Deliverable 1.3
ZDM Management Rules

<table>
<thead>
<tr>
<th>Document version</th>
<th>Final</th>
</tr>
</thead>
<tbody>
<tr>
<td>Submission Date</td>
<td>04/10/2017</td>
</tr>
<tr>
<td>Dissemination Level</td>
<td>Public</td>
</tr>
<tr>
<td>Contribution to</td>
<td>WP 1</td>
</tr>
<tr>
<td>Document Owner</td>
<td>UNINOVA</td>
</tr>
<tr>
<td>File Name</td>
<td>GOOD MAN Deliverable 1.3</td>
</tr>
<tr>
<td>Revision</td>
<td>0.6</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Project Acronym</th>
<th>GOOD MAN</th>
</tr>
</thead>
<tbody>
<tr>
<td>Project Title</td>
<td>Agent oriented zero defect multi-stage manufacturing</td>
</tr>
<tr>
<td>Grant Agreement n.</td>
<td>723764</td>
</tr>
<tr>
<td>Call</td>
<td>H2020-IND-CE-2016-17</td>
</tr>
<tr>
<td>Project Duration</td>
<td>36 months, from 01/10/2016 to 30/09/2019</td>
</tr>
<tr>
<td>Website</td>
<td>go0dman-project.eu</td>
</tr>
</tbody>
</table>
Revision History

<table>
<thead>
<tr>
<th>REVISION</th>
<th>DATE</th>
<th>INVOLVED PARTNERS</th>
<th>DESCRIPTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.1</td>
<td>08/09/2017</td>
<td>UNINOVA</td>
<td>Deliverable structure</td>
</tr>
<tr>
<td>0.2</td>
<td>22/09/2017</td>
<td>UNINOVA</td>
<td>Overview of T1.3 including the specification of the rule design process, bridge to 2.2, as well as the implementation details.</td>
</tr>
<tr>
<td>0.3</td>
<td>28/09/2017</td>
<td>UNINOVA, IPB, BOC</td>
<td>Integration of IPB's contribution regarding Section 4, as well as of the feedback from the M12 meeting into the main document.</td>
</tr>
<tr>
<td>0.4</td>
<td>29/09/2017</td>
<td>UNINOVA</td>
<td>Addition of the executive summary, arrangement of the references, acronyms and final proof-reading.</td>
</tr>
<tr>
<td>0.5</td>
<td>01/10/2017</td>
<td>LOC</td>
<td>Feedback and document revision</td>
</tr>
<tr>
<td>0.6</td>
<td>02/10/2017</td>
<td>UNINOVA</td>
<td>Feedback integration</td>
</tr>
</tbody>
</table>

List of Contributors:

Mafalda Rocha (UNINOVA), Ricardo Silva Peres (UNINOVA), André Dionísio Rocha (UNINOVA), João Pedro Matos (UNINOVA), José Barbosa (IPB), Nelson Rodrigues (IPB), Wilfrid Utz (BOC), Cristina Cristalli (LOC)

Disclaimer: The information in this document is subject to change without notice. Company or product names mentioned in this document may be trademarks or registered trademarks of their respective companies.

All rights reserved.

The document is proprietary of the GOODMAN consortium members. No copying or distributing, in any form or by any means, is allowed without the prior written agreement of the owner of the property rights.

This document reflects only the authors’ view. The European Community is not liable for any use that may be made of the information contained herein.

GOOD MAN project has received funding from the European Union’s Horizon 2020 research and innovation programme under grant agreement No 723764.
Executive Summary

Rule-based engines date from the early 1970’s, having since evolved considerably with several applications in domains such as Expert Systems, Process Control and Real-time Stream Processing. The basic premise of such systems is that as input data arrives, it is monitored and matched against a set of conditions of interest, then taking appropriate action and these rules are triggered.

The main goal of this deliverable is to describe the demonstrator developed in Task 1.3, pertaining to the definition, design and implementation of the management rules and overall integration approach to reach the project’s goal of Zero-Defect Manufacturing.

