UBIWARE Deliverable D1.2:

Progress Statuses of the Industrial Cases

March, 2008

Date	March 7, 2008
Document type	Report
Dissemination Level	UBIWARE project consortium
Contact Author	Vagan Terziyan
Co-Authors	Artem Katasonov, Oleksiy Loboda, Sergiy Nikitin
Work component	WP7
Deliverable Code	D1.2
Deliverable Owner	IOG, JYU
Deliverable Status	Mandatory, Internal
Intellectual Property Rights	Unaffected

Table of Contents

UBIWARE Deliverable D1.2:
Workpackage WP7:

Introduction

The objective of this workpackage is to trial UBIWARE on real industrial cases. This has two major goals for such case studies. The first goal is to evaluate the scientific concepts behind UBIWARE and to find problems and issues in UBIWARE that would otherwise be overlooked. The second goal is to facilitate the further utilization of UBIWARE in the industry. Several specific cases, proposed by the industrial partners, are analyzed, designed and prototyped based on the UBIWARE platform. The reasons for prototyping are the same: to identify issues in UBIWARE that would get overlooked if the work was only theoretical and thus abstract, and to demonstrate the benefits of UBIWARE in a tangible way so to facilitate future industrial adoption.

There are three industrial cases, those of ABB, Fingrid and Metso Automation.

During the Year 1, with respect to all three cases the following tasks have been performed:

Task T1.1_w7: Case analysis: identification of relevant industrial resources, their dependencies and interactions

Task T1.2_w7: Connecting to relevant industrial resources: Development of appropriate resource adapters

UBIWARE Deliverable D1.2:
Workpackage WP7:

1 Fingrid case

1.1 Background

During case analysis phase the following systems were considered:

- **Event History Database** (Oracle) in the office environment, to which data is automatically replicated from SCADA's event history database.
  - The focus is on **R1**-alarms, i.e. equipment alarms that require some maintenance actions to be performed.
  - The database also keeps events on when a maintenance worker entered or left a substation (billing is based on working time).
  - R3, R4, and R5 alarms, i.e. disturbances in the network, can also be considered.
  - Major R3, R4, and R5 alarms are also manually fed into Fingrid extranet (sähkömarkinnat => käyttöhäiriöt page)
- **Elnet system** (Oracle) that stores information about assets: towers, feeders, substations (i.e. the whole power network) + which maintenance service provider serves which working area.
  - **MapInfo** GIS is used for geographic representation of the power network.
- **Tosu system** (MS Access) that is used by the maintenance service providers to report to Fingrid the costs for the work performed.
- **The lightning info-service** – FMI provides for Fingrid data on all the lightning events in Finland. Additionally, FMI provides a Google Maps –based application, which geographically shows lightnings data combined with the data on the locations of Fingrid power lines.

1.2 Special requirements

Data security is a central concern. It is major reason, along with **safety**, for putting the focus on historic analysis of events rather on the real-time operation. It is also the reason for that no real-time systems will be accessible for UBIWARE development for a while.

1.3 Fingrid industrial prototype

The goal of industrial prototype for the first year is to implement Statistic analysis of the Event History data using UBIWARE platform integrated with two industrial resources: Event History DB and a Human. Pilot application delivers reports on how many R1 alarms were happening per month /year per working area.

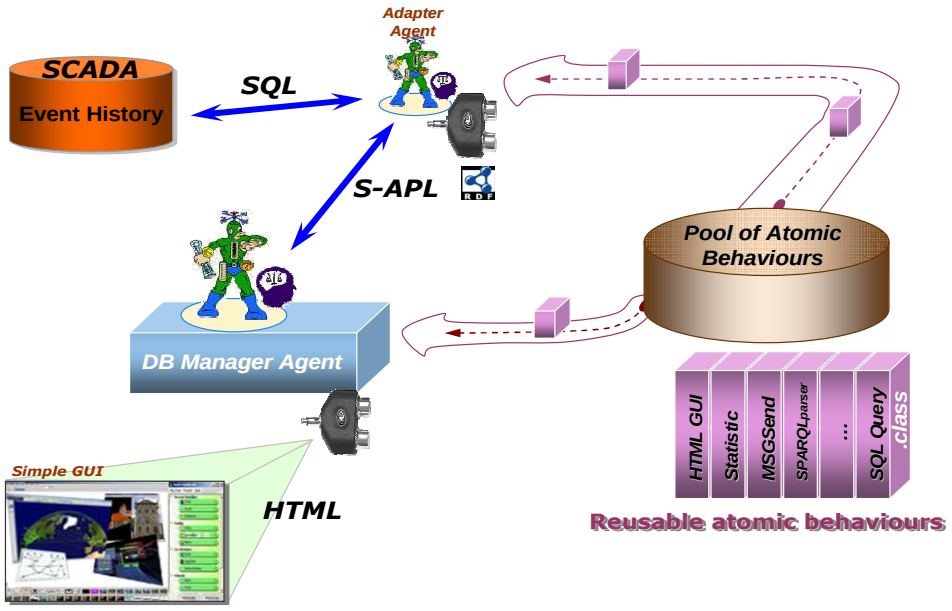


Figure 1 - Industrial prototype – Fingrid case

During development of Fingrid industrial prototype SQLReaderBehavior has been developed. It is included now to the platform and can be reused to any case-implementation that requires reading data from SQL database. SQLReaderBehavior is java implementation of JDBC database connector. Together with S-APL configuration script it represents one of the key points of UBIWARE platform – configurable adapter.

```
{gb:I gb:do java:ubiware.shared.SQLReaderBehavior} gb:configuredAs {
        x:driver gb:is "oracle.jdbc.OracleDriver" .
        x:url gb:is "jdbc:oracle:thin:@hades:1521:UBIWARE" .
        x:username gb:is "FG" .
        x: password gb:is "FG" .
        x: query gb:is "SELECT UTCTIME,ALARMTEXT,PRIORITY,STATIONNAME
                                                FROM EVENTLOG WHERE PRIORITY=1".
       gb:Success gb:add {
          {gb:I gb:do :Print} gb:configuredAs {x:print gb:is "Success."}
       }
  }
```

SQLReaderBehavior configuration script

Configuration script requires following parameters to be defined:

- driver – Java implementation of JDBG driver.
- url – connection string for source database
- username – username for database access
- password – password for database access
- query – SQL query for retrieving data

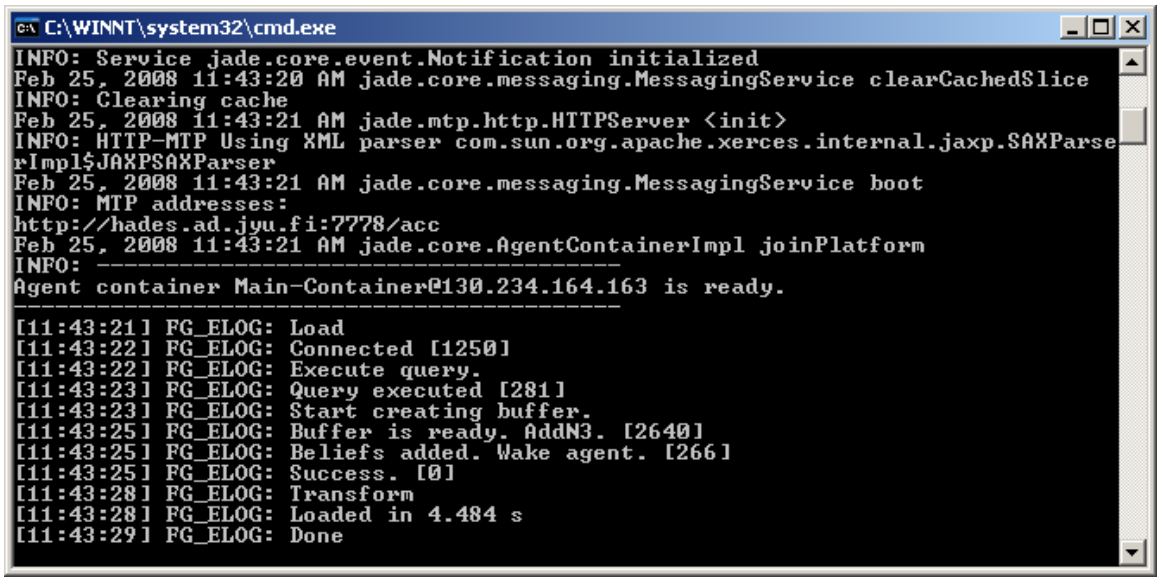


Figure 2 - SCADA Event log database adapter startup

During its startup adapter agent queries target database and stores retrieved data into memory in form of *{S P O}* triple.

```
"1" <#row> {
     "UTCTIME" <#column> "0007-10-09 13:23:00.0".
     "ALARMTEXT" <#column> "09.10.07 15:23:00.043  K       DCTSta  TDS RTCalc
Alarm TP       ...  RTC INVALID v=11: 11: 11: 11: 11: 11 s=IOSet_8011 f=rtc"
     }.
"2" <#row> {
     "UTCTIME" <#column> "0007-10-09 13:23:00.0".
     "ALARMTEXT" <#column> "09.10.07 15:23:00.044  K       DCTSta  TDS RTCalc
Alarm TP       ...  RTC INVALID v=11117: 11117: 11117: 11117: 1111 s=IOSet_"
     }.
```

After Event log data successfully read from database it needs to be transformed into format specified in Fingrid ontology (APPENDIX A). The following S-APL code does transformation:

```
  {
     EVENTLOG :table {?rowId :row {
                UTCTIME :column ?utc.
                ALARMTEXT :column ?text.
                PRIORITY :column ?p.
                STATIONNAME :column ?station}}.
          fg:Network fg:hasStationGroups {{
                     ?group fg:hasStation ?station} gb:is gb:Optional}
  }  => {
      gb:I gb:remove {EVENTLOG :table *}.
      {
       {gb:true = exists(?group)} -> {
           fg:EventHistory fg:hasEvents {{
              fg:alarm is ?text.
              fg:time is ?utc.
              fg:type is ?p.
              fg:substation is ?station.
              fg:group is ?group
} fg:id ?rowId}
}
  } gb:All ?rowId .
}.
```

After successful transformation Event log data looks like:

```
{
    <http://www.fingrig.fi/elog#alarm> "is" "02.10.07 14:51:58.256  K    R  RADC
RT222 H1 SUOD. LAUKAISU   ..E  Hälyttää".
    <http://www.fingrig.fi/elog#time> "is" "2007-10-02 11:51:58.0".
    <http://www.fingrig.fi/elog#type> "is" "1".
    <http://www.fingrig.fi/elog#substation> "is" "RADC".
    <http://www.fingrig.fi/elog#group> "is" "Raisio"
} <http://www.fingrig.fi/elog#id> "351".
{
    <http://www.fingrig.fi/elog#alarm> "is" "25.05.2007 16:00:42  K   ESDC
HARKU-LINKIN SÄÄTÖSEKV.   ...  Käyntiin annettu / MY4".
    <http://www.fingrig.fi/elog#time> "is" "2007-05-25 13:00:42.0".
    <http://www.fingrig.fi/elog#type> "is" "1".
    <http://www.fingrig.fi/elog#substation> "is" "ESDC".
    <http://www.fingrig.fi/elog#group> "is" "Vantaa"
} <http://www.fingrig.fi/elog#id> "5997".
```

DB Manager Agent provides simple HTML-based user interface for producing reports on R1 alarms.

