Enabling the business-based
Internet of Things and Services

(FP7 257852)

D6.1 Design of business rules representation and execution

Published by the ebbits Consortium

Dissemination Level: Public

European Commission
Information Society and Media

SEVENTH FRAMEWORK PROGRAMME

Project co-funded by the European Commission within the 7th Framework Programme
Objective ICT-2009.1.3: Internet of Things and Enterprise environments

Document control page

Document file: D6.1 Design of business rules representation and execution_1.0.doc
Document version: 1.0
Document owner: Matts Ahlsén (CNET)

Work package: WP6 – Mainstream business systems
Task: T6.6 – Design of business rules representation and execution
Deliverable type: R

Document status:
x approved by the document owner for internal review
x approved for submission to the EC

Document history:

Version	Author(s)	Date	Summary of changes made
0.1	Matts Ahlsén (CNET)	2011-04-08	Deliverable outline, ToC
0.2	Matts Ahlsén (CNET)	2011-04-29	Revision of partner contributions
0.3	Matts Ahlsén (CNET)	2011-05-16	Revision of partner contributions
0.5	Matts Ahlsén (CNET)	2011-05-25	Sections 3,4 & 6: Terminologies, ISA-95/IEC 62264, object models, architecture
0.6	Jesper Thestrup (In-Jet),	2011-05-30	Motivations for modelling approach, process models, sections 3 and 4
0.7	Yves Martin (SAP AG)	2011-06-03	Section 4 on ontological support, Sections 5 and 6 on rule repositories and execution with Drools
0.8	Ferry Pramudianto (FIT)	2011-06-10	Candidate rule languages in section 5
0.9	Karol Furdik (IS)	2011-06-12	Candidate rule languages in section 5
0.91	Karol Furdik (IS)	2011-06-20	Candidate rule languages in section 5
0.92	Matts Ahlsén (CNET)	2011-06-21	Updates for internal review
0.95	Matts Ahlsén (CNET)	2011-06-22	Version for internal review
0.96	Matts Ahlsén (CNET)	2011-06-27	Update according to review comments
0.97	Yves Martin (SAP AG	2011-06-28	Update according to review comments
1.0	Matts Ahlsén (CNET)	2011-06-30	Final version form submission to the European Commission

Internal review history:

Reviewed by	Date	Summary of comments
Jan Hreno (TUK)	2011-06-24	Approved with comments
Michael Jacobsen (TNM)	2011-06-26	Approved with comments

Index:

List of figures:

1. Executive summary

Business rules are important building blocks in the ebbits architecture, they are intended to provide a logical foundation for automated and flexible response to physical world events with dynamic execution of related business activity thru services.

This report investigates approaches to the design and representation of business rules for application in ebbits. We start by characterizing the overall approach to modelling in ebbits, relating the work in WP6 to other work packages, such as WP3 and WP4.

Well-defined terminology and object models are important in a rules framework, as a basis for interoperability. To this end, the ISA-95 standard for business and manufacturing systems interoperability, was proposed as a starting point and a knowledge source in ebbits. We make an overview of the two first parts of the ISA-framework, describing the approach taken to terminology and object models.

We then address the representation of business rules, by reviewing a number of different rule languages and rule design frameworks and tools. The Drools rules framework has been chosen as the initial technology for rules design and execution.

Finally we look at the execution of business rules and on the relationship to the foreseen run-time environment. We also define the run-time environment for the execution of business rules employing both event management and service orchestration.

The ultimate contribution of this work in the course of the project, lies in the architecture providing a seamless integration of rules, events and services.

2. Introduction

The research and development work undertaken in WP4-8 aims to develop and build the components for an ebbits prototype platform for lifecycle management. The development is based on the set of requirements derived from technical scenarios and use cases developed in WP2.

WP4 develops ontologies and semantic components to be used by the Ontology Manager in the Service Layer and WP5 develops components for distribution of support for production optimisation, which is the central part of the Application Layer functionality. WP7 develops event and Data Management structures for the Data Management Layer and the service execution subsystem whereas WP8 develops network and communication infrastructure and semantic integration of physical world objects into the platform as part of the Network Management Layer.

But it is extremely important that the ebbits platform is not just a technical platform for data collection and fusion. It must be aligned with the real world enterprise and business processes that have been established throughout the entire business ecosystem.

In order to secure the alignment with business ecosystems, work in WP3 and WP6 will be undertaken in parallel with the development work. The overall aim of the work is to facilitate the deployment of the ebbits platform in real world business ecosystems.

The work in WP3 focus on describing business processes and business logic starting with specific use cases and moving into describing existing and future business applications in the two domains. Moreover, the economic aspects in terms of business models and the general business conditions in the selected user domains will also be investigated.

WP6 develops the knowledge base for integration of physical world objects with business management systems as part of the Application Layer. Investigation and description of enterprise and business services will provide knowledge of interaction of business processes and business logic. An important aspect of this work involves the investigation of an approach to business rules design, representation and execution. The integration of distributed intelligence and centralised enterprise information systems will also be investigated. Components will be developed for interoperability of enterprise systems and ebbits distributed intelligence and business rules execution in the Service Orchestration Layer.

The resulting ebbits components will allow the ebbits platform to be easily and seamlessly integrated with legacy business management systems. The technical integration will be performed in WP9 and sets of end-to-end business applications will be developed in WP10 to demonstrate the ebbits functionality in near real-life demonstrations.

3. Modelling framework

3.1 Motivations for enterprise and business modelling in ebbits

This chapter motivates the enterprise-centered work undertaken in parallel with the technological development work. The overall aim of the work is to secure the alignment with legacy business ecosystems and facilitate the deployment of the ebbits platform in real world business ecosystems. This chapter helps the reader to understand the consistency in the overall conceptual business-technical framework performed in WP3, WP6 and WP10.

