IOG

GENERAL NETWORKING FRAMEWORK (GNF)

DELIVERABLE 3.1

Technical report

SmartResource: Proactive self-maintained resources in Semantic Web

16.05.06

University of Jyväskylä

Agora Center

Author: Industrial Ontologies Group

Contact Information: e-mail: vagan@it.jyu.fi

Title: General Networking Framework (GNF), Deliverable 3.1

Work: Technical report

Status of document: working draft

URL: http://www.cs.jyu.fi/ai/OntoGroup/Deliverable3_1.htmT

Number of Pages: 24

Additional Appendix: SmartResource Project: 3-rd year (2006) Status Report Presentation (which includes details on General Networking Framework concepts and architectures) – 51 slide

Abbreviations

RSCDF – Resource State/Condition Description Framework

RGBDF – Resource Goal/Behavior Description Framework

GUN – Global Understanding eNvironment

Contents

1 Introduction

Our intention is to make devices proactive in a sense that they can analyze their state independently from other systems and applications, initiate and control own maintenance proactively. Resource state can provide knowledge about resource condition, whereas both resource condition and goal of the resource will result in certain behavior of active resource towards effective and predictive maintenance.

The main objective of the Industrial Ontologies Group is to contribute to fast adoption of Semantic Web and related technologies to local and global industries. It includes research and development aimed to design a Global Understanding Environment (GUN) as next generation of Web-based platforms by making heterogeneous industrial resources (files, documents, services, devices, business processes, systems, organizations, human experts, etc.) web-accessible, proactive and cooperative in a sense that they will be able to automatically plan own behavior, monitor and correct own state, communicate and negotiate among themselves depending on their role in a business process, utilize remote experts, Web-services, software agents and various Web applications. Three fundamentals of such platform are Interoperability, Automation and Integration. Interoperability in GUN requires utilization of Semantic Web standards, RDF-based metadata and ontologies and semantic adapters for the resources. Automation in GUN requires proactivity of resources based on applying the agent technologies. Integration in GUN requires ontology-based business process modeling and integration and multi-agent technologies for coordination of business processes over resources.

The SmartResource project in its research and development efforts analyzes Global Understanding Environment decomposing it into three frameworks: General Adaptation Framework (GAF) [1], [2], [3], General Proactivity Framework (GPF) [4], [5] and General Networking Framework (GNF).

Following the aim of Industrial Ontologies Group (IOG), finally, GUN Ontology will be a result of an integration of RS/CDF-based [3] domain ontology, RG/BDF-based [4], [5] domain ontology and RP/IDF-based domain ontology and should be able to include various available models for describing all GAF, GPF and GNF – related domains.

The conceptual difference between RS/CDF, RG/BDF and RP/IDF is shown in Figure 1.

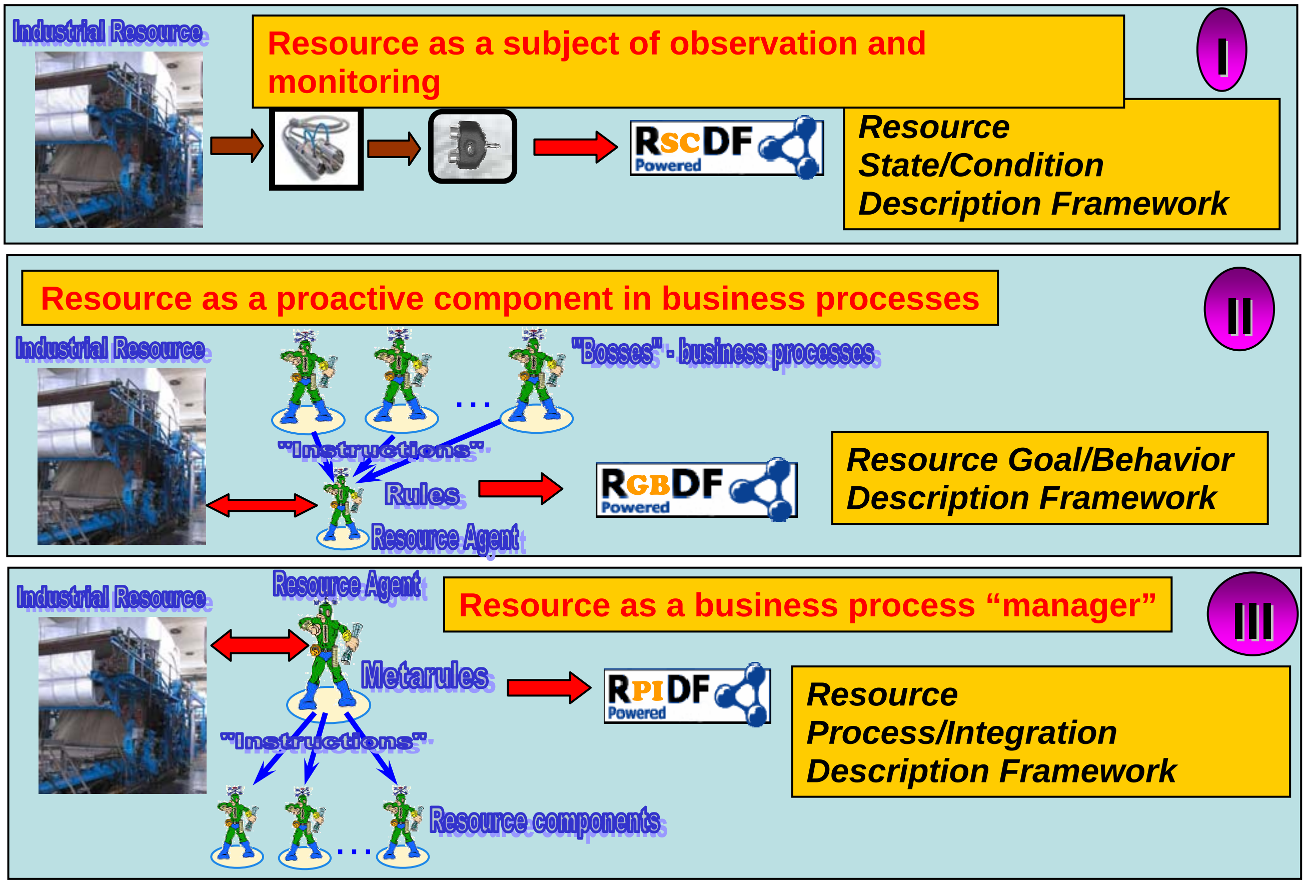

Figure 1

The General Networking Framework (GNF) considers an opportunity of ontological modeling of business processes as integration of component behavioral models of various business actors (agents representing smart resources in the web) in such a way that this integration will constitute the behavioral model of an agent responsible for the “alliance” of the components. This means that such “corporate” agent will monitor behaviors of the proactive components against the constraints provided by the integration scenario. Such model is naturally recursive and this means that the corporate agent can be a component in a more complex business process and will be monitored itself by an agent from the more higher level of hierarchy. Hierarchy of agents can be considered as possible mapping from the part-of ontological hierarchy of the domain resources.