For this purpose, a rule-based integration approach is proposed, effectively bridging the gap between the results of WP4 and WP2 related to the developments of both the Knowledge Management and Multi Agent Cyber-Physical System. This approach should translate the complexity of the knowledge capturing and advanced data analytics of WP4 into straightforward rules compliant with GOOD MAN’s common data representation format, thus enabling fast and decoupled reasoning at the shop-floor level.
# Table of Contents

1. Introduction...................................................................................................................... 6  
   1.1 Objective of the deliverable.......................................................................................... 6  
   1.2 Structure of the deliverable......................................................................................... 6  
2. Solution Overview.............................................................................................................. 7  
   2.1 Rules Design............................................................................................................... 8  
   2.2 Bridge to Task 2.2..................................................................................................... 9  
3. Implementation .................................................................................................................. 11  
   3.1 Connection to the Knowledge Management ............................................................... 12  
   3.2 Interfacing with the Multi Agent-based Cyber-Physical System................................. 13  
4. ZDM Management Strategies and Rules Demonstrator .................................................... 14  
   4.1 Overview of the demonstrator.................................................................................... 14  
   4.2 Multiagent System Interface...................................................................................... 15  
   4.3 Knowledge Management System Interface............................................................... 16  
5. Conclusion ......................................................................................................................... 18  
References............................................................................................................................. 19
List of Figures

Figure 1 – Overview of the integration approach for T1.3 ............................................................... 7
Figure 2 – Generic Rule Structure ................................................................................................. 8
Figure 3 – Rule Composition ......................................................................................................... 9
Figure 4 – Rule Structure integrated into the AutomationML model .................................................. 10
Figure 5 – Task 1.3’s overall implementation ............................................................................... 11
Figure 6 – Connection to the Knowledge Management ................................................................. 12
Figure 7 – Connection to the Cyber-Physical System ..................................................................... 13
Figure 8 – Motor testbed illustration ............................................................................................ 14
Figure 9 – Graphical representation of temperature and pitch of the motor (examples) .................. 14
Figure 10 - ADOxx Modelling View of the current/voltage rule .................................................... 15
Figure 11 – Knowledge Management Interfaces (Demonstrator) .................................................. 17

List of Tables

Table 1 – Main attributes to be included in the GOODMAN ........................................................... 9

Acronyms

CPS  Cyber Physical System
KM  Knowledge Management
ZDM  Zero Defect Manufacturing
DMN  Decision Model and Notation
MAS  Multi-Agent System
RA  Resource Agent
PA  Product Agent
PTA  Product Type Agent
IMA  Independent Meta Agent
API  Application Programming Interface
REST  Representational State Transfer
CAEX  Computer Aided Engineering Exchange
1. Introduction

1.1 Objective of the deliverable

This deliverable contains the outcome of Task 1.3, entitled “Development of ZDM Management Strategies and Rules”, which entails the design and development of both the rules and integration strategy to achieve the project’s Zero-Defect Manufacturing (ZDM) goals. The deliverable is a Demonstrator, so this document briefly describes the main concepts behind the demonstrator and illustrates its functionalities.

Through the methodology developed in Task 1.2 [1], GOOD MAN intends to introduce a fully integrated but at the same time decoupled ZDM solution, making usage of complex data analytics and Knowledge Management (KM) approaches (developed in Work Package 4), a Multi Agent-based Cyber-Physical System (MAS-CPS) and a common data exchange and representation format (designed and developed in Work Package 2) deployed and running at the shop-floor level.

Hence, Task 1.3 aims at implementing the aforementioned methodology through the design and development of an integration layer capable of linking and interoperating the KM, Data Analytics and the MAS-CPS. To do so, the integration layer uses a service-based approach where the higher-level modules can write (insert, update or remove) management rules to and from a rule repository, from which the agents can also consult the specific rules associated with their particular resource. To enable this, the rule server is able to receive the rules, translate them into the common data representation format established in Task 2.2, to posteriorly be consumed and used by the agents to control the system’s quality parameters and react accordingly. This can be done either by taking or suggesting appropriate corrective actions, or alerting operators for potential quality issues.

To this end, the document contemplates the design phase of the generic rules’ structure, as well as the implementation of the integration approach to bridge the gap between WP2 and WP4, culminating in the description of the demonstration performed during the project’s M12 meeting.