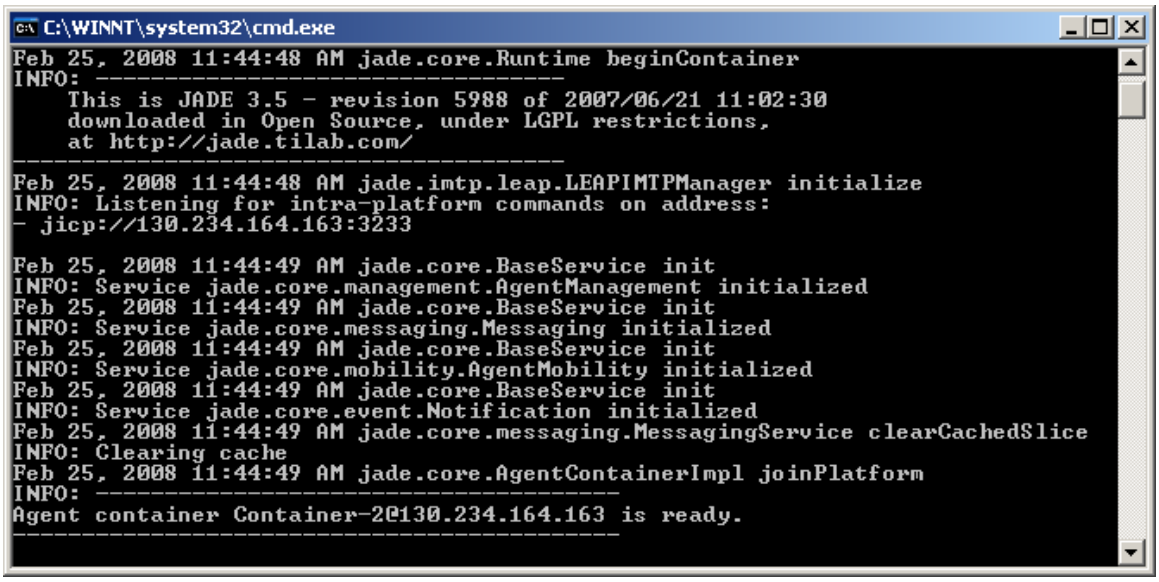
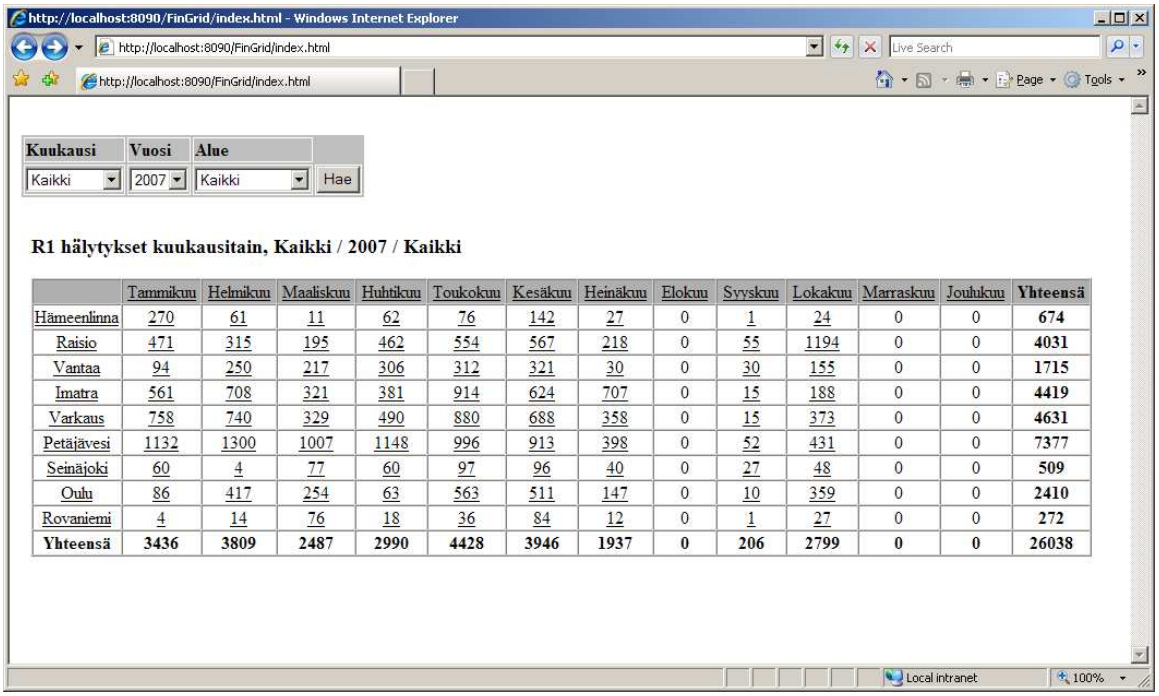


Figure 3 - DB Manager Agent startup

During DB Manager Agent startup java:ubiware.core.AgentServer is started. Agent also loads communication, statistics calculation and XML producing scripts. java:ubiware.shared.XmlWriterBehavior RAB is used for generating XHTML and java:ubiware.shared.HttpResponseSenderBehavior is utilized for sending response to web-browser.



	Tammikuu	Helmikuu	Maaliskuu	Huhtikuu	Toukokuu	Kesäkuu	Heinäkuu	Elokuu	Syyskuu	Lokakuu	Marraskuu	Joulukuu	Yhteensä
Hämeenlinna	270	61	11	62	76	142	27	0	1	24	0	0	**674**
Raisio	471	315	195	462	554	567	218	0	55	1194	0	0	**4031**
Vantaa	94	250	217	306	312	321	30	0	30	155	0	0	**1715**
Imatra	561	708	321	381	914	624	707	0	15	188	0	0	**4419**
Varkaus	758	740	329	490	880	688	358	0	15	373	0	0	**4631**
Petäjävesi	1132	1300	1007	1148	996	913	398	0	52	431	0	0	**7377**
Seinäjoki	60	4	77	60	97	96	40	0	27	48	0	0	**509**
Oulu	86	417	254	63	563	511	147	0	10	359	0	0	**2410**
Rovaniemi	4	14	76	18	36	84	12	0	1	27	0	0	**272**
Yhteensä	**3436**	**3809**	**2487**	**2990**	**4428**	**3946**	**1937**	**0**	**206**	**2799**	**0**	**0**	**26038**