The above motivates the main research objective of SmartResource project in 2006: “Design of a General Networking Framework as a platform for integration individual behaviors of proactive smart resources into a business process with opportunity to manage the reliability of components by certification, personal trust evaluations and exchange”.

2 Process as a resource in GUN environment

2.1 SmartResource Agent Architecture

The main feature of the SmartResource Platform is process performance via Resource Agents communication. All the Platform Agents are designed in a common way to provide interoperability and common approach. General SmartResource Agent Architecture is represented in Figure 2.

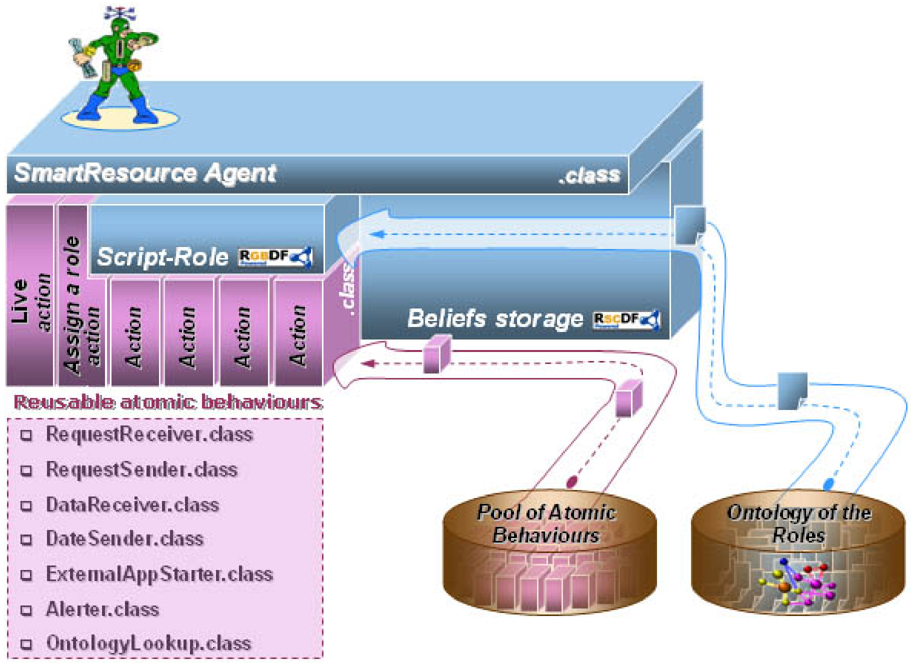

Figure 2

The main functionality of the Agent is based on performance of a behavior (set of behaviour-rules) that corresponds to assigned role. The behaviour description is a RG/BDF based script, which can be loaded from the ontology of the Roles. Based on RG/BDF Script-Role and RS/CDF based beliefs descriptions, Agent runs reusable atomic behaviours – Actions (executable modules). Action performance results change of Environment State. In another words, this performance modifies Agent beliefs. Such atomic behaviours can be downloaded from remote pool of atomic behaviours on demand. But basic set of them and frequently used Actions can be placed locally on the Agent platform. The basic set of Actions that recently used in current prototype of SmartResource Platform is represented in Figure 2.

2.2 Process – GUN resource

Before we will talk about process integration issues, let us answer the question: What is a process in GUN environment? Accordingly first axiom (see Figure 3) of the Global Understanding eNvironment, Process – is similar resource to other resources in GUN (Device, Service and Human/Expert), but does not belong to the world of physical resources. As all GUN resources, Process has own properties that describe Process's state, history, sub processes and belongingness to upper-process (super-process). Thus, following principles of GUN resource, each Process is enhanced with an Agent that serves Process as a resource and actually realizes it as a behavior engine. Each process is a sequence of the actions (rgbdfs:Execution) that results in achievement of the final goal. So, each Agent per se is a process. In this case Agent Behavior plays role of a sequence of the actions and final result is represented by Agent Goal.

Axiom 1: ***Each resource in dynamic Industrial World is a process and each process in this world is a resource.***

Axiom 2: ***Hierarchy of subordination among resource agents in GUN corresponds to the "part-of" hierarchy of the Industrial World resources.***

Figure 3

Each GUN resource can theoretically be involved to several processes, appropriate commitments and activities, which can be either supplementary or contradictory. This means that the resource is part of several more complex resources and its role within each of the resource might be different.

There are some models of upper-process organization. But before we will talk about these models, we should state some definition. Let us consider executable module as an atomic non configurable actions. Thus, the choreography of a subject resource by its Agent via action performing is a non configurable atomic leaf-process. In this case, Agents behave accordingly to certain plan – planned set of behaviours. But, such simple processes can be organized in alliances – Process. The main function of a Process-Agent is the orchestration of a set of sub processes. Following this approach, architectures of arbitrary nested processes can be built, where leaf-processes are physical world Resource-Agents (Device-Agent, Service-Agent and Human/Expert-Agent).

One aim of Process (upper-process) creation is to organize cooperative work of sub processes for improving their individual performance. Each Agent should be supplied with a behaviour-planer module that generates plan for behaviour performance without any conflicts. And in this particular case, Process-Agent should utilize behaviour-planer to build plan of sub processes cooperative work and set constraints on their own plans. Another aim Process creation is to utilize other processes to reach another separate, lat us say - group-goal. In this case, achievement of the sub processes' goals depends on commitments and contracts between all parties. Thus, Agent-owner of this group-goal plays two roles: role of the sub process as another sub processes in this Process (with one difference – it has just goal and does not have atomic behaviour) and role of Process-Agent that performs orchestration of the sub processes. If we separate these two roles, we come to first model where we have blank sub process (has just goal and does not have any atomic behaviour) among sub processes, but achievement of this group-goal takes biggest priority. Figure 4 shows us generalized model when Process-Agent replans sub processes behaviours accordingly to sub processes goals achievement priorities.