1.2 Structure of the deliverable

This document is structured as follows: Section 2 provides an overview of the general solution, specifying the rule design and the bridge to Task 2.2. Afterwards, Section 3 describes its implementation, focusing on the connections to both the Knowledge Management and the Multi Agent Cyber-Physical System, along with their respective services. Finally, Section 4 details the demonstration that resulted from the combined efforts of Task 1.3, WP2 and WP4, followed by some brief conclusions in Section 5.
2. Solution Overview

The developments of Task 1.3 aim at delivering an implemented solution of GOOD MAN’s ZDM methodology through the proper integration of all its key modules, allowing the correct flow of information through the different layers that constitute the GOOD MAN overall solution. To this extent, a service-based rule server was developed permitting the communication between the more complex, higher-level functionalities (KM and Data Analytics) and the Multi Agent-based CPS.

To do so, it is necessary to translate the decision rules resulting from the application of complex data analysis algorithms and the domain expert interaction by the KM, into simple, straightforward rules compliant with the common data format adopted by the CPS.

These rules can then be used by the CPS to adequately monitor and control the system’s quality parameters in runtime. This can be done for instance by modeling when to call or trigger maintenance actions, when to adapt production parameters as well as which parameters to tune, or even when to ask for new configurations for a given station.

As such, Task 1.3 defines the dataflow and provides the means for decision rules to be translated from the Decision Model and Notation (DMN) format, into an AutomationML [2] repository, and finally being made available to the agents to be interpreted in runtime. The overall integration approach can be seen in Figure 1.

![Figure 1 – Overview of the integration approach for T1.3](image)

As it can be observed, there are three key interactions involved in the proposed approach. More specifically, the KM layer can push new or updated rules to be relayed to the CPS, which can in turn consult existing rules, or request an update in case it finds itself with insufficient knowledge to process a given event.
Furthermore, this integration layer also serves as a way to decouple the CPS from the Knowledge Management specifications, allowing for a truly modular approach. This means that even if the KM layer is modified, only the interface to AutomationML needs to be adapted, making this process invisible to the CPS since its common data representation format remains the same, thus requiring no additional programming effort on that front.

2.1 Rules Design

The design of the rules’ structure was done in two phases. In an early stage, a draft structure of the rules was modeled in order to serve as the basis for the discussion with the remaining partners involved in the task. This generic structure is represented in Figure 2.

![Generic Rule Structure](image)

In essence, these structures define that rules can generically be described by a given condition that needs to be verified, and a resulting action to be carried out once the respective rule is triggered. For this, a condition can be decomposed into X, the parameter being evaluated, a comparison operator (e.g. equals, greater than...) and a threshold Y.

This type of rule structure has been the foundation of rule engines in several domains including process control and data stream processing [3]–[5]. These engines typically deal with condition/action pairs, usually expressed using the aforementioned “if-then” statements, analyzing inputs for any relevant conditions that might be met, taking action accordingly.

Based on this, two key points were raised during the discussions with the remaining partners. The first of which related to the fact that a rule can have dependencies, for instance Rule2 associated with Resource2 should only trigger if Rule1 associated with Resource1 has previously been triggered as well. The other concerns the issue of multiple conditions, meaning that a given resulting action might only be necessary if not one, but multiple conditions have been met. This case is illustrated in Figure 3.
This case is hereby referred to *Rule Composition*, as it can be likened to an AND operator, where for instance *Result1* is enacted if *Condition1 *∧ *Condition2 *∧ *Condition3* have been verified. Contrastingly, an OR operator would just translate into different rules being defined.

As previously stated, this format is intentionally simple and straightforward in order to allow the CPS to quickly reason during execution and take action as necessary, without being encumbered by additional degrees of complexity.

This level of relatively simpler expressiveness is only possible due to the fact that the complexity is dealt with at the KM layer, through its Complex Event Processing and Machine Learning models. These are capable of coping with the underlying trends, correlations and hidden patterns of the shop floor’s parameters, and can in turn compute the necessary thresholds and features that need to be controlled in runtime by the CPS, represented as the aforementioned condition/action pairs.

### 2.2 Bridge to Task 2.2

Considering that a critical part of the integration approach for T1.3 is the representation of the rules in AutomationML, it is also necessary to include the required descriptions in the GO0DMAN data model being developed in T2.2.

For this purpose, the main requirements in terms of attributes collected from the generic structure are represented in Table 1.