Figure 4 - Main view – R1 equipment alarms by group of stations summary

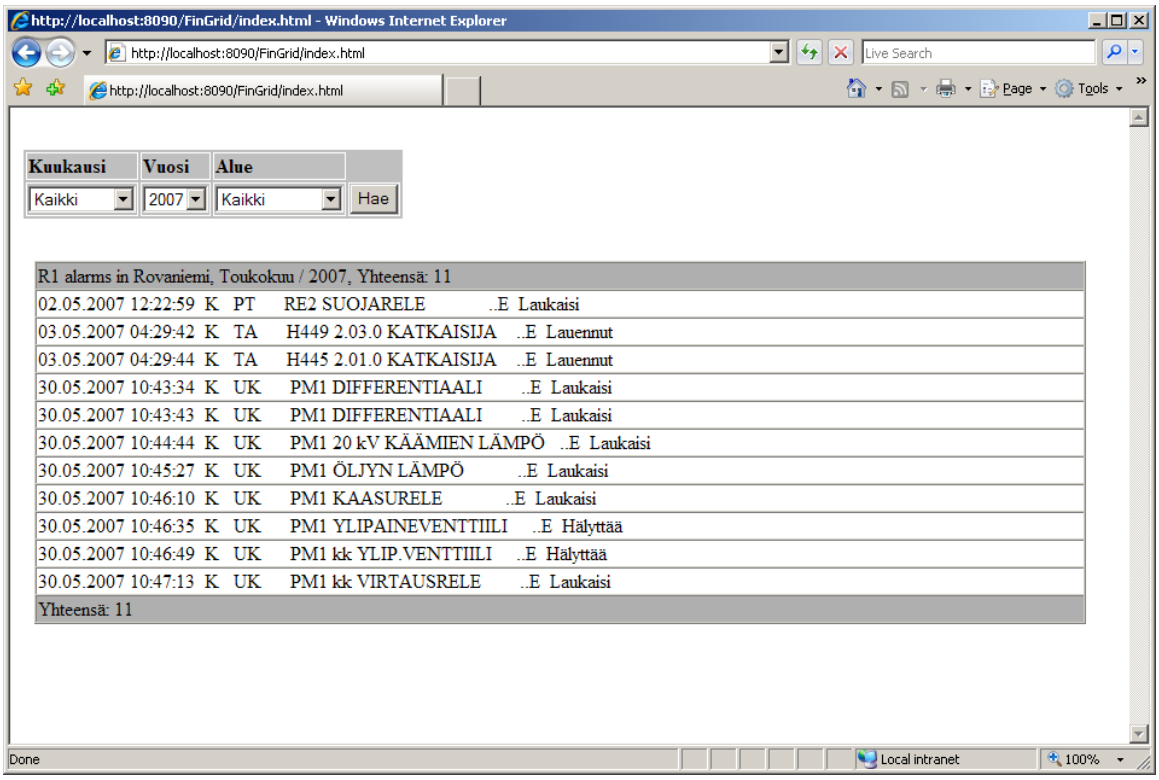
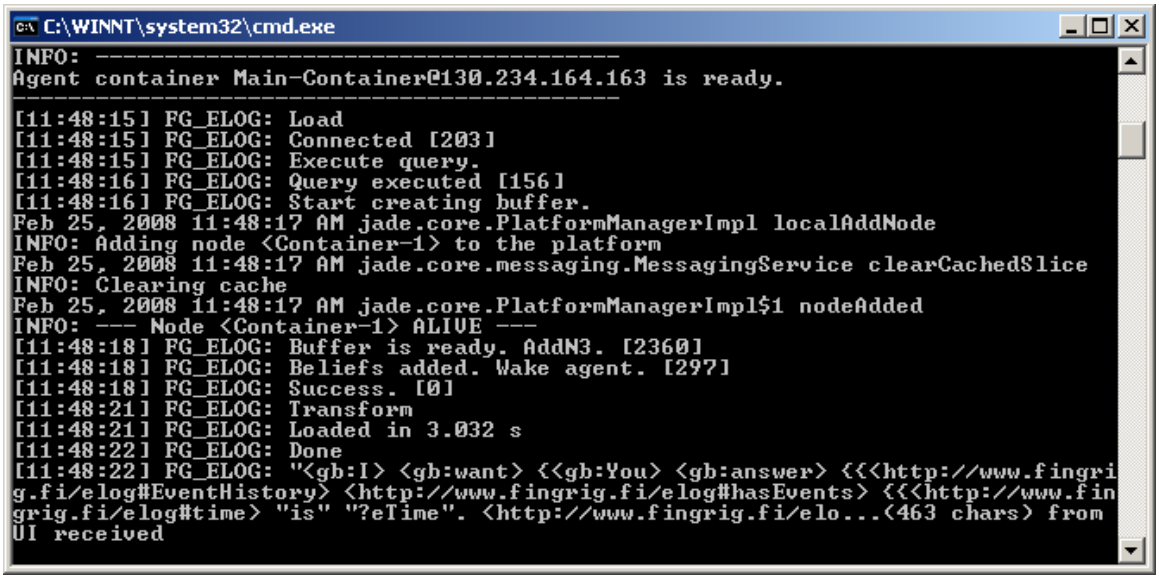


Figure 5 - R1 equipment alarms by group of stations per month

Every report requires communication between DB Manager and Adapter agents.



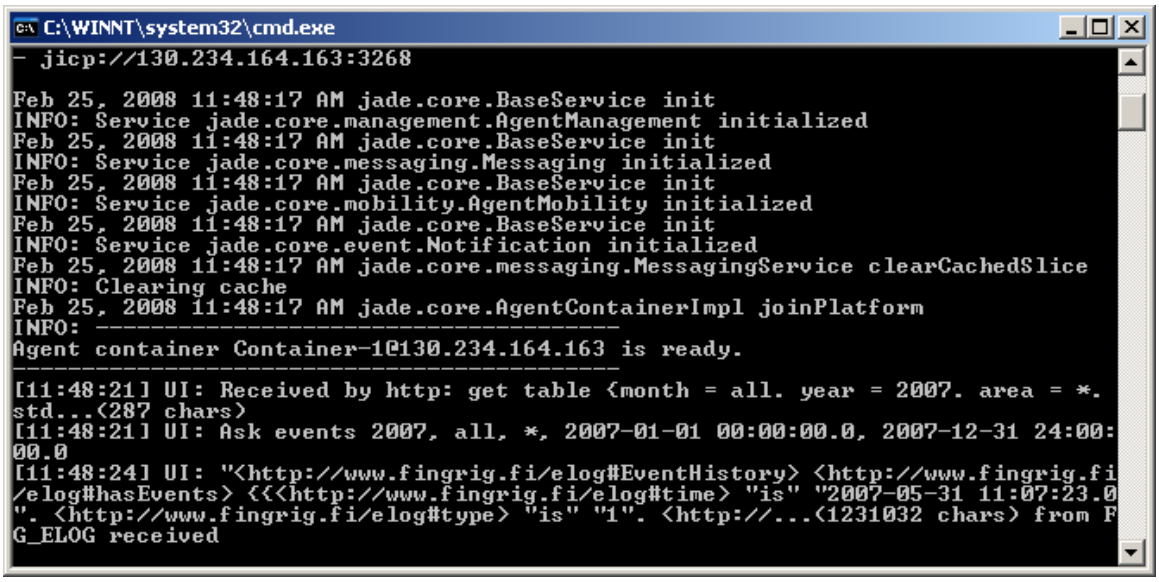


Figure 6 - Example of communication between agents

Sample of communication script looks like:

```
gb:I gb:want {gb:You gb:answer {
       {
              fg:EventHistory fg:hasEvents {{
                     fg:time is ?eTime.
                     fg:type is 1.
                     fg:group is ?group
              } fg:id ?rId}.
              ?group = ?area.
              ?eTime <= "?ed"^^xsd:string.
              ?eTime >= "?std"^^xsd:string
              } gb:All ?rId
       }
}
```

1.4 Future opportunities

According to case studies the following functions could potentially be automated:

- **Extension of Statistic analysis of the Event History data**
  - Analysis of efficiency of maintenance service providers. In case of an R1 alarm in their working area, the provider is notified automatically. One question is how much time it takes the provider to reach the substation to perform maintenance.
  - Filtering out (as an option) events while “kuluvalvonta” is off

- **Integrating data from Event History, Elnet and Tosu**
  - Analysis of the relationship between alarms (Event history) and the types of equipment (Elnet).
  - Understanding what alarm has led to what maintenance actions at what cost.

- **Integrating data from Event History and Lightening data**
  - Matching the locations and times of R3,4,5 alarms with the locations and times of lightning strikes to automatically filter out lightning-caused disturbances (normally require no action to be performed).

UBIWARE Deliverable D1.2:
Workpackage WP7:

2 ABB case

2.1 Background

During case analysis phase the following systems were considered:

- **Event History Table** (MicroSCADA). Event log that contains records about events on that happened on feeders, substation and control centre.
- **Weather station of KOILLIS-SATAKUNNAN SÄHKÖ OY(KSS).** Website (http://62.197.182.85/saa/index.html) contains history of weather parameters for last 8 days.

2.2 Special requirements

IEC 61850 Standard support

2.3 ABB industrial prototype

The goal of ABB industrial prototype for the first year is to provide Analysis of the Event History data along with simple classification for events based on descriptions provided by experts (Events annotation). This includes:

- Group events by feeder, substation.
- Analysis of automatic reclosing sequences and providing assistance to operator for event classification.

Prototype application consists of 2 agents: Event sender Agent and Control Agent. Event sender simulates appearing event in real time and pushes events to Control Agent. Control Agent receives events, does reasoning and produces HTML GUI.

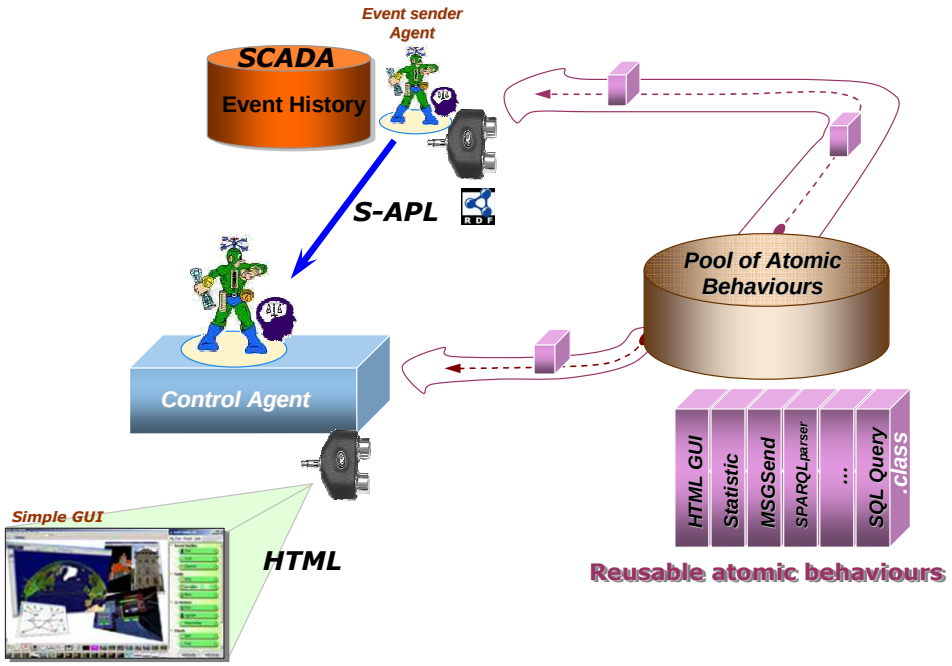
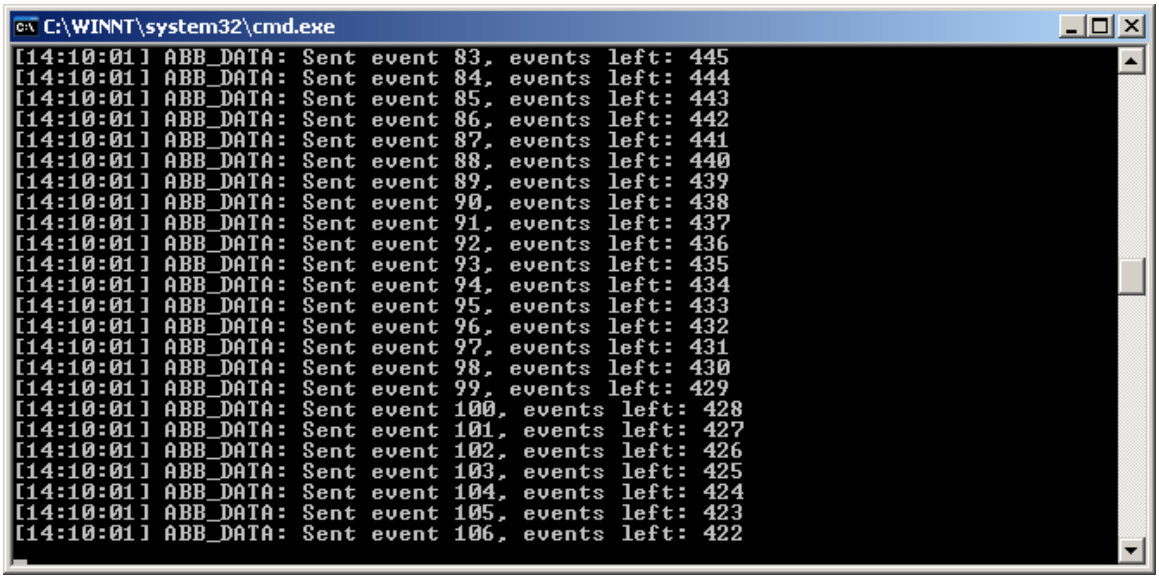