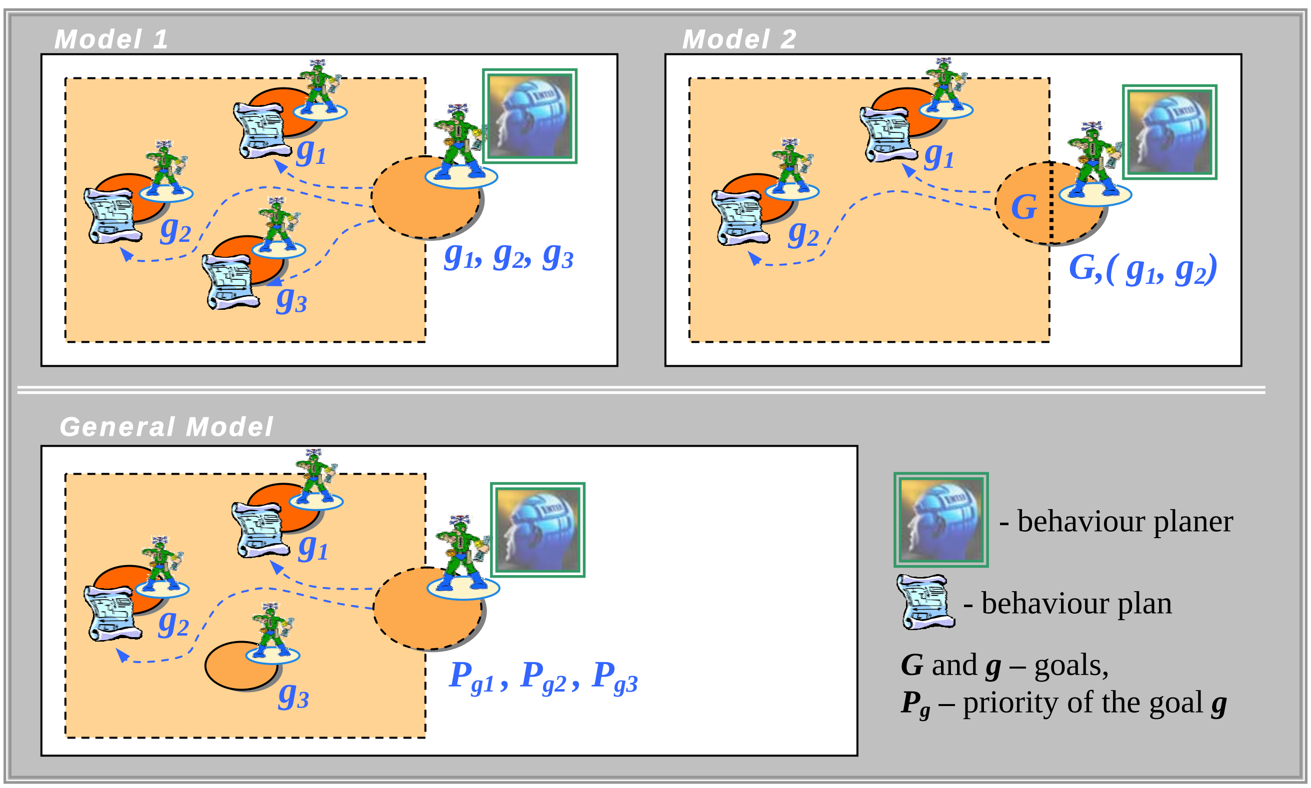

Figure 4

Nobody can guarantee stability of an environmental data if data space is shared among several Processes. It brings a need to replan the behaviour depending on the changes. The optimal way to reduce amount of replans is to collect all Processes that share same data space under one upper-process, if it is possible.

Generally, all the behaviours are represented by the set of rules that operate with the classes of resources (not the concrete instances). But during the behaviour processing by Behavior Engine all the rules are bounded with concrete instances. After such bindings we may have the conflict situations. If two processes use different instance spaces (spaces of facts, desires and etc.), then no conflicts may happen. But, if they share the same instance space, they can block others process performance by changing the shared information space. Actually while those Resource-Agents are living separately (resources are not members of some biggest process), no one cares about this conflicts of performance and they are concentrated just on achievement of the own goals. But when those two processes are members of another bigger upper-process, the duty of the Process-Agent is to resolve the conflicts via setting the constraints for behaviours of its members to reach the own goal and goals of the members (if it has been mentioned in a contract of the process). Initial behaviour of Process-Agent contains such set of actions as:

- Collection of all the behaviors of process members and convert them to the set of rules;
- Applying an algorithm to build a sequence of actions (performance plan) for optimal achievement of a final goal and intermediate goals (if necessary) based on behavior-rules of sub processes;
- Setting the constraints on behaviours of the members for conflict situations (when several rules may be applied, but result the different states – Environment State). In another words, we have a need to define and provide the meta-behavior-rules for the sub processes.

Such constraints (for process behaviour-rules) change behaviour of the Resource-Agent and restrict the degrees of an Agent freedom. Actually with its degree of freedom sub process sacrifices to upper-process when becomes a part of it. It is not necessary, that it negatively affects sub process's goal achievement, but often the opposite – it can result to speedup of the goal achievement.