**Table 1 – Main attributes to be included in the GO0DMAN**

<table>
<thead>
<tr>
<th>Condition</th>
<th>Rule</th>
<th>Result</th>
</tr>
</thead>
<tbody>
<tr>
<td>ValueID</td>
<td>ID</td>
<td>Description</td>
</tr>
<tr>
<td>Operator</td>
<td>Description</td>
<td>Priority</td>
</tr>
<tr>
<td>Threshold</td>
<td>Collection&lt;Condition&gt;</td>
<td>Dependencies</td>
</tr>
<tr>
<td>Result</td>
<td></td>
<td>Type</td>
</tr>
</tbody>
</table>

With this, the respective classes were added into the GO0DMAN data model, as a joint effort between T1.3 and T2.2. The result can be seen in Figure 4.
This joint effort also included the implementation of the parsers to and from the AutomationML data model, which enable the addition of new rules into the rule store, or the update of existing rules, as well as enabling external entities (i.e. CPS) to poll existing rules.

The following section will further detail the particular aspects pertaining to the implementation of T1.3’s integration layer.
3. Implementation

The main development in Task 1.3 encompasses the rules’ design and a development of a GOOD MAN Rule Integration Module. This module is responsible to store not only the rules used by the agents to coordinate the execution, correctly verify the quality issues and if necessary trigger some recovery mechanism and strategy.

The GOOD MAN Rule Integration Module was designed and developed as a REpresentational State Transfer (REST) Web Server in order to facilitate and allow an easy and quick integration of this module with external modules, in this case, a DMN tool at the top level and a Multi-Agent System at the bottom, shown in Figure 5.

![Figure 5 – Task 1.3’s overall implementation](image)

The REST Web Server has been developed using Restlet framework (http://www.restlet.com). The developed services allow external tools and software to consult/edit the AutomationML file where
the system’s topology and system’s characteristics are stored, more specifically the Zero Defect oriented rules.

Hence, whenever a service is externally called, an Application Programming Interface (API) developed to manage the AutomationML is used to process that call, writing and/or consulting the AutomationML file. The developed API works as a parser capable to translate the received Java objects or DMN files into the previously presented data structure or vice-versa. With this mechanism, the services are now able to transform a DMN file and store the information into the AutomationML, regarding the other way around, the services consult the AutomationML and translate the data stored in the file to the respective Java objects.

### 3.1 Connection to the Knowledge Management

The higher-level tool (Knowledge Management, WP4) is responsible to manage the DMN rules that will be used to define the system’s behavior. Hence whenever a rules’ change is required, the design is performed on Knowledge Management Level using BOC’s ADOxx implementation. The implementation is integrated in the overall knowledge management approach as discussed in D4.1 and support the design, verification and validation of quality management rules in the context of the overall system. The updated and released rules are then exported to the services defined below, following a passive/push strategy. One of the three designed and implemented services is triggered:

- **insertRule(resourceID, dmnFile):** This service is responsible to add a new rule into the GOOD MAN Rule Integration Module. To insert a new rule, it is necessary to send the DMN file contending the rule and the unique ID that describes the associated resource;
- **updateRule(resourceID, dmnFile):** Similarly to the previous service, to update a rule it is necessary to send the resource’s ID and the rule to update;
- **removeRule(resourceID, dmnFile):** Similarly to the previous services, to remove a rule it is necessary to send the resource’s ID and the rule to remove;

In Figure 6 the workflow whenever a knowledge management tool calls to insert, update or remove a rule is shown.

![Figure 6 – Connection to the Knowledge Management](image)

Whenever a rule needs to be changed, the Knowledge Management tool must call one of the previously specified services. Inside the services, the Rule Integration Parser is called and the received DMN file is translated to the AutomationML structure, Computer Aided Engineering Exchange (CAEX). Now the rule is ready to be processed and stored, updated or removed. At the end of this process, a service must respond with an acknowledgment that confirms the action.
An important consideration in the architecture of this integrated system relates to the update trigger for new rule definition. These triggers are understood as reported observations to improve quality-related issues in the system. The knowledge management system in GOOD MAN foresees two interaction patterns: a) human trigger, where based on the analysis and interpretation of an expert (operator, line manager, quality responsible, production process engineer) a manual intervention is performed; and b) data-driven trigger, where based on the results of the data analytics environment developed in WP4, the update process is started. The interventions are executed following a systematic approach that builds on the data analytics results, but involves strongly the experts from different fields in the design and analysis process. The outcome of this process is an update of the rule-set or change in the configuration of the overall system.