3.1.1 Describing the state of play in business systems and defining business logic processes

In WP3 we focused on the business environment and the digital ecosystems of the two user domains. The focus is on the current processes and the aim is to understand the state of play from an enterprise perspective and to define the proper semantic business logic processes.

Accurate identification of current standard processes is a crucial factor for establishment of new, innovative procedures and processes based on the ebbits platform. Current status within enterprises was described using use cases from both operational business processes as well as from a managerial point of view.

Current, standard processes in the manufacturing and agriculture domains were identified and described in form of use cases in deliverable D3.1 Enterprise use cases. The use cases provide an easy way to comprehend the functional requirements and to identify the various interactions between the users and the systems within a given business environment.

The use cases are divided into general use cases (for all types of businesses) and domain specific use cases (for ebbits domains). The general use cases comprise typical business processes such as ordering goods or services and logging of data for accounting. The specific use cases are for example an energy reduction management process in manufacturing and the veterinarian monitoring animals at the farm and in the slaughterhouse in agriculture.

Implementation of semantic interoperability requires a well defined, generic business vocabulary. Inspired by the use cases we thus move on to create vocabularies across vertical and horizontal business application domains. The vocabularies will be used to define a semantic ontology framework, which can be used in a wide range of application domains.

For the proper orchestration and execution of business services, a semantic business decision model approach will be adopted. The business decision model is a representation of a specific situation of events and related services and it combines strategic business priorities and goals with real-time information about the specific situation. A precise definition of roles and decision making processes and procedures, together with the vocabularies, will allow us to define the semantic business logic and decisions process models (D3.3 Business Logic Models).

The generalised business process and logic models will be applied back to the specific user domains in ebbits thus defining the D3.4 Business framework for applications for production and energy optimisation and the D3.5 Business framework for online food traceability in lifecycle perspective. The objective of this work is to develop the usage scenario, the business framework and the metrics for the ebbits prototype applications.

Finally, the use cases, the process descriptions, and the vocabularies will be significant input to the development of new economic business models to understand value creation in the new environment. Whereas the *business process models* describe the *logic processes*, the work on *business models* will focus on the *economic values*. It thus aims to develop a viable and sustainable economic business framework for the ebbits platform for enterprise systems.

3.1.2 Investigating the dynamism and interoperability of enterprise systems

In WP6 we zoom in on the dynamism and the interoperability of mainstream business systems. We develop the knowledge base for integration of physical world objects with business management systems and develop components that define how data and information will be propagated through the ebbits platform. We will further investigate how business logic executed by ebbits applications will interact in real-time and on demand with enterprise system and business services.

Business rules execution in the ebbits architecture will, in its full implementation, need to react ubiquitously to every change in the environment according to prevalent business logic and pre-established rules. The key requirements for the business rules execution is thus that the ebbits platform is able to recognise and respond to physical world events and interact on demand with legacy enterprise system and business services to achieve overall strategic business goals.

The Event Management subset will consult business rules and business logic ontologies in order to provide the instructions for the Service Orchestration subset to perform correctly.

A response will be initiated by the occurrence and detection of an event. When the event has been detected, the Event Management subset will evaluate the event and its context and seek the corresponding deterministic action. The deterministic action in this context means that the action has been identified and described as a series of business rules and business logic, all created as executable services, which can be orchestrated in the Service Orchestration subset.

To develop the interoperability aspects, we need first to identify and describe a generic taxonomy and representation of business rules and business logic and their interaction across enterprise systems. We will then map this representation to a design of the Service Orchestration subset and to a taxonomy. Using a formalistic abstraction of the business rules representation allows us to leverage acknowledged standards such as ISA-95 to ebbits Service Orchestration.

The ISA-95.01 Enterprise control system models standard is a candidate for defining the hierarchical business roles logic, which will be implemented in the ebbits Service Orchestration subsystem. Different models focus on a specific aspect of the integration requirements. Functional models can be used to define which functions are executed within the enterprise and which information flows from one function to another. The resource object models form the basis for a consistent set of complex object models that make logical groups of the information that has to be exchanged.

The business rules execution will be designed so as to use conceptual models as the strategic decision level: What are the strategies to be used to achieve a certain goal? At the process layer, a comprehensive set of different kinds of operational rules related to specific operative business decision determine what is to be executed. Next, the operational decision models are designed using business logic, which describes the functional algorithms that handle and prioritise the exchange of information and interaction between services. And finally, at the lowest level, the business logic is interacting with sensors, actuators and subsystems, which represent physical world events that are being captured, processed and fused to centralised and decentralised services.

The first task is to establish the knowledge base and design the business rules representation and execution. This result of this work will be described in the present deliverable.

Future work we aim to enhance the mechanisms that ensure that both information and meta-data propagate through the distributed setup and can be processed at selectable points to ensure appropriate integration and interoperability while, at the same time, keeping alternatives for flexibility and degree (close, loose) of coupling as well as allocation of functionality.

Further, we will develop flexible and powerful mechanisms that provide matching between different information representations and enable information mappings will be performed, in order to ensure interoperability. Enhancement is also needed in terms of powerful query answering mechanisms in order to cope with the different constraints when dealing with large amounts of information.

Finally, we also need to investigation and enhance matching services between platform and enterprise systems. It is crucial to have a service description at hand which provides all details necessary in order to supply the right service to the service request. Bringing context into service matchmaking helps to facilitate searches and make the results more relevant.

The result of WP6 is not only a knowledge base for integration of physical world objects with business management systems and interaction of business processes and business logic. Components will also be developed for interoperability with centralised enterprise systems and ebbits distributed business rules execution. The resulting ebbits components will allow the ebbits platform to be easily and seamlessly integrated with legacy business management systems.