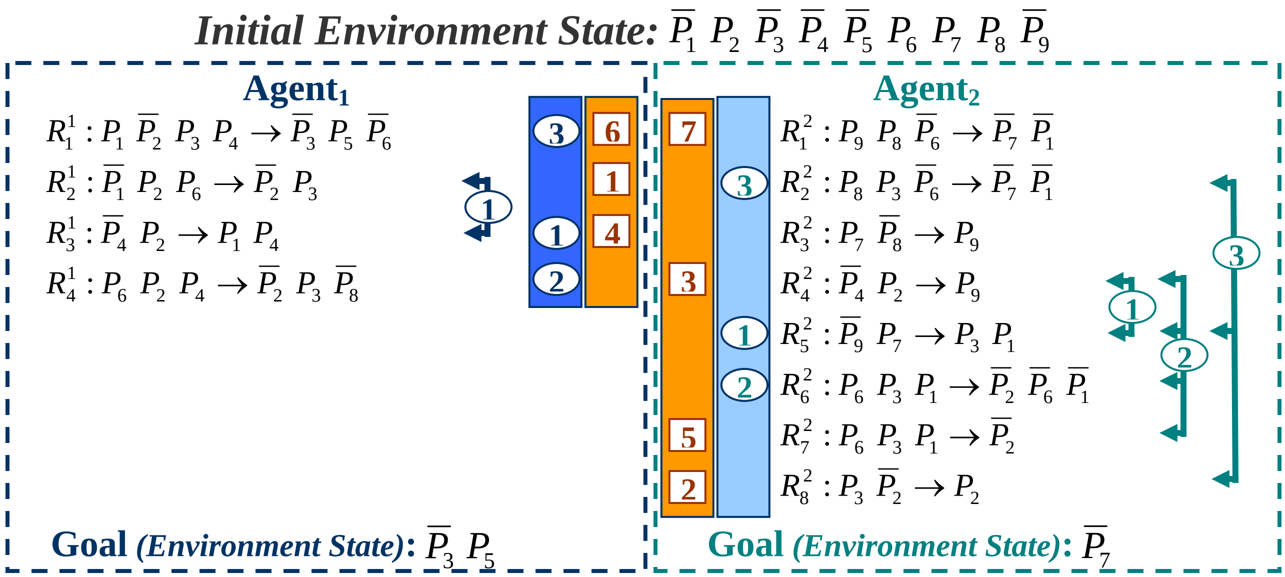

Figure 5

Let us consider an example that explains some case for process integration. For the easiest rule representation we will use Production Model of knowledge representation. But we should remember that Agent operates with the RG/BDF behaviours, not with the rules. RG/BDF behaviour is a subclass of RG/BDF rule and has Execution (Executable module) in the right part. In turn, each Execution (action performance) results to certain changes in the Environment State. Further, in figures, we convert the behaviours to the rules for easier explanation. In Figure 5 we can see two sets of rules that are behaviours of two separate Agents. Each of them has own goal sub state of the Environment (a sub set of Environment statements) and shares the common State of Environment. Numbers in the circles show the order of rule applying for each rule set to optimally achieve the correspondent goal. Also from the arrows you can see the rules that can be applied at the same time for current state of the Environment.

From the Figure 6 we can see rule set of the process that is an upper-process for previous two processes. This rule set is a combination of the sub-processes rules. Again from the figure we can see the final goal of this upper-process and order of rule applying. This order also is shown in Figure 6 via numbers in squares. And now we can see that rule orders of sub processes and order of upper-process are different. This is because upper-process is aimed to resolve the conflicts between sub processes and organizes their cooperative work.

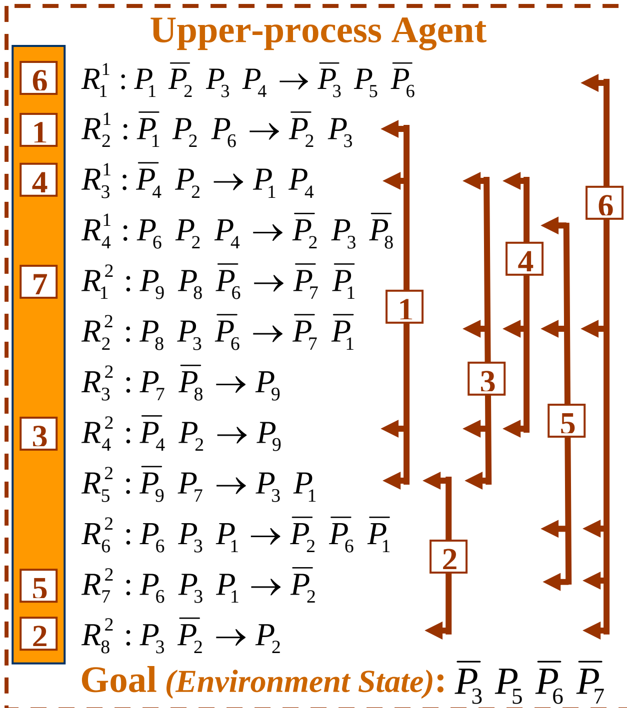

Figure 6

For example, the rule order $\{R_3^1, R_4^1, R_1^1\}$ of Agent$_{1}$ is the optimal one to reach the goal. Rule R_3^1 will be applied first, even if on the first step two rules $\{R_2^1, R_3^1\}$ can be applied, but applying of the rule R_2^1 result stop of the process. This is because P_1 and P_4 can not be achieved any more until some another process updates (changes) the Environment State with P_1 and P_4. But for the upper-process, that is aimed to achieve own goal and sub goals (goals of the sub processes), the rule R_2^1 should be applied first, because the rule order $\{R_2^1, R_8^2, R_4^2, R_3^1, R_7^2, R_1^1, R_1^2\}$ is optimal order for conflict resolution and achievement of all the goals. Also the rule R_4^1 should not be applied in any case, because it results stop of the second process (Agent$_{2}$). In that case, statement P_8 will not be achieved by this process.

Taking into account all the above, the main functionality of the upper-process is to define the rule constraints for sub processes with the aim to realize orchestration of them. With all this we come to meta-rules for Agent's Behaviours. Figure 7 shows us meta-rule enhanced Agent Behaviours (Process$_{1}$ and Process$_{2}$). Now the rule order of upper-process is determined by the constraints of sub processes' behaviour-rules. But as we mentioned before, Agent operates with the

RG/BDF Behaviours, not with the rules. In this case, meta-rule enhancement means Agent behavior-rules set extending with additional behaviours that plays role of meta-rule and switch the behavior-rules conditions.

