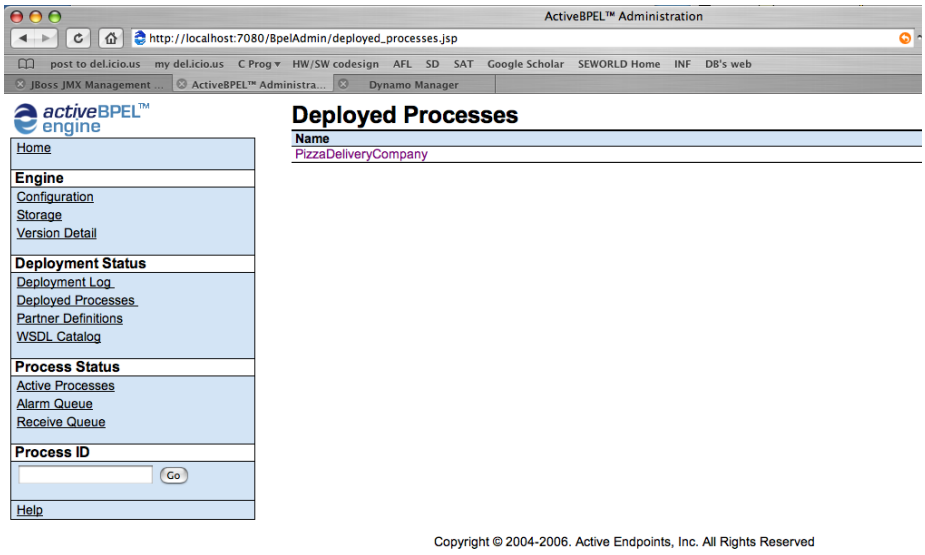

Figure 1: Process `PizzaDeliveryCompany` deployed successfully.

structure and the priority of the rules attached to a monitored process can be modified using the "Dynamo Supervision Manager", as shown in Figure 4.

3 WS-CoL grammar

⟨analyzer⟩ ::⇒ ⟨rules⟩ | ⟨recovery⟩
⟨recovery⟩ ::⇒ ⟨complete-strategy⟩ | strategy
⟨complete-strategy⟩ ::⇒ ⟨ifStrategy⟩ ⟨elseIfStrategy⟩* ⟨elseStrategy⟩$^?$
⟨ifStrategy⟩ ::⇒ **if** ⟨condition⟩ ⟨strategy⟩
⟨elseIfStrategy⟩ ::⇒ **elseif** ⟨condition⟩ ⟨strategy⟩
⟨elseStrategy⟩ ::⇒ **else** ⟨strategy⟩
⟨condition⟩ ::⇒ **(** ⟨rules⟩ **)**
⟨strategy⟩ ::⇒ **{** ⟨steps⟩ **}**
⟨steps⟩ ::⇒ ⟨rec-step⟩ (**or** ⟨rec-step⟩)*
⟨rec-step⟩ ::⇒ ⟨actions⟩
⟨actions⟩ ::⇒ action (**and** ⟨action⟩)*
⟨action⟩ ::⇒ ⟨identifier⟩ **(** ⟨list⟩$^?$ **)**
⟨rules⟩ ::⇒ ⟨sub-rule⟩ ((==> | <== | <==>) ⟨sub-rule⟩)* **;**
⟨sub-rule⟩ ::⇒ ⟨and-expression⟩ (|| ⟨and-expression⟩)*
⟨and-expression⟩ ::⇒ ⟨equals-expression⟩ (&& ⟨equals-expression⟩)*
⟨equals-expression⟩ ::⇒ ⟨relational-expression⟩ ((== | **!=**) ⟨relational-expression⟩)$^?$
⟨relational-expression⟩ ::⇒ ⟨operator-expression⟩ ((> | >= | < | <=) ⟨operator-expression⟩)$^?$
⟨operator-expression⟩ ::⇒ ⟨basic-expression⟩ ((+ | **-** | ***** | / | %) ⟨basic-expression⟩)*
⟨basic-expression⟩ ::⇒ ⟨dot-expression⟩ | ⟨variable⟩ | ⟨exists⟩ | ⟨forall⟩ | ⟨let⟩ | ⟨store⟩ | ⟨avg⟩ | ⟨min⟩ | ⟨max⟩ | ⟨sum⟩ | ⟨product⟩ | **true** | **false** | ⟨NUMBER⟩ | ⟨string-value⟩
⟨forall⟩ ::⇒ **(forall $** ⟨identifier⟩ **in** ⟨variable⟩ **;** ⟨sub-rule⟩ **)**