3.2 Process models

Process modelling is a comprehensive term used in process engineering. It refers to procedures of the same nature that are classified together into a model. In this sense a process model is a description of a process at the type level; a (real) process so becomes an instantiation of it. The same process model is used to describe the structure and development of many applications and thus, has many instantiations.

One possible use of a process model is to prescribe how things must/should/could be done in contrast to the process itself which is really what happens. A process model is roughly an anticipation of what the process will look like. What the process shall be will be determined during actual system development.

An organisation is a complex system integrating processes, functions, resources, stakeholders etc. In order to understand how it is structured and how its components are interrelated, a manageable model of reality is necessary. An enterprise can be analysed and integrated through its business processes. Business process modelling (also indicated as BPM) is the activity of representing both the current ("as is") and future ("to be") processes of an enterprise, so that the current process may be analysed and improved. Business process improvement covers both business process redesign and re-engineering. Typically this is performed by business analysts and managers who are seeking to improve process efficiency and quality.

In general terms a process of an enterprise could be defined as the course (activities and passages) between every input and every output. There are several definitions and classifications to business processes, with similarities:

- all the activities that take one or more inputs, transforms and adds value to them, and provides one or more outputs;
- the combination of a set of activities within an enterprise with a structure describing their logical order and dependence, and whose objective is to produce a desired result;
- a set of one or more linked procedures or activities which collectively realise a business objective or policy goal, normally within the context of an organisational structure defining functional roles and relationships.

Some authors propose a classification of business processes into different categories, again with different approaches. A first division is between "customer processes" which results in a product or service that is received by an organisation's external customer, "administrative processes' that are invisible to external customers but essential for effective business management and "management processes" are actions that managers are required to take to support enterprise business. Another possibility is the classification in strategic, operational and enabling processes; the first is comparable to the management process, operational process is related to the production workplace, while the last one is similar to the strategic process but more interested on the area of possible interventions for enabling new opportunities. In this second version there are similarities to the Michael Porter's value chain framework, which classifies organisational activities into primary and secondary. Other authors argue that most definitions are too oriented towards traditional business and fail to express the nature of processes that need to develop and adapt to challenging environment, especially considering new paradigms such as e-Business.

The emphasis on business process change (reengineering) has gone through a number of phases in the last years. First, there was the Total Quality Management that refers to programs and initiatives to emphasise incremental improvement in work processes and outputs over an open-ended period of time. In the early 1990s business process reengineering has become one of the most popular topics in organisational management, creating new ways of doing business. Since improving the business performance was not achieved by automating existing business activities, many leading organisations have conducted business process change in order to gain competitive advantage. At the beginning the focus was on internal business processes radical change, in particular an integration between total quality management and business process reengineering.

A second stage emerged when the Internet phenomenon took off and provided an ICT internetworked infrastructure that enabled electronic business and new forms of Web-based business processes. To meet customer demand, companies depend on close cooperation with customers and suppliers. Business process change driven by e-Business could not be based only on radical redesign of intra-organisational processes, but should be extended to the entire business network (internal and external). An online partnership must extend far beyond presenting promotional and pre-sales activities on companies' Web sites. It has to drill deep into a company's processes in order to create totally different business models. Therefore, most companies need to re-evaluate core processes to strengthen customer service operations, streamline supply chains and reach new customers. Traditional companies are forced to change their current business models and create new ones. The use of the Web and supply chain management has opened up the opportunities for exchanging information and managing knowledge around the new processes.

3.2.1 Classification of process models

Process models are usually classified against four different criteria.

Classification by coverage

There are four types of coverage where the term process model has been defined differently:

- Activity-oriented: related set of activities conducted for the specific purpose of product definition; a set of partially ordered steps intended to reach a goal.
- Product-oriented: series of activities that cause successive product transformations to reach the desired product.
- Decision-oriented: set of related decisions conducted for the specific purpose of product definition.
- Context-oriented: sequence of contexts causing successive product transformations under the influence of a decision taken in a context.

Classification by alignment

Processes can be of different kinds, so depending on their characteristics they can be modelled in various ways:

Strategic processes:

- investigate alternative ways of doing a thing and eventually produce a plan for doing it,
- are often creative and require human co-operation; thus, alternative generation and selection from an alternative are very critical activities;

Tactical processes:

- help in the achievement of a plan,
- are more concerned with the tactics to be adopted for actual plan achievement than with the development of a plan of achievement;

Implementation processes:

- are the lowest level processes,
- are directly concerned with the details of the what and how of plan implementation.

Classification by granularity

Granularity refers to the detail level of the process model and affects the kind of guidance, explanation and trace that can be provided. High granularity limits these to a rather coarse level of detail, whereas fine granularity provides more detailed capability. The nature of granularity needed is dependent on the situation at hand.

Project manager, customer representatives, management require rather large-grained process description as they want to gain an overview over time, budget, and resource planning for their decisions. In contrast, software engineers, users, testers, analysts, or software system architects will prefer a fine-grained process model for the details of the model deliver them with instructions and important execution dependencies such as the dependencies between people.

While notations for fine-grained models exist, most traditional process models are large-grained descriptions. Process models should, ideally, provide a wide range of granularity.

Classification by flexibility

It was found that while process models were prescriptive, in actual practice departures from the prescription can occur. Thus, frameworks for adopting methods evolved so that systems development methods match specific organisational situations and thereby improve their usefulness. The development of such frameworks is also called Situational Method Engineering.