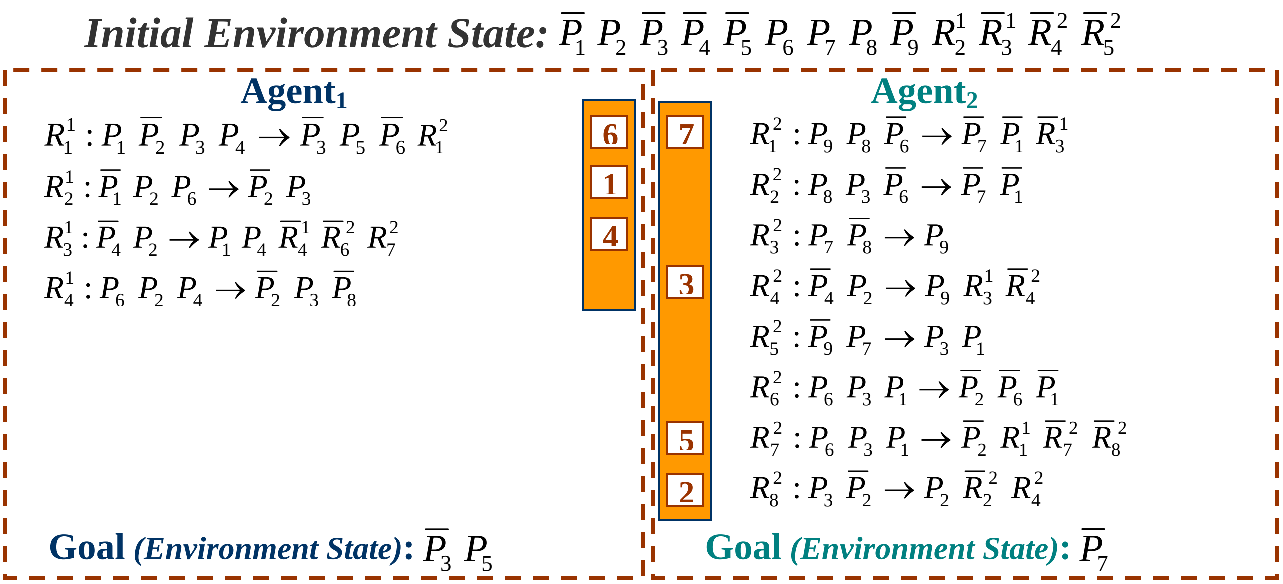

Figure 7

```
<rdfs:Class rdf:about="&rgbdfs;RuleConditionSetter">
      <rdfs:comment>Executable module that sets rule
                    condition value</rdfs:comment>
      <rdfs:label>RuleConditionSetter</rdfs:label>
      <rdfs:subClassOf rdf:resource="&rgbdfs;Execution"/>
</rdfs:Class>

<rdf:Property rdf:about="&rgbdfs;subjectRule">
      <rdfs:label>subjectRule</rdfs:label>
      <rdfs:domain rdf:resource="&rgbdfs;RuleConditionSetter"/>
      <rdfs:range rdf:resource="&rgbdfs;RuleStatement"/>
</rdf:Property>

<rdf:Property rdf:about="&rgbdfs;ruleCondition">
      <rdfs:label>ruleCondition</rdfs:label>
      <rdfs:domain rdf:resource="&rgbdfs;RuleConditionSetter"/>
      <rdfs:range rdf:resource="&rgbdfs;RuleCondition"/>
</rdf:Property>
```

Figure 8

Upper-process Agent should provide these additional behavior-rules with the necessary RG/BDF Executions (atomic executable modules - Actions) that perform a behaviour-rules condition switching. Following this approach it makes sense to extend RG/BDFS with special RG/BDFS

Execution, which changes the rule condition. We define ***rgbdfs:RuleConditionSetter*** as a subclass of rgbdfs:Execution and supply this class with two properties: ***rgbdfs:subjectRule*** and ***rgbdfs:ruleCondition*** (see Figure 8). Thus, Action (atomic executable module), that changes rule condition, gets as an input certain instance of rgbdfs:RuleConditionSetter class and references to subject rule (its condition should be set) and condition value itself. As a result of such Action performance, correspondent Fact Statement about rule condition will be added to Active Data Space.