3.2 Interfacing with the Multi Agent-based Cyber-Physical System

At the bottom, the CPS is implemented through a Multi-Agent System (MAS). This ecosystem must be able to consult the rules associated to each agent/resource in order to know how to behave regarding the ZDM strategies. Hence whenever an agent wants to consult the associated rules or a specific rule it can use one of the two available services:

- consultByResourceId(resourceID): Sending the unique ID of the resource, the agent must receive a list of all the rules associated to the resource;
- consultByRuleId(ruleID): With this service, the agent can consult all the information associated to the rule with the rule ID sent.

In Figure 7 the workflow whenever an agent calls one of the previously presented services is shown.

![Figure 7 - Connection to the Cyber-Physical System](image)

Whenever an agent calls one of the two services, the Rule Integration Parser firstly queries the AutomationML Storage in order to retrieve the desired information (one rule or a list of rules). The extracted information, structured according to CAEX is now parsed to a Java object or a list of Java objects and posteriorly sent to the agent.

Additionally, an update of the rules can also be pushed directly from the KM layer, in which the rule server can also act as a client for a service provided by the CPS itself. In this case, the Independent Meta Agent (further explained in Subsection 4.2) acts the entry point for this push mechanism, being then responsible for propagating the rules accordingly to the agents abstracting their respective resources.
4. **ZDM Management Strategies and Rules Demonstrator**

In order to be able to show the ZDM Management Strategies and Rules, a simplified real application case has been used. An electrical motor testbed has been developed for testing purposes of the GOOD MAN overall system approach, i.e. considering the MAS, the global data analytics, the knowledge generation and with the rule server integration.

4.1 **Overview of the demonstrator**

The testbed, depicted in Figure 8, is composed by an electrical motor, a control system and a data measurement instrumentation system. In particular, the following tests are performed:

- Temperature test;
- Voltage test;
- Vibration test;
- Current test.

In this case, the simple testbed aims at simulating a multi-stage stage production line of electrical motors, where different measurements are performed and specific controls are implemented.

![Figure 8 – Motor testbed illustration](image)

Two examples of the type of data that the system is able to collect are depicted in Figure 9. Here, one can find the representation of the pitch value associated to the motor vibration and (at the lower graph) the motor's frame temperature.

![Figure 9 – Graphical representation of temperature and pitch of the motor (examples)](image)
Briefly, the system control is performed by the publication of Message Queuing Telemetry Transport (MQTT) topics that triggers the execution of different types of tests (i.e. electrical, mechanical or temperature). At the end of each one of the tests, the system publishes an end topic to inform the associated agent of the testing conclusion. During the testing phase, the system continuously publishes the designated testing data.

In order to enable the CPS to perform the runtime quality control of these parameters, a rule was defined through the composition of maximum current and minimum voltage threshold values. This rule also enables the testing and validation of the integration between the rule server from Task 1.3 and the CPS. Its DMN table representation can be seen as defined in the ADOxx Modelling Environment depicted in Figure 10.

![Figure 10 - ADOxx Modelling View of the current/voltage rule](image)

The rule server developed in Task 1.3 is responsible for the translation of this DMN rule to the common AutomationML format used by the agents. After this translation is performed, the entire decision table for a given resource is then represented in the AML format, being provided to its respective agent through a REST webservice as further explained in Section 4.2.

With this, as the CPS collects new current and voltage measurements, it can check whether or not these are within the boundaries specified by the quality management rule. In this particular case, if the conditions are triggered, the system identifies the issue as the motor being in an overcurrent state and notifies the operator that corrective action is required.

### 4.2 Multiagent System Interface

The testbed simulates the testing phase of an electrical motor, where 3 Resource Agent (RA) are put into place to manage, collect and analyze the data from: electrical tests, mechanical tests and temperature tests. One Product Type Agent (PTA) represents the products catalogue and the necessary Product Agents (PAs) are launched as needed. Finally, one Independent Meta Agent (IMA) is put into place to act as the entry point for the rule repository and global data storage.