Figure 7 - Industrial prototype – ABB case



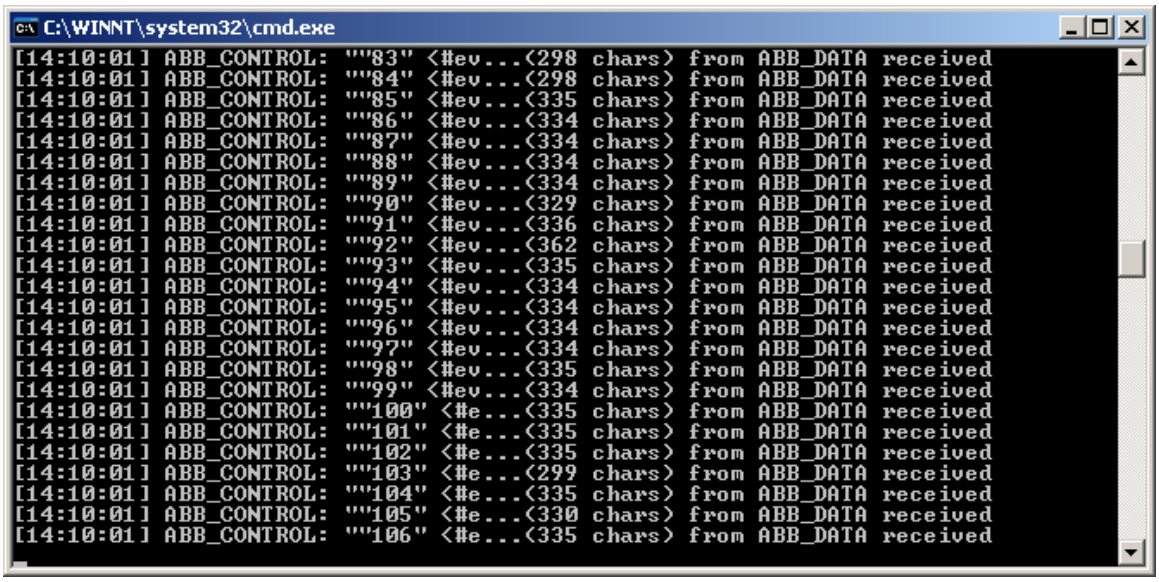


Figure 8 - Communication between agents

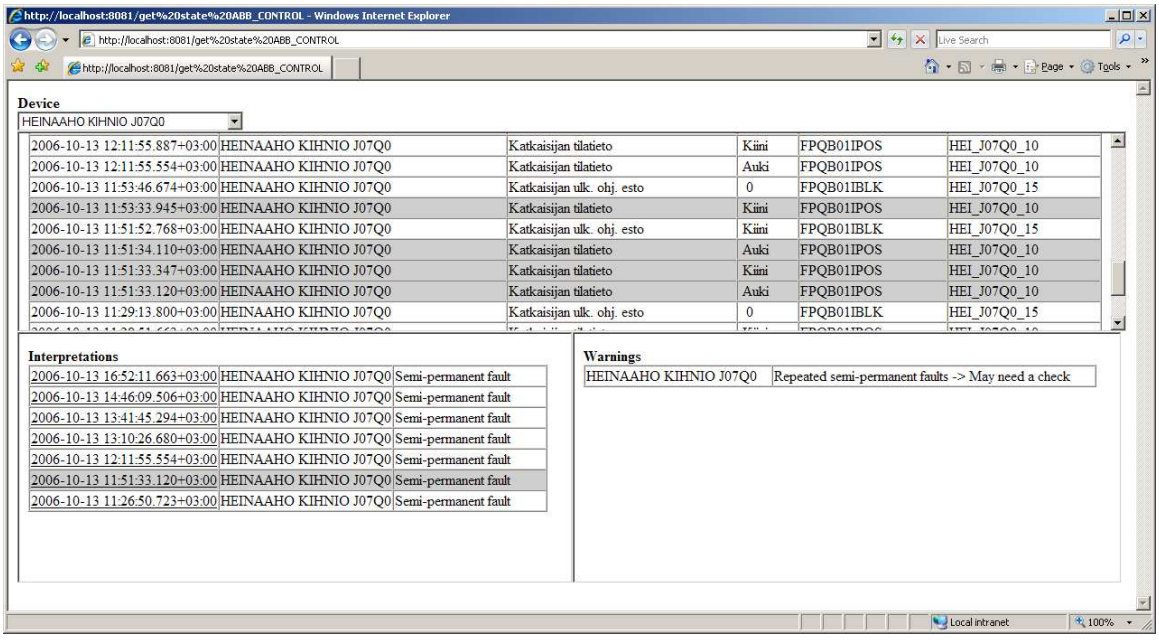


Figure 9 - Main view – Control center panel

2.4 Future opportunities

- **Integrating weather data.**
  - Analysis of the relationship between alarms in event history and the weather conditions.
- **Integrating feeders' location data** (feeders' coordinates)
  - Possibility to display feeders' locations on map.
- **Integrating information about landscape.**
  - Analysis of dependence different faults on landscape.

UBIWARE Deliverable D1.2:
Workpackage WP7:

3 Metso case

3.1 Background

Metso Automation has been running research activities with Industrial Ontologies Group related to the Semantic Integration of industrial information for four years already. For the last one and a half years the cooperation strengthened and a separate privately funded project was launched. The project called SWIMMER aims at exploring the possibilities and add-values the Semantic Technology can bring to enhance future products of Metso Automation. The Ubiware project has a specific case for Metso Automation, which complements the activities in the private project. Within the UBIWARE project Metso Automation expects to grasp UBIWARE platform functionality and capabilities. Special attention is paid to the adaptation of new sources and semantic querying of the industrial data. Lately there was an express of interest related to filtering and classification of events.

3.2 Metso Prototype

The prototype demonstrates the infrastructure for web service-based information flows to the UBIWARE platform, the adaptation of the received information and simple browsing of the collected data with primitive web-based interface, which issues queries to the platform via HTTP.
Metso Automation extensively uses Web Service technology within the company that allows simple and fast creation of data flows to and from different sources. Metso Automation has arranged a flow of SOAP/XML messages to the University server. These messages may contain information from different customers and different information systems. Metso sends us only those, which they consider as interesting ones for information integration and future querying. The schema below (see Figure 10) describes the setup for storage of the information from Metso.

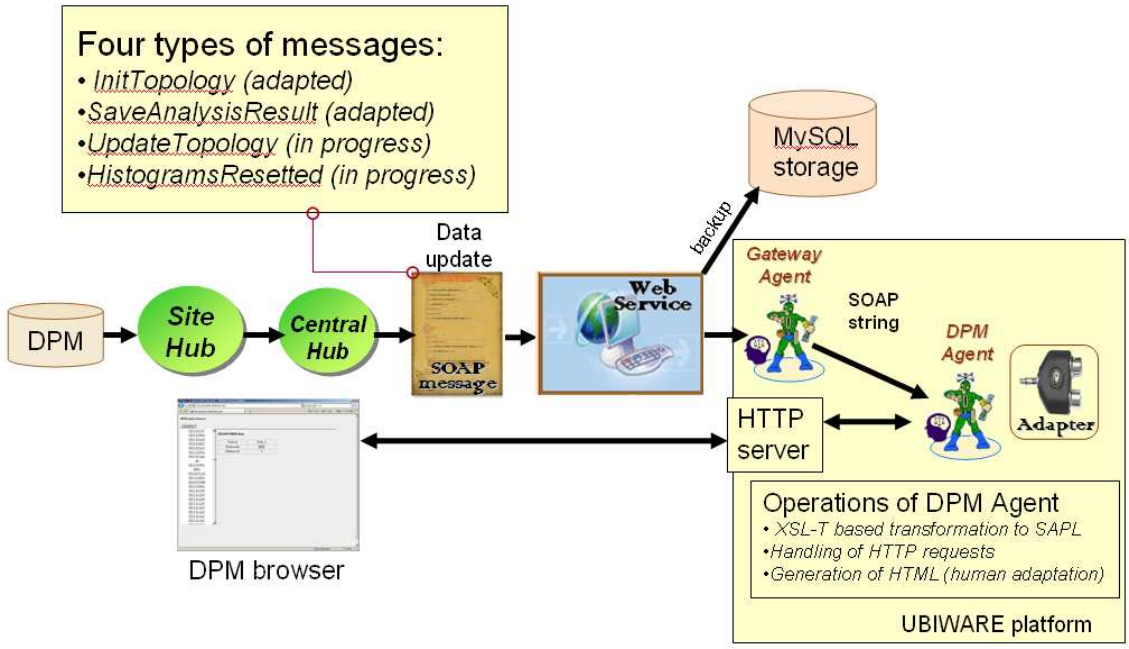


Figure 10 - Infrastructure for information flow from Metso

As you can see from the picture, the messages are passed to the UBIWARE platform via UbiwareGatewayAgent – an external agent that is registered to the platform by the standalone web service and implements an abstract class of JADE - `jade.wrapper.gateway.GatewayAgent`.