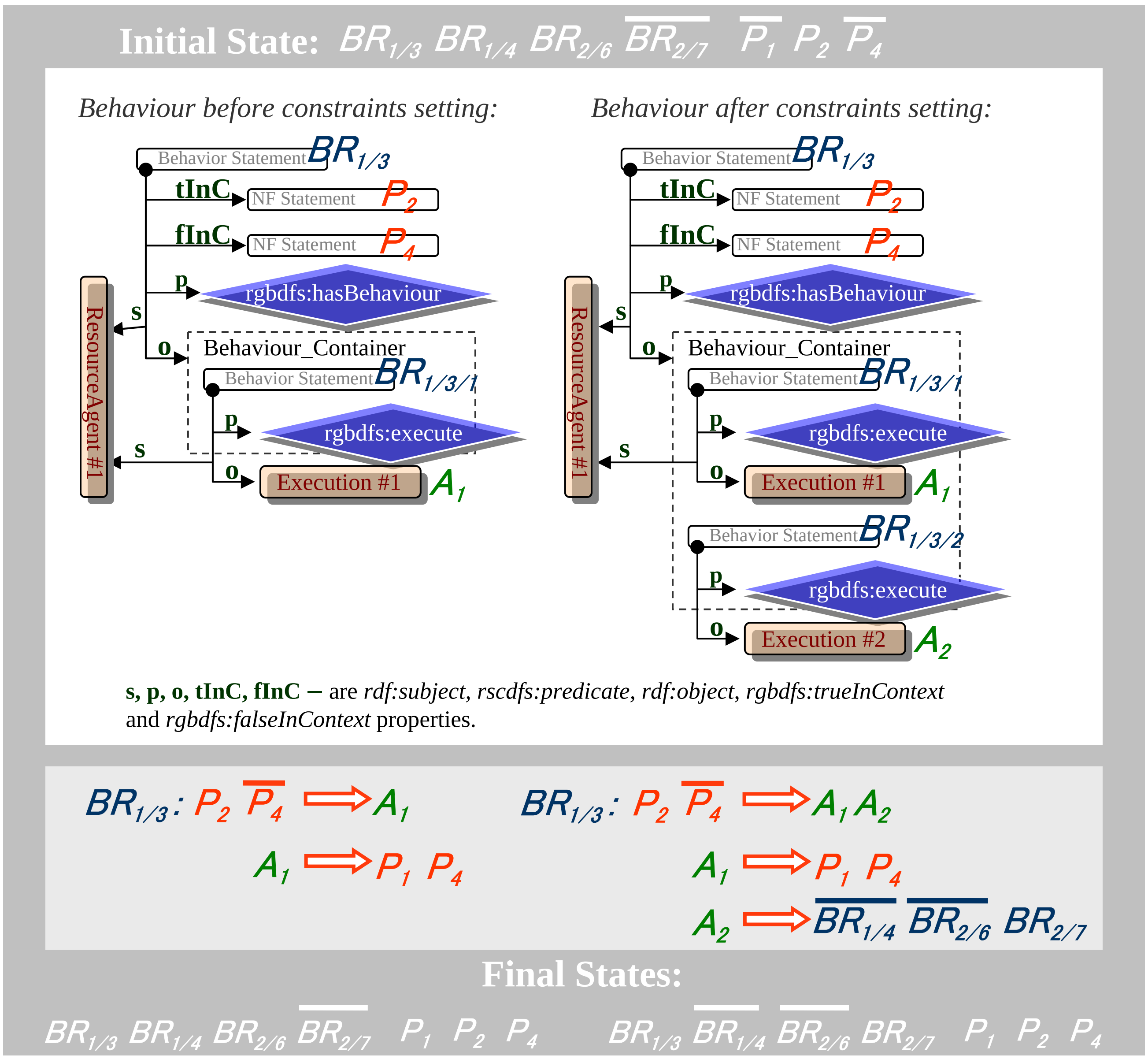

Figure 9

The RG/BDF behaviour-rule description approach fits very well the constraints definition via adding a restriction behaviour statement. Figure 9 shows an example of RG/BDF representation of behaviour-rule R_3^1 before and after the constraints adding (also you can take a look on RDF/XML serialization that is represented in an Appendix).

3 Conclusion

Each industrial resource can theoretically be involved to several processes, appropriate commitments and activities, which can be either supplementary or contradictory. This means that the resource is part of several more complex resources and its role within each of the resource might be different. Modeling such resources with GUN can be provided by appropriate resource agent, which can make clones of it and distribute all necessary roles among them. Each industrial resource, which joins some commitment, will behave according to restrictions the rules of that commitment require. The more commitments individual resource takes, the more restriction will be put on its behavior. The main feature of the General Networking Framework is smart way of managing commitments (processes and contracts) of any proactive world resource (SmartResource) to enable cooperative behavior of it towards reaching also group goals together with the individual ones. Taking into account that world of industrial products and processes has multilevel hierarchy (based on *part_of* relation), we can say that it results to a hierarchical structure of GUN agents, which are meant to monitor appropriate world components in a cooperative manner.

4 References

[1] Kaykova O., Khriyenko O., Kovtun D., Naumenko A., Terziyan V., Zharko A., Challenges of General Adaptation Framework for Industrial Semantic Web, In: A. Sheth and M. Lytras (eds.), Advanced Topics in Semantic Web, Idea Group, Vol. 1, 33 pp. (to appear).

[2] Kaykova O., Khriyenko O., Naumenko A., Terziyan V., Zharko A., RSCDF: A Dynamic and Context Sensitive Metadata Description Framework for Industrial Resources, In: Eastern-European Journal of Enterprise Technologies, Vol.3, No.3, June 2005, ISSN: 1729-3774, 30 pp.

[3] Khriyenko O., Terziyan V., A Framework for Context-Sensitive Metadata Description, In: International Journal of Metadata, Semantics and Ontologies, ISSN 1744-2621, 11 pp. (to appear).

[4] Kaykova O., Khriyenko O., Terziyan V., Zharko A., RGBDF: Resource Goal and Behaviour Description Framework, In: M. Bramer and V. Terziyan (Eds.): Industrial Applications of Semantic Web, Proceedings of the 1-st International IFIP/WG 12.5 Working Conference IASW-2005, August 25-27, 2005, Jyvaskyla, Finland, Springer, IFIP, Vol.188, pp. 83-99.

[5] Khriyenko O., Proactivity Layer of the Smart Resource in Semantic Web, In: 17th International Conference on Database and Expert Systems Applications - DEXA '06 , September 4-8, 2006, Andrzej Frycz Modrzewski Cracow College, Krakow, Poland, Springer, LNCS, 10 pp. (submitted 7 March, 2006).

5 APPENDIX

Behaviour performance without restrictions setting:

Mental Agent Data:


```
<rgbdfs:Behaviour_Statement rdf:about="BR1/3"
   rdfs:comment="Behaviour statement"
   rdfs:label="BR1/3">
  <rgbdfs:subject rdf:resource="Agent1"/>
  <rgbdfs:predicate rdf:resource="rgbdfs:has_Behaviour"/>
  <rdf:object rdf:resource="BContainer1/3"/>
  <rgbdfs:trueInContext rdf:resource="P2"/>
  <rgbdfs:falseInContext rdf:resource="P4"/>
</rgbdfs:Behaviour_Statement>

<rgbdfs:Behaviour_Statement rdf:about="BR1/4"
   rdfs:comment="Behaviour statement"
   rdfs:label="BR1/4">
   ...
</rgbdfs:Behaviour_Statement>

<rgbdfs:Behaviour_Statement rdf:about="BR2/6"
   rdfs:comment="Behaviour statement"
   rdfs:label="BR2/6">
   ...
</rgbdfs:Behaviour_Statement>

<rgbdfs:Behaviour_Statement rdf:about="BR2/7"
   rdfs:comment="Behaviour statement"
   rdfs:label="BR2/7">
   ...
</rgbdfs:Behaviour_Statement>

<rgbdfs:Behaviour_Container rdf:about="BC1/3"
   rdfs:comment="Behaviour container"
   rdfs:label="BC1/3">
  <rgbdfs:bMember rdf:resource="BS1/3/1"/>
</rgbdfs:Behaviour_Container>

<rgbdfs:Behaviour_Statement rdf:about="BR1/3/1"
   rdfs:comment="Behaviour statement"
   rdfs:label="BR1/3/1">
  <rgbdfs:subject rdf:resource="Agent1"/>
  <rgbdfs:predicate rdf:resource="rgbdfs:execute"/>
  <rdf:object rdf:resource="Execution1"/>
</rgbdfs:Behaviour_Statement>

<rgbdfs:NF_Statement rdf:about="P1"
   rdfs:comment="Describes statement P1"
   rdfs:label="P1">
   ...
</rgbdfs:NF_Statement>

<rgbdfs:NF_Statement rdf:about="P2"
```

```
   rdfs:comment="Describes statement P2"
   rdfs:label="P2">
   ...
</rgbdfs:NF_Statement>

<rgbdfs:NF_Statement rdf:about="P4"
   rdfs:comment="Describes statement P4"
   rdfs:label="P4">
   ...
</rgbdfs:NF_Statement>

<rgbdfs:ResourceAgent rdf:about="Agent1"
   rdfs:comment="Resource Agent instance"
   rdfs:label="Agent1">
   ...
</rgbdfs:ResourceAgent>

<rgbdfs:Execution rdf:about="Execution1"
   rdfs:comment="Execution instance – Atomic Action that also results
                 appearance of the fact statements: P1 and P4"
   rdfs:label="Execution1">
   ...
</rgbdfs:Execution>
```