Method construction approaches can be organised in a spectrum ranging from 'low' to 'high' flexibility. Lying at the 'low' end of this spectrum are rigid methods, whereas at the 'high' end there are modular method construction. Rigid methods are completely pre-defined and leave little scope for adapting them to the situation at hand. On the other hand, modular methods can be modified and augmented to fit a given situation. Selecting a rigid approach allows each project to choose its method from a panel of rigid, pre-defined methods, whereas selecting a path within a method consists of choosing the appropriate path for the situation at hand. Finally, selecting and tuning a method allows each project to select methods from different approaches and tune them to the project's needs.

3.2.2 Business process mapping

The two phrases ‘process mapping’ and ‘process modelling’ are often used interchangeably. It has its origins in industrial engineering and as a management tool initially was developed by General Electric (Perumpalath, 2005). In the literature different definitions are found:

- identifying, documenting, analysing and developing an improved process, or
- enabling a common understanding and analysis of a business process, or also
- a representation of the company’s operation or a specific part of the operation, usually in the form of a graphical depiction of the structure and activities of the operation. The model often shows the relationships between work steps and their sequence portraying workflow.

This is considered an excellent process management tool that mainly aims at reducing waste (due to overproduction, defects, unnecessary inventory), and identifies, eliminates irrationalities and inconsistencies in processes. There are five key stages to process mapping:

- study of current flow of processes
- identification of sources of waste
- consider whether the sequence of activities can be rearranged and made efficient
- optimisation of flow layout – e.g. reducing distances between stages
- removal of unnecessary activities.

The idea is to develop “As-Is” (current state) and “To-be” (ideal) process maps to assist the process. Some advantages and disadvantages are that the maps give a clearer explanation of a process than words; by working on maps, better understanding of the tasks and problems that face the organisation can be gained; however, process maps can prove to be too distracting and can slow down actions to ensure the integrity of the maps; they can also take on a life of their own and lose relevance to those working on the process; the maps cannot be relied on always as a good means of communication between layers of management.

There are a number of approaches to process mapping. They can be divided into three general categories: flow diagramming tools, case tools and simulation tools.

Flow diagramming tools	**Programs for the visualisation and management of flow diagrams.**
Case tools	Applications for the edition and comprehension of scenarios and use cases.
Simulation tools	Applications framework for the simulation of large and complex business systems.

Table 1 Classification of process mapping categories

One possibility is to review and classify modelling techniques against business processes perspectives (or the 'nature' of the business process), another option is to use a similar modelling methodology, but using different measure parameters, 'purpose of the model' and 'change model permissiveness'. The first dimension is further divided into the four categories: descriptive models for learning, descriptive and analytical models for decision support to process development and design, accomplished or analytical models for decision support during process execution and control and enactment support models to IT. Some process models are 'active' as they allow users to make changes, or are dynamic themselves, whereas other models are 'passive' as the user could not change them without remodelling the process. This characteristic between active and passive models is referred to the 'change model permissiveness'.

3.2.3 Tools, techniques and methodologies

Business process modelling projects can have different goals and similarly those creating the models could use different methods and tools (Table 2).

Focus of method / tool	Example
Strategic planning	Balanced Scorecard - BSC, Benchmarking
Accounting techniques	Activity Based Costing Analysis - ABC, Return on Investment - ROI
Continuous improvement	Total Quality Management - TQM, ISO Standard
Static process modelling or functional decomposition modelling	Data Flow Diagrams - DFD, IDEF - Integrated Definition (IDEF0)
Action coordination modelling	Action Workflow modelling method, IDEF3
Dynamic process modelling (simulation)	Petri Nets

Table 2 Process models methods and focus

Methods and tools generally do not adhere to one particular business process modelling standard, but it must be pointed out that most modelling techniques used in business today have been developed for industrial engineering, software engineering or information systems modelling environment.

Over the last three decades, a well-established procedure for modelling information systems was based on two complementary aspects of analysis: data modelling (entity-relationship modelling) and function modelling (data-flow diagramming). Since events which trigger a response in an information system come from within the organisation or from the external environment (an input), a business process view is seen as a third representational framework.

Several business process modelling techniques have been reviewed and classified. In some cases the purpose was to develop an enterprise information system. According to the findings, the choice of suitable technique depends on the project requirements, as well as the implementation scenario. As the scenario changes, the technique should also be changed correspondingly. The importance of a language that is understandable and unambiguous by both the end user and the developer also emerged.

Other categorisation of business process modelling techniques were developed where the modelling techniques are classified by the purpose that they would have when used in business process modelling projects. According to this classification, modelling techniques could have informational (data), organisational (where, who), behavioural (when, how) and functional (what) focus, and can be used to fulfil different objectives: understanding and communicating, process improvement, process management, process development and process execution. A single process modelling technique that covers all aspects of process modelling, especially the aspect of process dynamics, does not exist. One approach for solving this last aspect is to extend functional decomposition methods with event triggers in order to introduce task interdependence into the model. Another is to extend action coordination methods with added workflow structure through Petri net activity representation. The third approach is to develop new process modelling methods that are focused on process flow and process dynamics, such as IDEF3 and Activity Decision Flow diagrams.

3.2.4 Simulation

Business process change involves changes in people, processes and technology. As these changes happen over time, simulation appears to be a suitable process modelling method. Simulations could be useful because they allow playing with a model of part of the real world, without having to play with the real thing. The real world model is developed using computer software and the model can be used for experimentation.

The simulation modelling shows the process as a whole, drawbacks of the existing process, bottlenecks in the process execution and provides critical insight into process execution. Simulation can vary depending on the process characteristics (discrete, continuous, stochastic, deterministic, critical event, time-scale, machine simulation, human-machine simulation). The results of the simulation modelling represent a good foundation for a business process reengineering, also as a next step towards e-Business introduction. In case of redesigned or reengineered processes the effects can be simulated before implementation. This improves the efficiency of the practice, thus reducing expenditure.