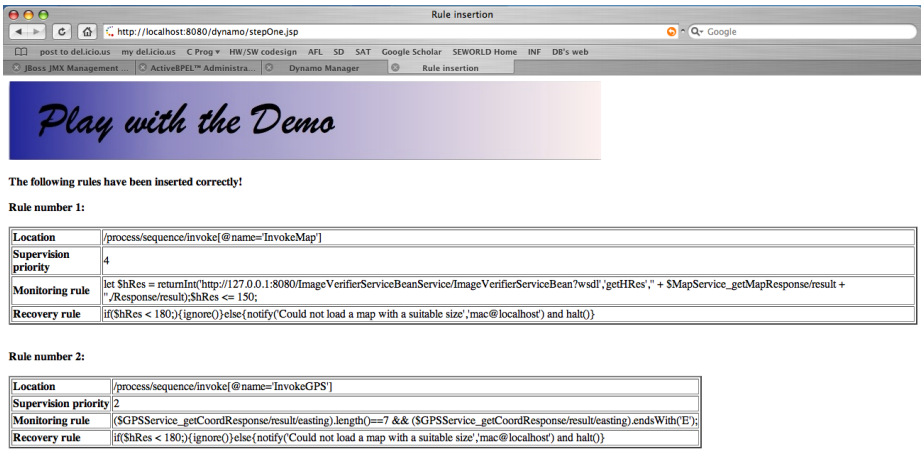

Figure 2: The result of inserting two monitoring rules.

⟨exists⟩ ::⇒ **(exists $** ⟨identifier⟩ **in** ⟨variable⟩ **;** ⟨sub-rule⟩ **)**
⟨dot-expression⟩ ::⇒ ⟨variable⟩ **.** ⟨identifier⟩ **(** ⟨list⟩$^{?}$ **)**
⟨let⟩ ::⇒ **let $** ⟨identifier⟩ **=** ⟨sub-rule⟩
⟨store⟩ ::⇒ **store $** ⟨identifier⟩ **=** ⟨sub-rule⟩
⟨avg⟩ ::⇒ **(avg $** ⟨identifier⟩ **in** ⟨variable⟩ **;** ⟨sub-rule⟩ **)**
⟨sum⟩ ::⇒ **(sum $** ⟨identifier⟩ **in** ⟨variable⟩ **;** ⟨sub-rule⟩ **)**
⟨min⟩ ::⇒ **(min $** ⟨identifier⟩ **in** ⟨variable⟩ **;** ⟨sub-rule⟩ **)**
⟨max⟩ ::⇒ **(max $** ⟨identifier⟩ **in** ⟨variable⟩ **;** ⟨sub-rule⟩ **)**
⟨product⟩ ::⇒ **(product $** ⟨identifier⟩ **in** ⟨variable⟩ **;** ⟨sub-rule⟩ **)**
⟨variable⟩ ::⇒ **(** (⟨ivar⟩ | ⟨evar⟩ | ⟨hvar⟩) **)** | (⟨ivar⟩ | ⟨evar⟩ | ⟨hvar⟩)
⟨ivar⟩ ::⇒ **$** ⟨identifier⟩ ⟨xpath-expression⟩$^{?}$
⟨evar⟩ ::⇒ ⟨returnType⟩ **(** ⟨string-value⟩ **,** ⟨string-value⟩ **,** ⟨string-value⟩ **,** ⟨xpath-expression⟩ **)**
⟨returnType⟩ ::⇒ **returnInt** | **returnBool** | **returnString**
⟨hvar⟩ ::⇒ **retrieve (** ⟨string-value⟩ (**,** ⟨string-value⟩)$^{?}$ (**,** ⟨NUMBER⟩) **,** ⟨xpath-expression⟩ **,** ⟨NUMBER⟩ **, $** ⟨identifier⟩ (**,** ⟨NUMBER⟩) **)**
⟨alias⟩ ::⇒ **$** ⟨identifier⟩
⟨list⟩ ::⇒ ⟨sub-rule⟩ (**,** ⟨sub-rule⟩)*
⟨string-value⟩ ::⇒ ⟨sub-string-value⟩ (**+** ⟨sub-string-value⟩)*
⟨sub-string-value⟩ ::⇒ ⟨identifier⟩ | ⟨literal⟩ | ⟨variable⟩
⟨xpath-expression⟩ ::⇒ ⟨union-expression⟩
⟨location-path⟩ ::⇒ ⟨absolute-location-path⟩ | ⟨relative-location-path⟩
⟨absolute-location-path⟩ ::⇒ (/ | //) (⟨i-relative-location-path⟩ | ε)
⟨relative-location-path⟩ ::⇒ ⟨i-relative-location-path⟩
⟨i-relative-location-path⟩ ::⇒ ⟨step⟩ ((/ | //) ⟨step⟩)*
⟨step⟩ ::⇒ ((⟨axis⟩ | ε) (((((⟨identifier⟩ **:**)$^{?}$ (⟨identifier⟩ | *****))) | ⟨special-step⟩) ⟨predicates⟩*) | ⟨abbr-step⟩ ⟨predicates⟩*
⟨special-step⟩ ::⇒ **processing-instruction (** ⟨identifier⟩$^{?}$ **)** | (**comment** | **text** | **node**) **( )**
⟨axis⟩ ::⇒ ⟨identifier⟩ **::** | **@**
⟨predicate⟩ ::⇒ **{** ⟨predicate-expr⟩ **}**
⟨predicate-expr⟩ ::⇒ ⟨expr⟩
⟨expr⟩ ::⇒ ⟨or-expr⟩
⟨primary-expr⟩ ::⇒ ⟨variable-reference⟩ | **(** ⟨expr⟩ **)** | ⟨literal⟩ | ⟨number⟩