The communication of the web service and the GatewayAgent is conducted via `WSBlackBoardBean` class – a class that encapsulates the getter and setter methods for setting up a serialized SOAP message and name of the recipient on the platform. Upon the arrival of the message, the web service constructs the `WSBlackBoardBean` and passes it to the GatewayAgent, that is registered to the JADE platform via static method `JadeGateway.init()` (see code listing below).

```
public String sendMessageToAgentPlatform(String msgid, String msg,
      String agentname){
      Properties p= new Properties();
      p.put(jade.core.ProfileImpl.MAIN_PORT, JADE_PORT);
      p.put(jade.core.ProfileImpl.MAIN_HOST, JADE_HOST);
           JadeGateway.init("ubiware.metso.gateway.UbiwareGatewayAgent",p);
      WSBlackBoardBean wsbbb=new WSBlackBoardBean();
      wsbbb.setReceiver(agentname);
      wsbbb.setMessage(msg);
      wsbbb.setMessageID(msgid);
      try {
            System.out.println("Service is trying to send message to the
            agent platform via gateway...");
            JadeGateway.execute(wsbbb);
            String answ=wsbbb.getMessage();
```

```
            System.out.println("Returned to the service in a blackboard:
            "+answ);
            return answ;
      } catch (Exception e) {e.printStackTrace();}
      return null;
}
```

The interaction of the service and the platform is synchronous, i.e. the service will wait for the answer from the platform infinitely and therefore, the logic of the agent should be carefully written to ensure that the answer will be given.

The `UbiwareGatewayAgent` creates an ACL-message from the content of the bean object and sends it to the recipient. The recipient is Ubiware Agent. This is the point of linkage of the UBIWARE platform and the external web service (see code listing below).

```
public class UbiwareGatewayAgent extends GatewayAgent {
      WSBlackBoardBean board = null;
      protected void processCommand(java.lang.Object obj) {
            if (obj instanceof WSBlackBoardBean) {
                  board = (WSBlackBoardBean)obj;
                  ACLMessage msg = new ACLMessage(ACLMessage.REQUEST);
                  msg.addReceiver(new AID( board.getReceiver(),
                                    AID.ISLOCALNAME) );
                  msg.setContent(board.getMessage());
                  msg.setOntology("SAPL");
                  msg.setConversationId("DPMMsg_"+board.getMessageID());
                  send(msg);
            }
            addBehaviour(new CyclicBehaviour(this){
                  public void action() {
                        ACLMessage msg = receive();
                        if ((msg!=null)&&(board!=null)) {
                              board.setMessage(msg.getContent());
                              releaseCommand(board);
                        } else block();
                  }
            });
      }
}
```

The `processComand()` method launches the cyclic behavior, that waits for the reply from the targeted `UbiwareAgent`. As soon as reply message arrives, the agent calls the `releaseCommand()` method to tell the web service, that the processing of the bean object is completed and now it contains the answer from the agent.

The processing logic of any UbiwareAgent is encoded in S-APL script. The script contains a set of rules that constitute the execution plan. In this sense S-APL and Ubiware platform can be used as a workflow engine, where the invocations of components are done via Reusable Atomic Behavior invocations. The sequence diagram below (see Figure 11) describes the logic of the `DPMAgent` – an agent, that serves as a handler of messages from the DPM database.

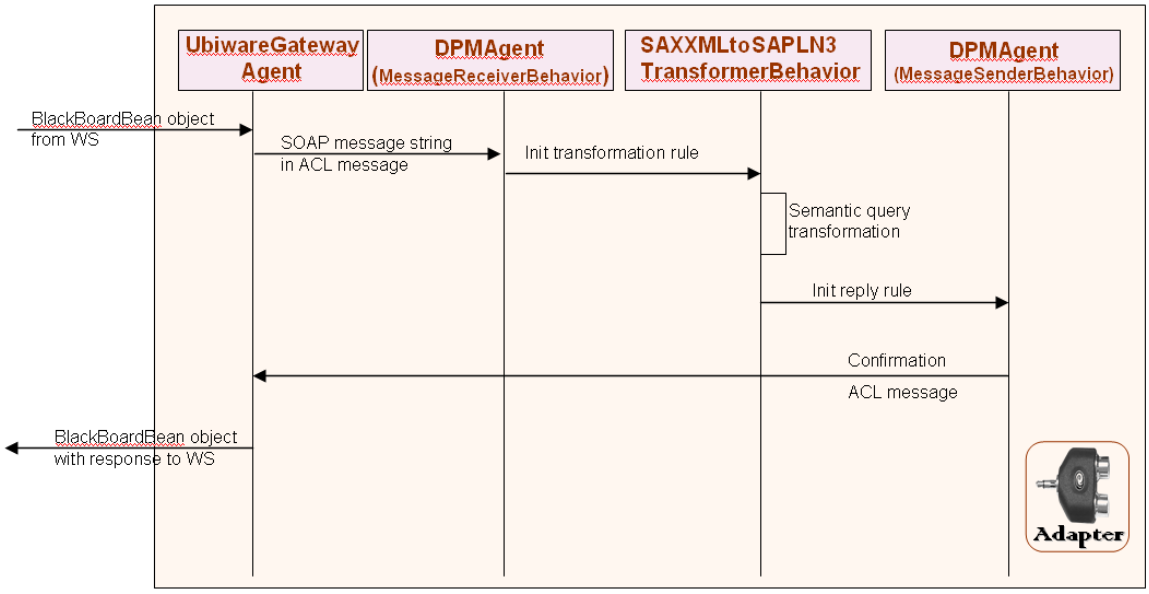


Figure 11 - Adapter sequence diagram

As you can see from the diagram above, the agent runs the `MessageReceiverBehavior` – a reusable atomic behavior for the processing of ACL messages. The behavior processes the incoming message and adds a set of triples to the beliefs of the agent that formulate a precondition for a message processing rule. The rule extracts the contents of the message and the sender. Then it deletes all the triples created by the `MessageReceiverBehavior` and constructs a new set of triples as a precondition for further processing by other rules. The processing rule can be considered as a dispatcher for the incoming messages.

```
/*Receive messages with SAPL ontology*/
{gb:I gb:do java:ubiware.shared.MessageReceiverBehavior}
      gb:configuredAs
            {x:waitOnlyFirst gb:is false. x:contentIsSAPL gb:is false}.

//Process the contents of the received message
{
   {?requestID x:received {x:conversationID gb:is ?convID.
            x:content gb:is ?content. x:sender gb:is ?sender}
   }=>{
            gb:I gb:remove {?requestID x:received *}.
            gb:I gb:add {?convID :newMessage ?content.
                         ?convID :msgSender ?sender.
                         ?convID :isProcessed false}
      }.
} gb:is gb:Rule.
```

The next rule continues the execution flow by starting the transformation behavior:

```
//Transform contents to SAPL using XSLT
{
  {?convID :newMessage ?content. ?convID :msgSender ?sender.
   ?convID :isProcessed false}=>
```

```
  {
   {gb:I gb:do java:ubiware.metso.SAXXMLtoSAPLN3TransformerBehavior}
     gb:configuredAs{
            x:xsltfile gb:is xslt/init_topology.xslt.
            x:xmlstring gb:is ?content.
            gb:Success gb:add {?convID :isTransformed true}
     }.
    gb:I gb:remove {?convid :isProcessed *}
  }
} gb:is gb:Rule.
```

The rule follows the same pattern: It does some action and adds new triple that is a precondition for the next rule, and then it removes the triples for its own precondition in order not to be executed again.
The next rule sends a reply message with the confirmation to the `GatewayAgent`:

```
//Reply to the sender
{
 {?convid :isTransformed true. ?convid :msgSender ?sender}
  =>{
      {gb:I gb:do java:ubiware.shared.MessageSenderBehavior}
            gb:configuredAs {
                  x:receiver gb:is ?sender .
                  x:content gb:is {Message stored Successfully} .
                  x:ontology gb:is "SAPL" .
                  x:conversationId gb:is ?convid
            }.
      gb:I gb:remove {?convid :isTransformed true}
 }
} gb:is gb:Rule
```

The `DPMAgent` also provides functionality for browsing through the contents of the collected information. The UBIWARE platform provides simple http server functionality. Any agent can start an HTTP server and listen to the incoming requests. The requests are then processed by the handling rules. Below is an excerpt from the S-APL script of the DPMAgent.

```
{gb:I gb:do java:ubiware.core.AgentServer}
            gb:configuredAs {x:port gb:is 8090}.
// --- Rule for loading the default page ---
{
 {{? ? index.htm} gb:eventFrom ?socket. ?socket != GUI}
 >>{
    {gb:I gb:do java:ubiware.shared.HttpResponseSenderBehavior}
      gb:configuredAs {
            x:messageFromFile gb:is ubiware/metso/web/dpmindex.htm.
            x:socket gb:is ?socket
      }
 }
} gb:is gb:Rule.
```

The first statement says, that the agent starts an http server and the next rule checks the precondition against the requested page (index.htm) and returns the default page (dpmindex.htm) (see Figure 12).