Simulations over time help to judge whether operational level changes can lead to achievement of strategic targets over time and can reduce risk aversion to implementation. However, simulations require careful planning in order to obtain results within cost and time targets.

The list of the available business process modelling tools supporting simulation is as long as over 50 names (Indihar-Stemberger, 2003). Simulation is sometimes called a technique of 'last resort' because it is used when the system to be modelled is too complex for analytical models. The interaction of people with processes and technology results in an infinite number of possible scenarios and outcomes that are not possible to predict and evaluate using widely popular static process modelling methods. The introduction of simulation into process modelling is most often applied in particular conditions:

- Dynamic – process behaviour varies over time
- Interactive – processes consist of a number of components which interact with one another
- Complicated – the process consists of many interacting and dynamic objects

The main advantage of simulation modelling is in its integration of following functions: analysis and assessment of business processes, either in quantitative or qualitative terms; development of "to-be" models in order to examine "what-if" scenarios and export them to implementation platforms, such as workflow management and enterprise resource planning systems.

Modern simulation software tools are able to model dynamics of the processes and show it visually, which then can enhance generating the creative ideas on how to redesign the existing business processes. Such tools include graphic user interface (GUI) that enables process animation and graphical display of simulation results. Several authors have reported the application of simulation for business process redesign. Despite the numerous advantages of simulation software, some user requirements are still not adequately met such ase lack of links with other packages (software compatibility) and lack of interfaces for data input.

3.3 ISA-95/IEC 62264 standard

ISA-95 specifies a standard for the interfaces between control systems on the manufacturing side and enterprise systems on the supplier side. In that sense, it specifies models, operations and processes which foster the integration of different manufacturing execution systems (MES), responsible for production, maintenance and quality management, and arbitrary enterprise resource planning systems (ERP), covering sales, finance and logistics, across all industries.

The ISA-95 standard was initially developed in an international working group of the American National Standards Institute (ANSI) and the Instrumentation, Systems and Automation Society (ISA). Starting from 2003, the different parts of the ISA-95 have been adopted as international standards through the IEC/ISO as IEC 62264 – Enterprise-control system integration[1](IEC/ISO 2003; IEC/ISO 2004).

The set of models described reaches from models of the physical structure of enterprises and production procedures to models of information flows and object models. Furthermore, the standard supports various sorts of processes, be it batch, continuous or repetitive processing.

All in all, ISA-95 provides a shared conceptualization for describing data exchange as well as workflows and processes in plant systems and therefore facilitates inter and intra-company system integration. The main benefits ISA-95 promises are:

- Reduce cost by using ISA-95 as a well described and structured method to define the interface between enterprise and production control systems.
- Reduce risk and avoid errors by using the standard as a catalogue of best practices how to successfully integrate systems, collected by a huge group of companies.
- Improve Communication by using a commonly shared vocabulary and methodology to describe functions, activities and departments within the enterprise. This facilitates company internal as well as external communication.

3.3.1 Structure

The ISA-95 standard consists of five parts which are introduced in the following. Every part covers one particular aspect of the enterprise-control systems integration. The information is mainly structured in UML models, which are the basis for the development of standard interfaces between ERP and MES systems. For a more in-depth description of the various parts the reader is referred to the respective documents distributed by the ISO/IEC and the ISA-95 organization[2].

ISA-95.01: Models and Terminology

The primary objective of this part of the standard is to describe the manufacturing domain and to specify what information should be exposed to and exchanged with ERP systems and what terminology is used to describe those two domains. Therefore, it provides various models to represent the organization of the company as well as of physical assets involved in manufacturing. Furthermore, it offers models to define the function associated with the interfaces between control functions and enterprise functions. A basic building block of this part is the functional hierarchy model describing the whole integration domain. It consists of 4 layers with corresponding scopes and time frames. The model is depicted below.

[1] In this document we use ISA-95 and IEC 62264 synonymously.
[2] http://www.isa.org, http://www.isa-95.com

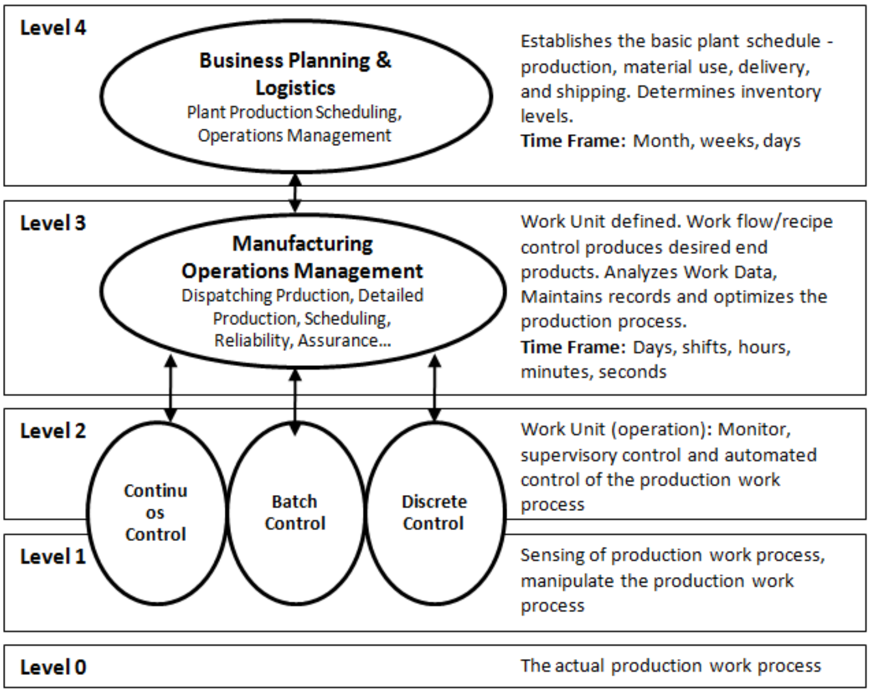


Figure 1: Hierarchical model of the control and enterprise system integration domain