Agent believes (Fact Data) - before performance:

```
<rscdfs:Statement rdf:about="BR1/3_condition"
   rdfs:comment="Describes a condition of the behaviour-rule statement"
   rdfs:label="BR1/3_condition">
  <rdf:subject rdf:resource="BR1/3"/>
  <rscdfs:predicate rdf:resource="rgbdfs:ruleConditionIs"/>
  <rdf:object rdf:resource="rgbdfs:Active"/>
</rscdfs:Statement>

<rscdfs:Statement rdf:about="BR1/4_condition"
   rdfs:comment="Describes a condition of the behaviour-rule statement"
   rdfs:label="BR1/4_condition">
  <rdf:subject rdf:resource="BR1/4"/>
  <rscdfs:predicate rdf:resource="rgbdfs:ruleConditionIs"/>
  <rdf:object rdf:resource="rgbdfs:Active"/>
</rscdfs:Statement>

<rscdfs:Statement rdf:about="BR2/6_condition"
   rdfs:comment="Describes a condition of the behaviour-rule statement"
   rdfs:label="BR2/6_condition">
  <rdf:subject rdf:resource="BR2/6"/>
  <rscdfs:predicate rdf:resource="rgbdfs:ruleConditionIs"/>
  <rdf:object rdf:resource="rgbdfs:Active"/>
</rscdfs:Statement>

<rscdfs:Statement rdf:about="BR2/7_condition"
   rdfs:comment="Describes a condition of the behaviour-rule statement"
   rdfs:label="BR2/7_condition">
  <rdf:subject rdf:resource="BR2/7"/>
  <rscdfs:predicate rdf:resource="rgbdfs:ruleConditionIs"/>
  <rdf:object rdf:resource="rgbdfs:Active"/>
</scbdfs:Statement>
```

```
<rscdfs:Statement rdf:about="P2"
   rdfs:comment="Describes fact statement P2"
   rdfs:label="P2">
   ...
</rscdfs:Statement>
```

Agent believes (Fact Data) - after performance:

```
<rscdfs:Statement rdf:about="BR1/3_condition"
   rdfs:comment="Describes a condition of the behaviour-rule statement"
   rdfs:label="BR1/3_condition">
  <rdf:subject rdf:resource="BR1/3"/>
  <rscdfs:predicate rdf:resource="rgbdfs:ruleConditionIs"/>
  <rdf:object rdf:resource="rgbdfs:Active"/>
</rscdfs:Statement>

<rscdfs:Statement rdf:about="BR1/4_condition"
   rdfs:comment="Describes a condition of the behaviour-rule statement"
   rdfs:label="BR1/4_condition">
  <rdf:subject rdf:resource="BR1/4"/>
  <rscdfs:predicate rdf:resource="rgbdfs:ruleConditionIs"/>
  <rdf:object rdf:resource="rgbdfs:Active"/>
</rscdfs:Statement>

<rscdfs:Statement rdf:about="BR2/6_condition"
   rdfs:comment="Describes a condition of the behaviour-rule statement"
   rdfs:label="BR2/6_condition">
  <rdf:subject rdf:resource="BR2/6"/>
  <rscdfs:predicate rdf:resource="rgbdfs:ruleConditionIs"/>
  <rdf:object rdf:resource="rgbdfs:Active"/>
</rscdfs:Statement>

<rscdfs:Statement rdf:about="BR2/7_condition"
   rdfs:comment="Describes a condition of the behaviour-rule statement"
   rdfs:label="BR2/7_condition">
  <rdf:subject rdf:resource="BR2/7"/>
  <rscdfs:predicate rdf:resource="rgbdfs:ruleConditionIs"/>
  <rdf:object rdf:resource="rgbdfs:Active"/>
</scbdfs:Statement>

<rscdfs:Statement rdf:about="P1"
   rdfs:comment="Describes fact statement P1"
   rdfs:label="P1">
   ...
</rscdfs:Statement>

<rscdfs:Statement rdf:about="P2"
   rdfs:comment="Describes fact statement P2"
   rdfs:label="P2">
   ...
</rscdfs:Statement>

<rscdfs:Statement rdf:about="P4"
   rdfs:comment="Describes fact statement P4"
```

```
   rdfs:label="P4">
   ...
</rscdfs:Statement>
```

Behaviour performance with restrictions setting:

Mental Agent Data:

```
<rgbdfs:Behaviour_Statement rdf:about="BR1/3"
   rdfs:comment="Behaviour statement"
   rdfs:label="BR1/3">
  <rgbdfs:subject rdf:resource="Agent1"/>
  <rgbdfs:predicate rdf:resource="rgbdfs:has_Behaviour"/>
  <rdf:object rdf:resource="BContainer1/3"/>
  <rgbdfs:trueInContext rdf:resource="P2"/>
  <rgbdfs:falseInContext rdf:resource="P4"/>
</rgbdfs:Behaviour_Statement>

<rgbdfs:Behaviour_Statement rdf:about="BR1/4"
   rdfs:comment="Behaviour statement"
   rdfs:label="BR1/4">
   ...
</rgbdfs:Behaviour_Statement>

<rgbdfs:Behaviour_Statement rdf:about="BR2/6"
   rdfs:comment="Behaviour statement"
   rdfs:label="BR2/6">
   ...
</rgbdfs:Behaviour_Statement>

<rgbdfs:Behaviour_Statement rdf:about="BR2/7"
   rdfs:comment="Behaviour statement"
   rdfs:label="BR2/7">
   ...
</rgbdfs:Behaviour_Statement>

<rgbdfs:Behaviour_Container rdf:about="BC1/3"
   rdfs:comment="Behaviour container BC1/3"
   rdfs:label="BC1/3">
  <rgbdfs:bMember rdf:resource="BS1/3/1"/>
  <rgbdfs:bMember rdf:resource="BS1/3/2"/>
</rgbdfs:Behaviour_Container>

<rgbdfs:Behaviour_Statement rdf:about="BR1/3/1"
   rdfs:comment="Behaviour statement"
   rdfs:label="BR1/3/1">
  <rgbdfs:subject rdf:resource="Agent1"/>
  <rgbdfs:predicate rdf:resource="rgbdfs:execute"/>
  <rdf:object rdf:resource="Execution1"/>
</rgbdfs:Behaviour_Statement>

<rgbdfs:Behaviour_Statement rdf:about="BR1/3/2"
   rdfs:comment="Behaviour statement"
   rdfs:label="BR1/3/2">
  <rgbdfs:subject rdf:resource="Agent1"/>
```

```
  <rgbdfs:predicate rdf:resource="rgbdfs:execute"/>
  <rdf:object rdf:resource="Execution2"/>
</rgbdfs:Behaviour_Statement>

<rgbdfs:NF_Statement rdf:about="P1"
   rdfs:comment="Describes statement P1"
   rdfs:label="P1">
   ...
</rgbdfs:NF_Statement>

<rgbdfs:NF_Statement rdf:about="P2"
   rdfs:comment="Describes statement P2"
   rdfs:label="P2">
   ...
</rgbdfs:NF_Statement>

<rgbdfs:NF_Statement rdf:about="P4"
   rdfs:comment="Describes statement P4"
   rdfs:label="P4">
   ...
</rgbdfs:NF_Statement>
<rgbdfs:ResourceAgent rdf:about="Agent1"
   rdfs:comment="Resource Agent instance"
   rdfs:label="Agent1">
   ...
</rgbdfs:ResourceAgent>

<rgbdfs:Execution rdf:about="Execution1"
   rdfs:comment="Execution instance – Atomic Action that also results
                 appearance of the fact statements: P1 and P4"
   rdfs:label="Execution1">
   ...
</rgbdfs:Execution>

<rgbdfs:Execution rdf:about="Execution2"
   rdfs:comment="Execution instance – Atomic Action that results behaviour-
                 rule conditions change: BR1/4 – Passive, BR2/6 – Passive,
                 BR2/7 - Active"
   rdfs:label="Execution2">
   ...
</rgbdfs:Execution>
```

Agent believes (Fact Data) - before performance:

```
<rscdfs:Statement rdf:about="BR1/3_condition"
   rdfs:comment="Describes a condition of the behaviour-rule statement"
   rdfs:label="BR1/3_condition">
  <rdf:subject rdf:resource="BR1/3"/>
  <rscdfs:predicate rdf:resource="rgbdfs:ruleConditionIs"/>
  <rdf:object rdf:resource="rgbdfs:Active"/>
</rscdfs:Statement>

<rscdfs:Statement rdf:about="BR1/4_condition"
   rdfs:comment="Describes a condition of the behaviour-rule statement"
   rdfs:label="BR1/4_condition">
```

```
  <rdf:subject rdf:resource="BR1/4"/>
  <rscdfs:predicate rdf:resource="rgbdfs:ruleConditionIs"/>
  <rdf:object rdf:resource="rgbdfs:Active"/>
</rscdfs:Statement>

<rscdfs:Statement rdf:about="BR2/6_condition"
   rdfs:comment="Describes a condition of the behaviour-rule statement"
   rdfs:label="BR2/6_condition">
  <rdf:subject rdf:resource="BR2/6"/>
  <rscdfs:predicate rdf:resource="rgbdfs:ruleConditionIs"/>
  <rdf:object rdf:resource="rgbdfs:Active"/>
</rscdfs:Statement>

<rscdfs:Statement rdf:about="BR2/7_condition"
   rdfs:comment="Describes a condition of the behaviour-rule statement"
   rdfs:label="BR2/7_condition">
  <rdf:subject rdf:resource="BR2/7"/>
  <rscdfs:predicate rdf:resource="rgbdfs:ruleConditionIs"/>
  <rdf:object rdf:resource="rgbdfs:Active"/>
</scbdfs:Statement>

<rscdfs:Statement rdf:about="P2"
   rdfs:comment="Describes fact statement P2"
   rdfs:label="P2">
   ...
</rscdfs:Statement>
```

Agent believes (Fact Data) - after performance:

```
<rscdfs:Statement rdf:about="BR1/3_condition"
   rdfs:comment="Describes a condition of the behaviour-rule statement"
   rdfs:label="BR1/3_condition">
  <rdf:subject rdf:resource="BR1/3"/>
  <rscdfs:predicate rdf:resource="rgbdfs:ruleConditionIs"/>
  <rdf:object rdf:resource="rgbdfs:Active"/>
</rscdfs:Statement>

<rscdfs:Statement rdf:about="BR1/4_condition"
   rdfs:comment="Describes a condition of the behaviour-rule statement"
   rdfs:label="BR1/4_condition">
  <rdf:subject rdf:resource="BR1/4"/>
  <rscdfs:predicate rdf:resource="rgbdfs:ruleConditionIs"/>
  <rdf:object rdf:resource="rgbdfs:Passive"/>
</rscdfs:Statement>

<rscdfs:Statement rdf:about="BR2/6_condition"
   rdfs:comment="Describes a condition of the behaviour-rule statement"
   rdfs:label="BR2/6_condition">
  <rdf:subject rdf:resource="BR2/6"/>
  <rscdfs:predicate rdf:resource="rgbdfs:ruleConditionIs"/>
  <rdf:object rdf:resource="rgbdfs:Passive"/>
</rscdfs:Statement>

<rscdfs:Statement rdf:about="BR2/7_condition"
   rdfs:comment="Describes a condition of the behaviour-rule statement"
```

```
   rdfs:label="BR2/7_condition">
  <rdf:subject rdf:resource="BR2/7"/>
  <rscdfs:predicate rdf:resource="rgbdfs:ruleConditionIs"/>
  <rdf:object rdf:resource="rgbdfs:Active"/>
</scbdfs:Statement>

<rscdfs:Statement rdf:about="P1"
   rdfs:comment="Describes fact statement P1"
   rdfs:label="P1">
   ...
</rscdfs:Statement>

<rscdfs:Statement rdf:about="P2"
   rdfs:comment="Describes fact statement P2"
   rdfs:label="P2">
   ...
</rscdfs:Statement>

<rscdfs:Statement rdf:about="P4"
   rdfs:comment="Describes fact statement P4"
   rdfs:label="P4">
   ...
</rscdfs:Statement>
```