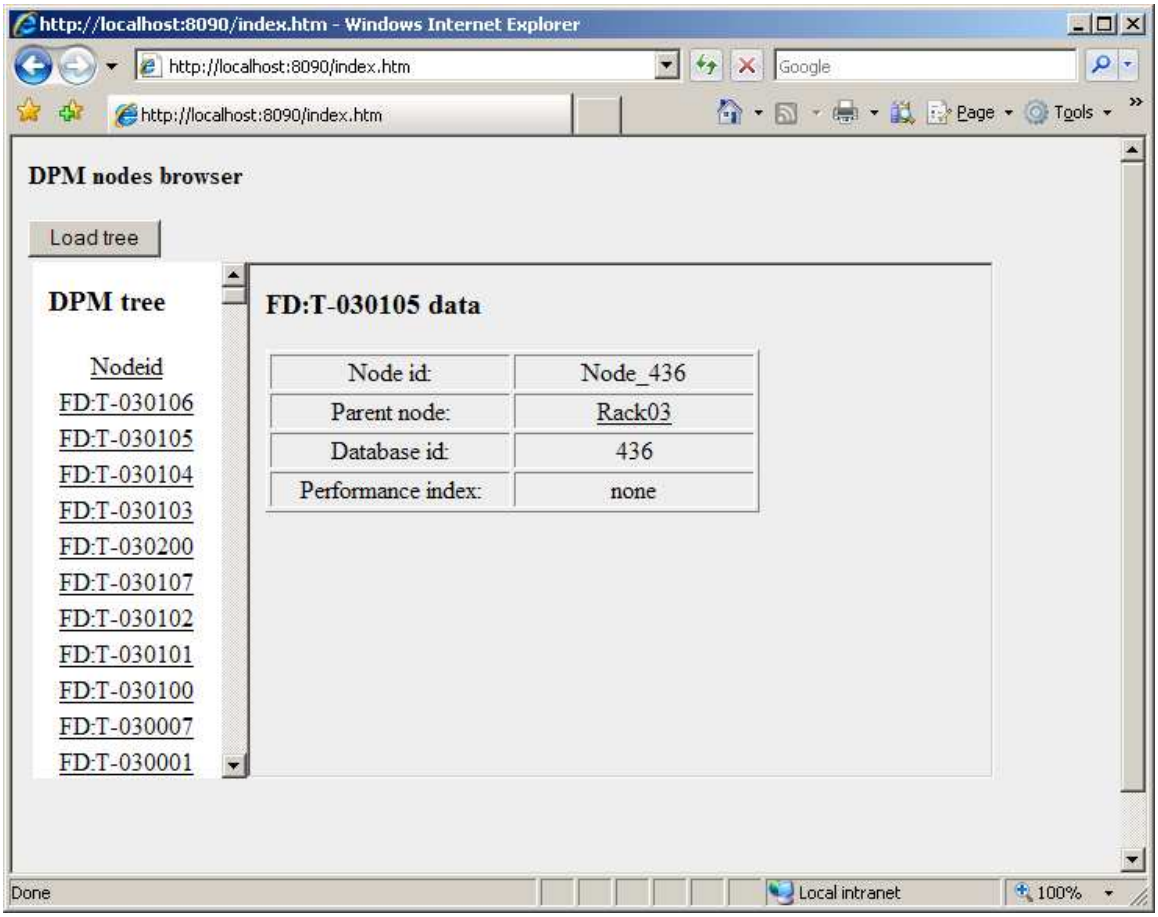


Figure 12 - Browsing the data

The querying of the data is implemented as S-APL rules in a SPARQL-like manner. When a user clicks on a node in the left pane, the browser sends an HTTP request to the server to extract information about the requested node and displays it in the right pane. The rule that handles this request has the following form:

```
{
  {{get :nodeHistory ?node} gb:eventFrom ?socket. ?socket != GUI.
    ?node :hasName ?nname.
    ?node :partOf ?parentid.
    ?parentid :hasName ?parname.
    ?node :dbId ?dbid.
    {?devanaresid :nodeId ?node.
    ?devanaresid :perfIndex ?perfind} gb:is gb:Optional
  } >>
{“Generate HTML here”...}
} gb:is gb:Rule.
```

The rule defines a pattern form triples and variables, so that agent's beliefs are matched against it filling the variables with values.

3.3 Future opportunities

Amongst future opportunities considered by Metso Automation and planned as a cooperative future work with Industrial Ontologies Group is strengthening of the joint SW project involving experts from both sides to incorporate the semantic features into new Metso Automation's products.

UBIWARE Deliverable D1.2:
Workpackage WP7:

APPENDIX A - *Fingrid ontology*

```
<?xml version="1.0"?>
<rdf:RDF
    xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema#"
    xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
    xmlns:owl="http://www.w3.org/2002/07/owl#"
    xmlns="http://www.cs.jyu.fi/ai/fg.owl#"
  xml:base="http://www.cs.jyu.fi/ai/fg.owl">
  <owl:Ontology rdf:about=""/>
  <owl:Class rdf:ID="Networks"/>
  <owl:Class rdf:ID="EventHistory"/>
  <owl:Class rdf:ID="Station"/>
  <owl:Class rdf:ID="Event"/>
  <owl:Class rdf:ID="StationGroup"/>
  <owl:ObjectProperty rdf:ID="hasStationGroups">
    <rdfs:range rdf:resource="#StationGroup"/>
    <rdfs:domain rdf:resource="#Networks"/>
  </owl:ObjectProperty>
  <owl:ObjectProperty rdf:ID="hasStation">
    <rdfs:domain rdf:resource="#StationGroup"/>
    <rdfs:range rdf:resource="#Station"/>
  </owl:ObjectProperty>
  <owl:ObjectProperty rdf:ID="hasEvents">
    <rdfs:domain rdf:resource="#EventHistory"/>
    <rdfs:range rdf:resource="#Event"/>
  </owl:ObjectProperty>
  <owl:FunctionalProperty rdf:ID="substation">
    <rdf:type
rdf:resource="http://www.w3.org/2002/07/owl#ObjectProperty"/>
    <rdfs:range rdf:resource="#Station"/>
    <rdfs:domain rdf:resource="#Event"/>
  </owl:FunctionalProperty>
  <owl:FunctionalProperty rdf:ID="type">
    <rdfs:domain rdf:resource="#Event"/>
    <rdf:type
rdf:resource="http://www.w3.org/2002/07/owl#DatatypeProperty"/>
    <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#int"/>
  </owl:FunctionalProperty>
```

```
    <owl:FunctionalProperty rdf:ID="alarm">
      <rdf:type
rdf:resource="http://www.w3.org/2002/07/owl#DatatypeProperty"/>
      <rdfs:domain rdf:resource="#Event"/>
      <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#string"/>
    </owl:FunctionalProperty>
    <owl:FunctionalProperty rdf:ID="time">
      <rdfs:range
rdf:resource="http://www.w3.org/2001/XMLSchema#dateTime"/>
      <rdf:type
rdf:resource="http://www.w3.org/2002/07/owl#DatatypeProperty"/>
      <rdfs:domain rdf:resource="#Event"/>
    </owl:FunctionalProperty>
    <owl:FunctionalProperty rdf:ID="group">
      <rdf:type
rdf:resource="http://www.w3.org/2002/07/owl#ObjectProperty"/>
      <rdfs:range rdf:resource="#StationGroup"/>
      <rdfs:domain rdf:resource="#Event"/>
    </owl:FunctionalProperty>
    <Station rdf:ID="PEL"/>
    <Station rdf:ID="TO"/>
    <Station rdf:ID="KS"/>
    <Station rdf:ID="KOI"/>
    <Station rdf:ID="ESDC"/>
    <StationGroup rdf:ID="Rovaniemi">
      <hasStation>
        <Station rdf:ID="OS"/>
      </hasStation>
      <hasStation>
        <Station rdf:ID="VAJ"/>
      </hasStation>
      <hasStation>
        <Station rdf:ID="IV"/>
      </hasStation>
      <hasStation>
        <Station rdf:ID="PT"/>
      </hasStation>
      <hasStation>
        <Station rdf:ID="KOS"/>
      </hasStation>
      <hasStation>
        <Station rdf:ID="TA"/>
      </hasStation>
      <hasStation>
        <Station rdf:ID="UK"/>
      </hasStation>
      <hasStation>
        <Station rdf:ID="MLT"/>
      </hasStation>
      <hasStation>
        <Station rdf:ID="VL"/>
      </hasStation>
      <hasStation>
        <Station rdf:ID="PI"/>
      </hasStation>
    </StationGroup>
```

```
<Station rdf:ID="AJ"/>
<StationGroup rdf:ID="Petäjävesi">
  <hasStation>
    <Station rdf:ID="MÄN"/>
  </hasStation>
  <hasStation>
    <Station rdf:ID="VH"/>
  </hasStation>
  <hasStation rdf:resource="#AJ"/>
  <hasStation>
    <Station rdf:ID="RAI"/>
  </hasStation>
  <hasStation rdf:resource="#KOI"/>
  <hasStation>
    <Station rdf:ID="KGV"/>
  </hasStation>
  <hasStation>
    <Station rdf:ID="JÄ"/>
  </hasStation>
  <hasStation>
    <Station rdf:ID="KJO"/>
  </hasStation>
  <hasStation>
    <Station rdf:ID="PE"/>
  </hasStation>
  <hasStation rdf:resource="#TO"/>
</StationGroup>
<Station rdf:ID="LOKT"/>
<Station rdf:ID="KI"/>
<Station rdf:ID="KA"/>
<Station rdf:ID="MEL"/>
<Station rdf:ID="SUM"/>
<Station rdf:ID="HY"/>
<Station rdf:ID="VJ"/>
<Station rdf:ID="ST"/>
<Station rdf:ID="KLT"/>
<Station rdf:ID="MYY"/>
<Station rdf:ID="RUO"/>
<Station rdf:ID="PUH"/>
<Station rdf:ID="HA"/>
<Station rdf:ID="TM"/>
<StationGroup rdf:ID="Hämeenlinna">
  <hasStation>
    <Station rdf:ID="AHE"/>
  </hasStation>
  <hasStation>
    <Station rdf:ID="HEK"/>
  </hasStation>
  <hasStation>
    <Station rdf:ID="VNJKT"/>
  </hasStation>
  <hasStation>
    <Station rdf:ID="HML"/>
  </hasStation>
  <hasStation>
    <Station rdf:ID="KML"/>
```

```
    </hasStation>
    <hasStation>
      <Station rdf:ID="HI"/>
    </hasStation>
    <hasStation>
      <Station rdf:ID="PRN"/>
    </hasStation>
    <hasStation>
      <Station rdf:ID="TKM"/>
    </hasStation>
    <hasStation>
      <Station rdf:ID="FO"/>
    </hasStation>
    <hasStation rdf:resource="#KA"/>
    <hasStation rdf:resource="#MEL"/>
    <hasStation>
      <Station rdf:ID="KR"/>
    </hasStation>
    <hasStation>
      <Station rdf:ID="NIK"/>
    </hasStation>
    <hasStation>
      <Station rdf:ID="KYM"/>
    </hasStation>
    <hasStation rdf:resource="#LOKT"/>
    <hasStation>
      <Station rdf:ID="VNJ"/>
    </hasStation>
    <hasStation>
      <Station rdf:ID="LO"/>
    </hasStation>
    <hasStation>
      <Station rdf:ID="KM"/>
    </hasStation>
    <hasStation rdf:resource="#HY"/>
  </StationGroup>
  <Station rdf:ID="TAE"/>
  <Station rdf:ID="UL"/>
  <Station rdf:ID="OLKT"/>
  <Station rdf:ID="HU"/>
  <Station rdf:ID="AN"/>
  <Station rdf:ID="VB"/>
  <Station rdf:ID="TLKT"/>
  <Station rdf:ID="TAI"/>
  <Station rdf:ID="VIH"/>
  <Station rdf:ID="LSA"/>
  <Station rdf:ID="KZN"/>
  <Station rdf:ID="SAV"/>
  <Station rdf:ID="UMH"/>
  <Station rdf:ID="MRT"/>
  <Station rdf:ID="ILH"/>
  <Station rdf:ID="MAR"/>
  <Station rdf:ID="NIL"/>
  <Station rdf:ID="NJ"/>
  <Station rdf:ID="SJ"/>
  <Station rdf:ID="UT"/>
```

```
<Station rdf:ID="KPP"/>
<Station rdf:ID="YL"/>
<Station rdf:ID="HAR"/>
<Station rdf:ID="KRJ"/>
<Station rdf:ID="RAS"/>
<Station rdf:ID="VAS"/>
<Station rdf:ID="PAR"/>
<Station rdf:ID="SEI"/>
<Station rdf:ID="PAM"/>
<Station rdf:ID="PR"/>
<Station rdf:ID="HT"/>
<StationGroup rdf:ID="Imatra">
  <hasStation>
    <Station rdf:ID="KAU"/>
  </hasStation>
  <hasStation rdf:resource="#YL"/>
  <hasStation>
    <Station rdf:ID="IMA"/>
  </hasStation>
  <hasStation rdf:resource="#VB"/>
  <hasStation>
    <Station rdf:ID="PUL"/>
  </hasStation>
  <hasStation rdf:resource="#TAI"/>
  <hasStation>
    <Station rdf:ID="KSL"/>
  </hasStation>
  <hasStation rdf:resource="#SUM"/>
  <hasStation>
    <Station rdf:ID="KPE"/>
  </hasStation>
  <hasStation>
    <Station rdf:ID="LKK"/>
  </hasStation>
  <hasStation rdf:resource="#MRT"/>
</StationGroup>
<Station rdf:ID="SLO"/>
<Station rdf:ID="PRV"/>
<Station rdf:ID="NU"/>
<Station rdf:ID="LPV"/>
<Station rdf:ID="RA"/>
<Station rdf:ID="OL_B"/>
<Station rdf:ID="SE"/>
<Station rdf:ID="WSF"/>
<StationGroup rdf:ID="Seinäjoki">
  <hasStation rdf:resource="#KS"/>
  <hasStation>
    <Station rdf:ID="TU"/>
  </hasStation>
  <hasStation rdf:resource="#VAS"/>
  <hasStation rdf:resource="#SJ"/>
  <hasStation>
    <Station rdf:ID="ISJ"/>
  </hasStation>
  <hasStation>
    <Station rdf:ID="UN"/>
```

```
    </hasStation>
    <hasStation>
      <Station rdf:ID="NIV"/>
    </hasStation>
    <hasStation>
      <Station rdf:ID="VE"/>
    </hasStation>
    <hasStation rdf:resource="#KZN"/>
    <hasStation>
      <Station rdf:ID="SIE"/>
    </hasStation>
    <hasStation rdf:resource="#WSF"/>
    <hasStation>
      <Station rdf:ID="KJ"/>
    </hasStation>
  </StationGroup>
  <StationGroup rdf:ID="Oulu">
    <hasStation>
      <Station rdf:ID="LE"/>
    </hasStation>
    <hasStation>
      <Station rdf:ID="PY"/>
    </hasStation>
    <hasStation rdf:resource="#NU"/>
    <hasStation rdf:resource="#VIH"/>
    <hasStation>
      <Station rdf:ID="PS"/>
    </hasStation>
    <hasStation rdf:resource="#ST"/>
    <hasStation>
      <Station rdf:ID="LEP"/>
    </hasStation>
    <hasStation rdf:resource="#HA"/>
    <hasStation>
      <Station rdf:ID="RTR"/>
    </hasStation>
    <hasStation rdf:resource="#KI"/>
    <hasStation rdf:resource="#RAS"/>
    <hasStation rdf:resource="#VJ"/>
    <hasStation>
      <Station rdf:ID="NTS"/>
    </hasStation>
    <hasStation>
      <Station rdf:ID="ISH"/>
    </hasStation>
    <hasStation rdf:resource="#UT"/>
    <hasStation>
      <Station rdf:ID="TIH"/>
    </hasStation>
    <hasStation rdf:resource="#SE"/>
    <hasStation rdf:resource="#PR"/>
  </StationGroup>
  <Station rdf:ID="AP"/>
  <Station rdf:ID="OLT"/>
  <Station rdf:ID="NKR"/>
  <Station rdf:ID="ES"/>
```

```
<Station rdf:ID="IN"/>
<Station rdf:ID="MNH"/>
<Station rdf:ID="VT"/>
<Station rdf:ID="RADC"/>
<Station rdf:ID="HARKU"/>
<StationGroup rdf:ID="Vantaa">
  <hasStation>
    <Station rdf:ID="HKI"/>
  </hasStation>
  <hasStation rdf:resource="#HARKU"/>
  <hasStation rdf:resource="#ES"/>
  <hasStation>
    <Station rdf:ID="VRK"/>
  </hasStation>
  <hasStation>
    <Station rdf:ID="VAA"/>
  </hasStation>
  <hasStation rdf:resource="#KRJ"/>
  <hasStation rdf:resource="#MYY"/>
  <hasStation rdf:resource="#IN"/>
  <hasStation rdf:resource="#PRV"/>
  <hasStation rdf:resource="#NJ"/>
  <hasStation rdf:resource="#MAR"/>
  <hasStation rdf:resource="#RUO"/>
  <hasStation rdf:resource="#TM"/>
  <hasStation rdf:resource="#AN"/>
  <hasStation>
    <Station rdf:ID="LS"/>
  </hasStation>
  <hasStation rdf:resource="#LPV"/>
  <hasStation rdf:resource="#ESDC"/>
</StationGroup>
<Station rdf:ID="SRV"/>
<Station rdf:ID="KOL"/>
<Station rdf:ID="KRN"/>
<Station rdf:ID="LI"/>
<Station rdf:ID="HUKT"/>
<Station rdf:ID="KEM"/>
<StationGroup rdf:ID="Raisio">
  <hasStation rdf:resource="#KEM"/>
  <hasStation rdf:resource="#UL"/>
  <hasStation rdf:resource="#SLO"/>
  <hasStation>
    <Station rdf:ID="LEJ"/>
  </hasStation>
  <hasStation rdf:resource="#PAR"/>
  <hasStation rdf:resource="#SEI"/>
  <hasStation rdf:resource="#OL_B"/>
  <hasStation rdf:resource="#HAR"/>
  <hasStation>
    <Station rdf:ID="O-HA"/>
  </hasStation>
  <hasStation>
    <Station rdf:ID="NAK"/>
  </hasStation>
  <hasStation>
```

```
      <Station rdf:ID="OL_A"/>
    </hasStation>
    <hasStation rdf:resource="#OLT"/>
    <hasStation rdf:resource="#OLKT"/>
    <hasStation rdf:resource="#TLKT"/>
    <hasStation rdf:resource="#TAE"/>
    <hasStation>
      <Station rdf:ID="KAL"/>
    </hasStation>
    <hasStation rdf:resource="#RADC"/>
    <hasStation>
      <Station rdf:ID="MP"/>
    </hasStation>
    <hasStation rdf:resource="#HT"/>
    <hasStation rdf:resource="#LI"/>
    <hasStation rdf:resource="#RA"/>
    <hasStation rdf:resource="#KOL"/>
  </StationGroup>
  <Station rdf:ID="KLH"/>
  <Station rdf:ID="SRK"/>
  <Station rdf:ID="VJR"/>
  <Networks rdf:ID="Network">
    <hasStationGroups rdf:resource="#Vantaa"/>
    <hasStationGroups>
      <StationGroup rdf:ID="Varkaus">
        <hasStation rdf:resource="#KPP"/>
        <hasStation rdf:resource="#NKR"/>
        <hasStation rdf:resource="#SRK"/>
        <hasStation rdf:resource="#KLT"/>
        <hasStation rdf:resource="#NIL"/>
        <hasStation rdf:resource="#LSA"/>
        <hasStation rdf:resource="#KLH"/>
        <hasStation>
          <Station rdf:ID="VAR"/>
        </hasStation>
        <hasStation rdf:resource="#PEL"/>
        <hasStation rdf:resource="#KRN"/>
        <hasStation rdf:resource="#SRV"/>
        <hasStation rdf:resource="#MNH"/>
        <hasStation rdf:resource="#SAV"/>
        <hasStation rdf:resource="#ILH"/>
        <hasStation>
          <Station rdf:ID="VART"/>
        </hasStation>
        <hasStation rdf:resource="#VJR"/>
        <hasStation>
          <Station rdf:ID="JUA"/>
        </hasStation>
        <hasStation rdf:resource="#AP"/>
        <hasStation rdf:resource="#PAM"/>
        <hasStation rdf:resource="#VT"/>
        <hasStation rdf:resource="#PUH"/>
        <hasStation rdf:resource="#HU"/>
        <hasStation rdf:resource="#UMH"/>
        <hasStation>
          <Station rdf:ID="JOE"/>
```

```
        </hasStation>
        <hasStation rdf:resource="#HUKT"/>
      </StationGroup>
    </hasStationGroups>
    <hasStationGroups rdf:resource="#Petäjävesi"/>
    <hasStationGroups rdf:resource="#Oulu"/>
    <hasStationGroups rdf:resource="#Rovaniemi"/>
    <hasStationGroups rdf:resource="#Seinäjoki"/>
    <hasStationGroups rdf:resource="#Raisio"/>
    <hasStationGroups rdf:resource="#Hämeenlinna"/>
    <hasStationGroups rdf:resource="#Imatra"/>
  </Networks>
</rdf:RDF>

<!-- Created with Protege (with OWL Plugin 3.3.1, Build 430)
http://protege.stanford.edu -->
```