The management phase is handled by publishing a start topic where the payload is the desired test identifier. An example of such action, from the agent side is depicted below. This is triggered by request of the motor PA that requests the necessary test.
Deliverable 1.3 ZDM Management Rules

Currently, the RA is processing the data in a stream approach by passing, for each received value, the values in the received rules from the rule server repository. To this extent, it is important to mention the current rule process management from the GOOD MAN MAS point of view.

The rule server service is managed at the IMA level, which is responsible for the acquisition of the agent rules for the whole MAS. Currently, the IMA is following a pooling approach, verifying at a given interval rate the availability of new rules.

To achieve this, the IMA is, for each agent, using the available services, checking if there are rules for them and, in case of a positive reply, sending to the designated agent (see the following code excerpt).

```java
private void handleProcessingStart(ACLMessage msg2) {
    LAST_PA_ID = msg2.getSender().getLocalName();
    eu.h2020.go0dman.support.BasicMethods.log()
        .info(myAgent.getLocalName() + " <= received a START_PROCESSING from " + LAST_PA_ID);
    eu.h2020.go0dman.support.MqttSupport.publishStartTopic(myAgent, LAST_PA_ID);
}
```

At the receiving agent side, the rule is parsed and implemented for data validation. A simplified example of such rule parsing and implementation is given in the following code excerpt.

```java
for (int i = 0; i < RA_AGENTS.length; i++) {
    try {
        Collection<GMManagementRule> ruleList =
            eu.h2020.go0dman.agent.IMA.RuleManagement.ruleListGet(RA_AGENTS[i].getName().getLocalName().toString());
        if (!(ruleList == null)) {
            for (GMManagementRule rule : ruleList) {
                sendRuleToAgent(RA_AGENTS[i].getName().getLocalName(), rule);
            }
        }
    } catch (Exception e) {
        // TODO Auto-generated catch block
        e.printStackTrace();
    }
}
```

At the receiving agent side, the rule is parsed and implemented for data validation. A simplified example of such rule parsing and implementation is given in the following code excerpt.

```java
if (!(eu.h2020.go0dman.support.BasicMethods.ruleCondition == null)) {
    eu.h2020.go0dman.support.BasicMethods
        .ruleVerification(eu.h2020.go0dman.support.BasicMethods.ruleCondition);
}
```

### 4.3 Knowledge Management System Interface

As the improvement loop for the demonstrator, the components of WP4 are used to a) monitor the quality check behavior and provide reports and dashboards, b) enable the knowledge-based improvement cycle and rule definition. In the initial version of this demonstrator, the focus was on
the integration and validation of the update mechanism defined in b), further extended to monitoring and quality KPI level at a later stage.

Figure 11 shows the integration of the rule-modelling perspective in the GO0D MAN Modelling Toolkit currently under development in Task 4.2.

Further adaptation and modifications to the demonstrator are planned in accordance with the development results in T4.2 and T4.3. These steps are outlined below:

a) Data Interpretation Support: based on the availability of data coming from the MAS system via the data analytics interface, quality dashboards and reports are established to enable a close-to-real-time monitoring of semantically rich data and support the analysis.

b) Integration of quality management rule validation and continuous improvement cycle: the current setup for the demonstrator support the analysis by an expert. The management method and approach as outlined in D1.2 is revisited to provide tool functionality in the knowledge management implementation for specific steps and tasks to be performed in iteration cycles.
5. Conclusion

In Task 1.3 a rule-based solution was developed in order to implement the ZDM methodology previously defined in Task 1.2. This was enacted by bridging the gap between GOOD MAN’s MAS-based CPS and the KM layer, as a combined solution to achieve the projects ZDM goals.

For this purpose, a rule server was implemented as the integration layer, aligned with the common AutomationML-based data exchange format resulting from the developments of T2.2. The compliance with the common format for data exchange ensures that the solution remains truly modular and decoupled, making the CPS independent from the internal data representations of the KM solutions.

This server provides the KM layer with the necessary mechanisms to push new or updated rules resulting from WP4’s data analysis and management tools into the CPS, while also giving the latter the means to consult existing rules when deemed necessary by its reasoning behaviors.

The objectives for Task 1.3 were achieved within the expected time-frame, culminating in a live demonstration during the project’s M12 meeting at the IPB facilities in Bragança, Portugal. In this demonstration, the rule server was instantiated and allowed the CPS to consult and interpret rules coming from a DMN file, as outputted by the KM layer.
References