ISA-95.01 and ISA-95.02 mainly address the interfaces between level 3 and level 4. The models defined in the context of this layer are: 1) hierarchy models to describe levels of functions and domains. The taxonomies specified here include terms like *enterprise*, *site*, *area*, *unit*, *work cell* and so on, 2) data flow models to specify functional as well as data flows and 3) object models used to model information types that need to be exchanged. The main scope of those models is related to the questions about how to create a product, about the capability to produce a product and about actual production of the product. During the process of defining interfaces between MES and ERP systems, abstract models are iteratively refined into more detailed ones. The models involved as well as their level of detail are illustrated in Figure 2.

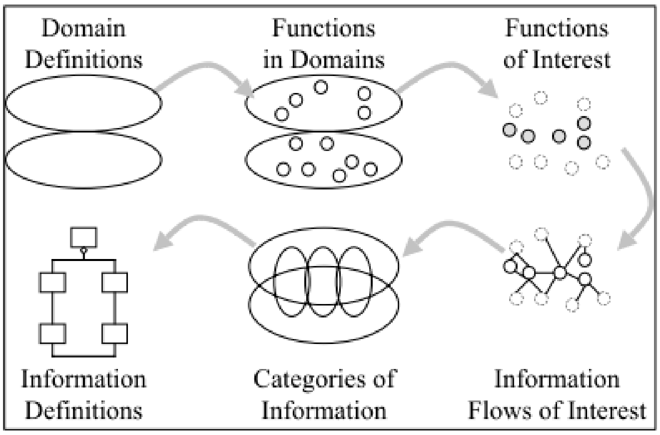


Figure 2: Models defined in ISA-95

ISA-95.02: Object Model Attributes

Part 2 details the object models in part with attribute descriptions. This provides a basis for the design of message formats as well as databases. Whereas part 1 of the standard describes the data that needs to be exchanged between MES and ERP systems with the help of the predefined object models on entity level, part 2 extends the latter by attributes for each object. In addition, a comprehensive description and examples for all the attributes are given. The production capability object contains e.g. the attributes ID, description, type and others.

Other parts

- ISA-95.03: Activity Models of Manufacturing Operations Management

  Part 3 of the standard focuses on reference-models for activities and functions operating between business planning and logistics (level 4) as well as process control functions (level 2).

- ISA-95.04: Object models and attributes for Manufacturing Operations Mgmt

  Part 4 further defines object models and attributes involved in data exchange between the activities identified and specified in part 3.

- ISA-95.05: Business to manufacturing transactions

  Part 5 of the standard describes the exchange of information (based on part 1 and part 2) of applications performing business and manufacturing activities with regard to information collection, retrieval, transfer and storage in support of enterprise-control system integration.

3.3.2 XML representation: B2MML

The object models and their attributes, defined in part 1 and 2 of the standard, are described on a conceptual rather than on an implementation level. The IEC 62264 part 2 standard provides in its informative sections examples of possible XML-encodings of the object models, it does not describe a complete XML vocabulary. However, the Business-To-Manufacturing Mark-up Language (B2MML) has been introduced by the WBF[3] and provides xml vocabularies for the object models in the standard.

3.4 Ebbits approach

The architecture style of ebbits can be characterized as *Event-driven SOA*, integrating *services* with advanced semantic *event* processing and *business rules*.

Events can range from lower level atomic signals to higher level semantically enriched message carriers. On a higher abstraction layer ebbits events are mapped to business rules, which can make use of advanced services, to implement the business logic for reporting, actuation of devices, as well as for further event generation. The first level of event processing is the mapping of events to the perimeter systems which are generating the event stimuli and providing entry points for actuation.

Rules may range from fairly low-level operational rules (e.g. the triggering of an alarm based on some threshold value), to more QoS based and business oriented rules.

[3] http://www.wbf.org, WBF- The Organization for Production Technology, formerly "World Batch Forum"