Figure 3: Output console of the monitored process

| ⟨function-call⟩
⟨literal⟩ ::⇒ ⟨LITERAL⟩
⟨number⟩ ::⇒ ⟨NUMBER⟩
⟨variable-reference⟩ ::⇒ **$**⟨identifier⟩
⟨function-call⟩ ::⇒ ⟨identifier⟩ **(** ⟨arg-list⟩$^{?}$ **)**
⟨arg-list⟩ ::⇒ ⟨argument⟩ (**,** ⟨argument⟩)*
⟨argument⟩ ::⇒ ⟨expr⟩
⟨union-expr⟩ ::⇒ ⟨path-expr⟩ (| ⟨path-expr⟩)*
⟨path-expression⟩ ::⇒ ⟨location-path⟩ | ⟨filter-expr⟩ ⟨absolute-location-path⟩$^{?}$
⟨filter-expr⟩ ::⇒ ⟨primary-expr⟩ ⟨predicate⟩$^{?}$
⟨or-expr⟩ ::⇒ ⟨and-expr⟩ (**or** ⟨and-expr⟩)*
⟨and-expr⟩ ::⇒ ⟨equality-expr⟩ (**and** ⟨equality-expr⟩)*
⟨equality-expr⟩ ::⇒ ⟨relational-expr⟩ ((= | **!**=) ⟨relational-expr⟩)$^{?}$
⟨relational-expr⟩ ::⇒ ⟨additive-expr⟩ ((< | <= | > | >=) ⟨additive-expr⟩)$^{?}$
⟨additive-expr⟩ ::⇒ ⟨mult-expr⟩ ((+ | **-**) ⟨mult-expr⟩)$^{?}$
⟨mult-expr⟩ ::⇒ ⟨unary-expr⟩ ((***** | /) ⟨unary-expr⟩)$^{?}$
⟨unary-expr⟩ ::⇒ ⟨union-expr⟩ | **-** ⟨unary-expr⟩

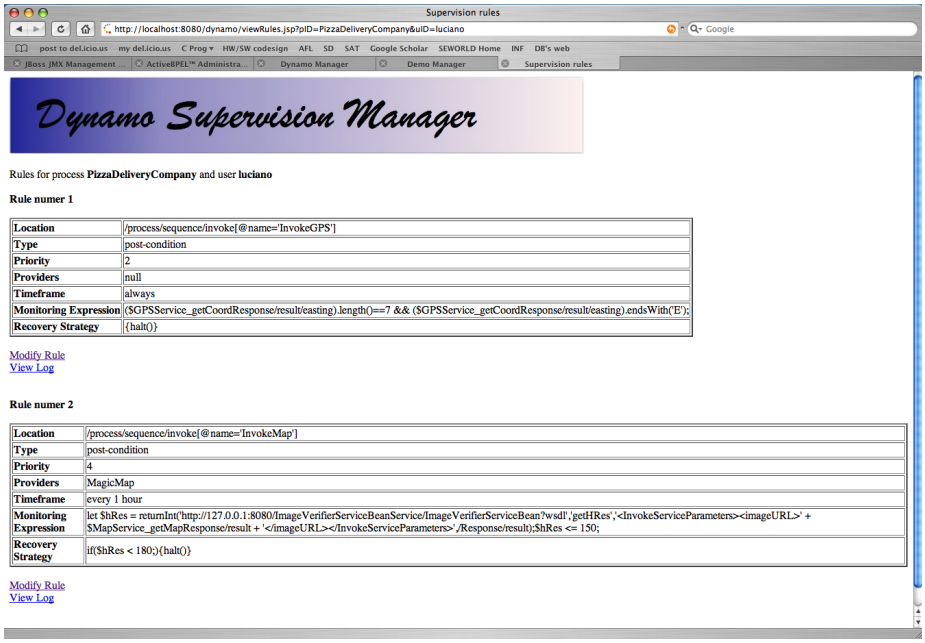

Figure 4: Monitoring rules modified with the “Dynamo Supervision Manager”