UBIWARE Deliverable D1.2:
Workpackage WP7:

APPENDIX B - *ABB ontology*

```
<?xml version="1.0"?>
<rdf:RDF
    xmlns="http://www.cs.jyu.fi/ai/abb.owl#"
    xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema#"
    xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
    xmlns:owl="http://www.w3.org/2002/07/owl#"
  xml:base="http://www.cs.jyu.fi/ai/abb.owl">
  <owl:Ontology rdf:about=""/>
  <owl:Class rdf:ID="Fault"/>
  <owl:Class rdf:ID="EventHistory"/>
  <owl:Class rdf:ID="Event"/>
  <owl:ObjectProperty rdf:ID="causedByEvents">
    <owl:inverseOf>
      <owl:ObjectProperty rdf:ID="isSymptomOf"/>
    </owl:inverseOf>
    <rdfs:domain rdf:resource="#Fault"/>
    <rdfs:range rdf:resource="#Event"/>
  </owl:ObjectProperty>
  <owl:ObjectProperty rdf:about="#isSymptomOf">
    <rdfs:range rdf:resource="#Fault"/>
    <rdfs:domain rdf:resource="#Event"/>
    <owl:inverseOf rdf:resource="#causedByEvents"/>
  </owl:ObjectProperty>
  <owl:ObjectProperty rdf:ID="isEventOf">
    <owl:inverseOf>
      <owl:ObjectProperty rdf:ID="hasEvents"/>
    </owl:inverseOf>
    <rdfs:domain rdf:resource="#Event"/>
    <rdfs:range rdf:resource="#EventHistory"/>
  </owl:ObjectProperty>
  <owl:ObjectProperty rdf:about="#hasEvents">
    <rdfs:domain rdf:resource="#EventHistory"/>
    <owl:inverseOf rdf:resource="#isEventOf"/>
    <rdfs:range rdf:resource="#Event"/>
  </owl:ObjectProperty>
  <owl:FunctionalProperty rdf:ID="otext">
```

```
      <rdfs:domain rdf:resource="#Event"/>
      <rdf:type
rdf:resource="http://www.w3.org/2002/07/owl#DatatypeProperty"/>
      <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#string"/>
    </owl:FunctionalProperty>
    <owl:FunctionalProperty rdf:ID="id2">
      <rdf:type
rdf:resource="http://www.w3.org/2002/07/owl#DatatypeProperty"/>
      <rdfs:domain rdf:resource="#Event"/>
      <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#string"/>
    </owl:FunctionalProperty>
    <owl:FunctionalProperty rdf:ID="hasText">
      <rdfs:range>
        <owl:DataRange>
          <owl:oneOf rdf:parseType="Resource">
            <rdf:rest rdf:parseType="Resource">
              <rdf:first
rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
              >Transient fault</rdf:first>
              <rdf:rest rdf:parseType="Resource">
                <rdf:rest rdf:parseType="Resource">
                  <rdf:first
rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
                  >Semi-permanent fault</rdf:first>
                  <rdf:rest rdf:parseType="Resource">
                    <rdf:rest rdf:resource="http://www.w3.org/1999/02/22-
rdf-syntax-ns#nil"/>
                    <rdf:first
rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
                    >Repeated semi-permanent faults -&gt; May need a
check</rdf:first>
                  </rdf:rest>
                </rdf:rest>
                <rdf:first
rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
                >Relay is again open</rdf:first>
              </rdf:rest>
            </rdf:rest>
            <rdf:first
rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
            >Relay is first-time open</rdf:first>
          </owl:oneOf>
        </owl:DataRange>
      </rdfs:range>
      <rdf:type
rdf:resource="http://www.w3.org/2002/07/owl#DatatypeProperty"/>
      <rdfs:domain rdf:resource="#Fault"/>
    </owl:FunctionalProperty>
    <owl:FunctionalProperty rdf:ID="EventTime">
      <rdf:type
rdf:resource="http://www.w3.org/2002/07/owl#DatatypeProperty"/>
      <rdfs:domain rdf:resource="#Event"/>
      <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#time"/>
    </owl:FunctionalProperty>
    <owl:FunctionalProperty rdf:ID="restext">
      <rdfs:domain rdf:resource="#Event"/>
```

```
      <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#string"/>
      <rdf:type
rdf:resource="http://www.w3.org/2002/07/owl#DatatypeProperty"/>
    </owl:FunctionalProperty>
    <owl:FunctionalProperty rdf:ID="EventDate">
      <rdfs:domain rdf:resource="#Event"/>
      <rdf:type
rdf:resource="http://www.w3.org/2002/07/owl#DatatypeProperty"/>
      <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#date"/>
    </owl:FunctionalProperty>
    <owl:FunctionalProperty rdf:ID="PT">
      <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#string"/>
      <rdf:type
rdf:resource="http://www.w3.org/2002/07/owl#DatatypeProperty"/>
      <rdfs:domain rdf:resource="#Event"/>
    </owl:FunctionalProperty>
    <owl:FunctionalProperty rdf:ID="ovalue">
      <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#string"/>
      <rdf:type
rdf:resource="http://www.w3.org/2002/07/owl#DatatypeProperty"/>
      <rdfs:domain rdf:resource="#Event"/>
    </owl:FunctionalProperty>
    <owl:FunctionalProperty rdf:ID="object_id">
      <rdf:type
rdf:resource="http://www.w3.org/2002/07/owl#DatatypeProperty"/>
      <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#string"/>
      <rdfs:domain rdf:resource="#Event"/>
    </owl:FunctionalProperty>
    <owl:FunctionalProperty rdf:ID="EventTimeStamp">
      <rdf:type
rdf:resource="http://www.w3.org/2002/07/owl#DatatypeProperty"/>
      <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#string"/>
      <rdfs:domain rdf:resource="#Event"/>
    </owl:FunctionalProperty>
    <owl:FunctionalProperty rdf:ID="id1">
      <rdf:type
rdf:resource="http://www.w3.org/2002/07/owl#DatatypeProperty"/>
      <rdfs:domain rdf:resource="#Event"/>
      <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#string"/>
    </owl:FunctionalProperty>
    <owl:FunctionalProperty rdf:ID="RI">
      <rdf:type
rdf:resource="http://www.w3.org/2002/07/owl#DatatypeProperty"/>
      <rdfs:domain rdf:resource="#Event"/>
      <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#string"/>
    </owl:FunctionalProperty>
    <owl:FunctionalProperty rdf:ID="object">
      <rdf:type
rdf:resource="http://www.w3.org/2002/07/owl#DatatypeProperty"/>
      <rdfs:domain rdf:resource="#Event"/>
      <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#string"/>
    </owl:FunctionalProperty>
    <owl:FunctionalProperty rdf:ID="UnprocessedEvent">
      <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#boolean"/>
      <rdf:type
rdf:resource="http://www.w3.org/2002/07/owl#DatatypeProperty"/>
```

```
      <rdfs:domain rdf:resource="#Event"/>
    </owl:FunctionalProperty>
    <Fault rdf:ID="AgainOpenRelay">
      <hasText rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
      >Relay is again open</hasText>
    </Fault>
    <Fault rdf:ID="RepeatedSemiPermanentFaults">
      <hasText rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
      >Repeated semi-permanent faults -&gt; May need a check</hasText>
    </Fault>
    <Fault rdf:ID="SemiPermanentFault">
      <hasText rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
      >Semi-permanent fault</hasText>
    </Fault>
    <Fault rdf:ID="FirstTimeOpenRelay">
      <hasText rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
      >Relay is first-time open</hasText>
    </Fault>
    <Fault rdf:ID="TransientFault">
      <hasText rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
      >Transient fault</hasText>
    </Fault>
  </rdf:RDF>

  <!-- Created with Protege (with OWL Plugin 3.3.1, Build 430)
http://protege.stanford.edu -->
```

UBIWARE Deliverable D1.2:
Workpackage WP7:

APPENDIX C - *Extracting a hierarchy of nodes*

```
// --- Rule for handling the request for tree loading ---
{
{{get tree ?root} gb:eventFrom ?socket. ?socket != GUI}
>>{
      gb:I gb:add {
       ?socket :hasXML {
            {00 :tag html} :branch {
             {00 :tag style} :branch {
                  {0 :attribute type} :leaf "text/css".
                   0 :leaf "td {width:150px; text-align:center;}
                              a:link {color:#000000}
                              a:visited {color:#000000}
                              a:active {color:#000000}
                              a:hover {color:#888888}"
                  }.
             {01 :tag body} :branch {
              {00 :tag H3} :branch {00 :leaf "DPM tree"}.
              {01 :tag table} :branch {
                {00 :tag tr} :branch{
                 {00 :tag td} :branch {
                   {01 :tag A} :branch {
                        {00 :attribute href} :leaf
                                          "get nodeHistory nodeid".
                        01 :leaf "Nodeid"
                   }.
                 }
                }
              }
             }
            }
       }.
       {?child :partOf *. ?child :hasName ?childname.
        gb:I gb:doNotBelieve {?socket :hasTriplesReady true}.
        ?socket :hasXML {{* :tag html} :branch {
            {* :tag body} :branch {
              {* :tag table} :branch ?tabobj}}}}
```

```
       =>{
          {?tabobj gb:hasMember {
            {00 :tag tr} :branch {
             {00 :tag td} :branch{
               {01 :tag A} :branch {
                {00 :attribute href} :leaf "get :nodeHistory ?child".
                {01 :attribute target} :leaf "content".
                 02 :leaf "?childname"
               }
              }
             }
            }
            } gb:All ?child.
                                                    ?socket   :hasTriplesReady
true
       }
  }.
  gb:I gb:remove {{get tree ?root} gb:eventFrom ?socket}
 }
} gb:is gb:Rule.
```