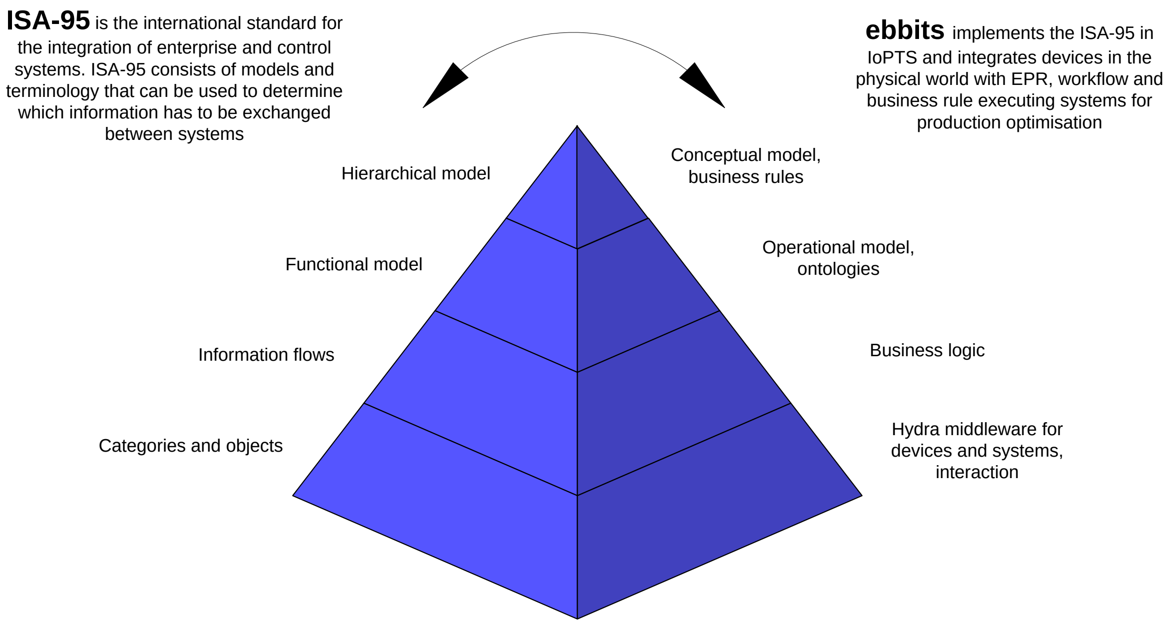


Figure 3: ISA-95 Enterprise control system models mapped to ebbits hierarchical service orchestration.

Ebbits rules are expressed over domains defined by the ontologies and data models derived from the two application scenarios in combination with the generic models given by applicable standards.

To this end, the starting point for the work in WP6 includes the two first parts of the ISA-95/IEC62264 standard, part 1: Models and terminology and, Part 2: Object Model Attributes. This scope may however change over the project iterations.

In general, the implementation of the ISA-95 by means of B2MML could be a potential adoption point of the standard in the context of the ebbits project as a whole. It would introduce existing and well established taxonomies and terminologies into the project.

4. Ebbits terminology and taxonomy

4.1 Ebbits terminology and business objects

The models used in ebbits should support both the manufacturing scenario and the traceability scenario. These two scenarios are very different in their objectives, requirements and in the technology employed. However, the overall objective of ebbits is to connect enterprise systems with the Internet of things and services (IoPTS), the architecture must be flexible enough to support a variety of business object models and terminologies.

Standards are important to middleware platforms like ebbits, this does not only relate to the generic level technology standards (e.g., web services, knowledge representation formalisms), but also to more domain focused ones to promote business systems interoperability. This is where domain specific terminologies and taxonomies become important. At the onset of the ebbits development, the ISA-95 standard was proposed as a possible baseline and source for the selection and design such a standard terminology and models to support interoperability.

There are a number of questions to consider when evaluating applicability of a framework like ISA-95 to ebbits, e.g.,

- In general, how does this approach compare with the notion and vision of the IoPTS?
- How can the manufacturing scenario be supported?
- How can the traceability scenario be supported?
- How can the sensor and monitoring perspective be supported? e.g., energy consumption monitoring for manufacturing control function.

4.2 Application/role of ISA-95

The Models and Terminology part of the ISA standard provides a conceptual model intended to characterize the Enterprise-Control systems (ERP-MES) interface, it defines the following sub models,

- A functional model, depicting functions at the ERP-MES boundary.
- Information flows between the functions.
- Information categories and object models. The object models are further detailed at the attribute level in part 2 of the standard.

We will below characterize these different models, with the objective of allowing us to further analyze their applicability in ebbits.

4.2.1 The functional model

The functional model provides the context for the information flows and systems interfaces. The standard explicitly states that this model is not intended as an organizational model, but as a pure functional one, where functions might be mapped to different organizational units depending on the applying enterprise.

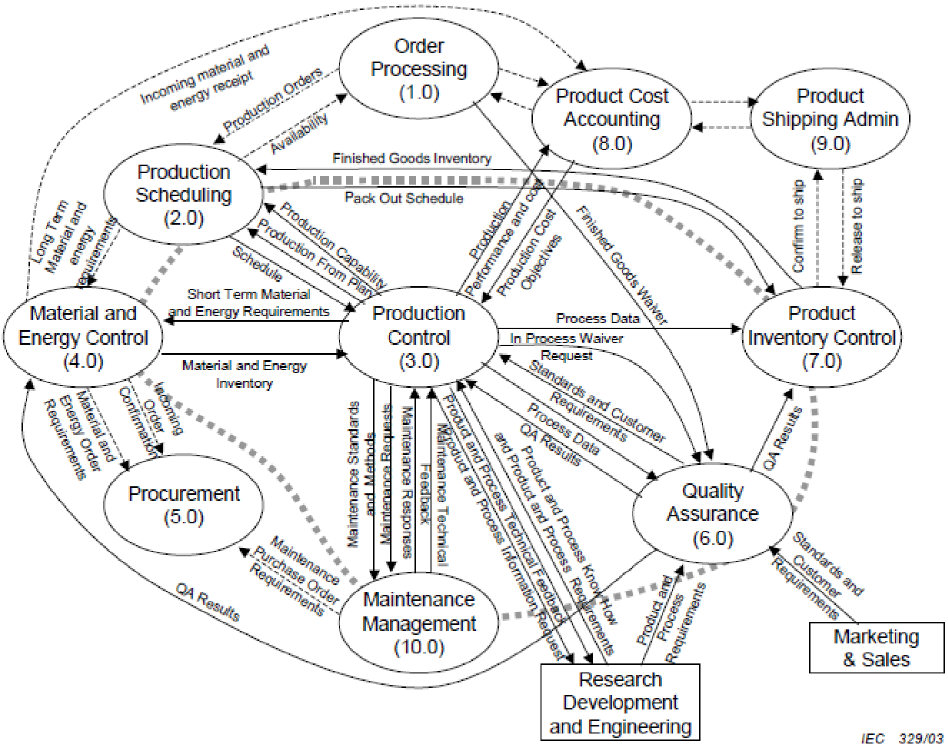


Figure 4: Function and information view of the Enterprise-Control boundary

The figure shows the overall function model where the central function "Production Control" represents the set of (sub) functions for manufacturing control. The connections between the functions indicate the main information flows relevant for the control of manufacturing, and the interface between enterprise and control functions is shown by the thicker dotted line. At this abstraction, some functions may map to either the control realm or the enterprise domain.

This is a fairly high-level model, and each function is further characterized by a set of sub-functions, with descriptions of their typical objectives, services and effects. The standard does not further formalize the definition these functions, but provides comprehensive text descriptions of these functions and sub functions.

We will not further detail the functional model in this deliverable, but give examples of those functions that can be considered for mapping to the ebbits architecture. These are also the functions considered to be at the intersection of enterprise-control interface in the standard model.

The *Production Control* function is subdivided into the following sub functions,

- Production control main functions
- Process support engineering
- Operations control
- Operations planning

Relevance and mapping to ebbits would here e.g. be, monitoring activities and reporting.

Material and energy control include sub functions for,

- Calculation of inventory, material loss and energy utilization
- Generation of request for materials and short/long term energy
- Management of

Relevance and mapping to ebbits would here e.g. be, energy efficiency monitoring and reporting.

Quality Assurance, may include activities for,

- Testing and classification of materials and equipments
- Collection of material quality data and test results
- Checking product data against quality requirements

Ebbits relevance would e.g., be related to monitoring and data fusion and reporting.

Product inventory control, example functions,

- Reporting on inventory, losses and balances
- Managing inventory of finished products
- Physical loading and shipment of goods and products

Role of ebbits could here be on support for the traceability scenario, including provision for identification and tracking of goods and products.

Maintenance Management

- Providing maintenance programmes
- Equipment monitoring to predict failure
- Provision of status and technical feedback on performance and reliability

The role of ebbits would here also be in sensing, monitoring and data fusion.

There might additional and other functions and sub functions, depending on the domain and the particular application, with relevance to the ebbits architecture and its role in the ERP-MES interface.

4.2.2 Information flow

The functions are interrelated by the information flows representing the main interfaces for information exchange. These information flows should be further analysed with respect to the use cases in the two ebbits scenarios. Such analysis may lead to a set of domain neutral ebbits workflows provided as part of the overall ebbits architecture.

The content and structure of the information exchanged are based on the definitions given in the object models of the standard.

4.2.3 Information categories and Object models

The standard framework defines three broad areas of information corresponding to,

- Product definition, i.e., information about products and requirements for their production.
- Production capability, i.e., information about capabilities to produce products e.g., with respect to capacity and equipment status.
- Production, i.e., information about scheduling and performance.

The broad areas are further structured into categories

These categories are further subdivided into set of objects corresponding to various resources including,

- Equipment (e.g., tools, machines, devices, lines, units, cells)
- Personnel (e.g., people, skills, staff groupings)
- Material (materials, material lots, waste materials, energy, end products).

Object models are defined for these resources, which then can be used to design interfaces and for information exchange. All information flows in the framework are considered to be built up from one or more of these resource objects.

Modelling equipment resources

The models describing concepts and terminologies related to various forms of equipment are shown below. As a frame of reference, a model of an Equipment Hierarchy shows the different levels of responsibility and controls in an enterprise (an overall decision making authority). The model relates to the different levels of responsibility to the Functional Hierarchy depicted in Figure 1.

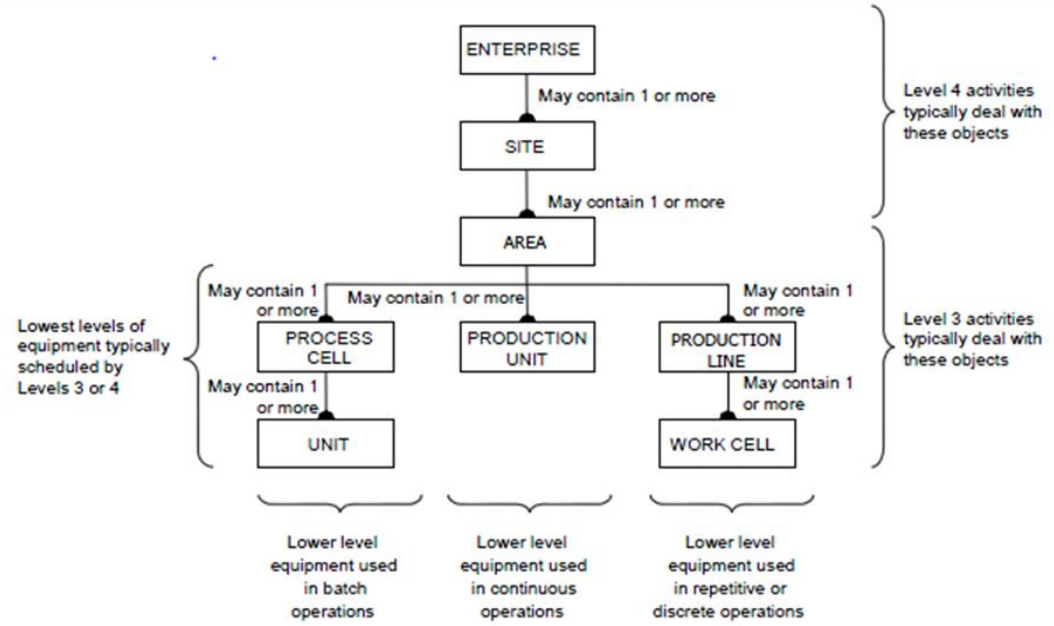


Figure 5: Equipment hierarchy (IEC 62264-1)

Concepts like Sites and Areas are any physical, geographical or logical groupings as determined by the enterprise or business itself. Cells, units and lines group the lower level equipments, e.g., a Production Unit may contain various types of equipment modules, sensor and actuators.

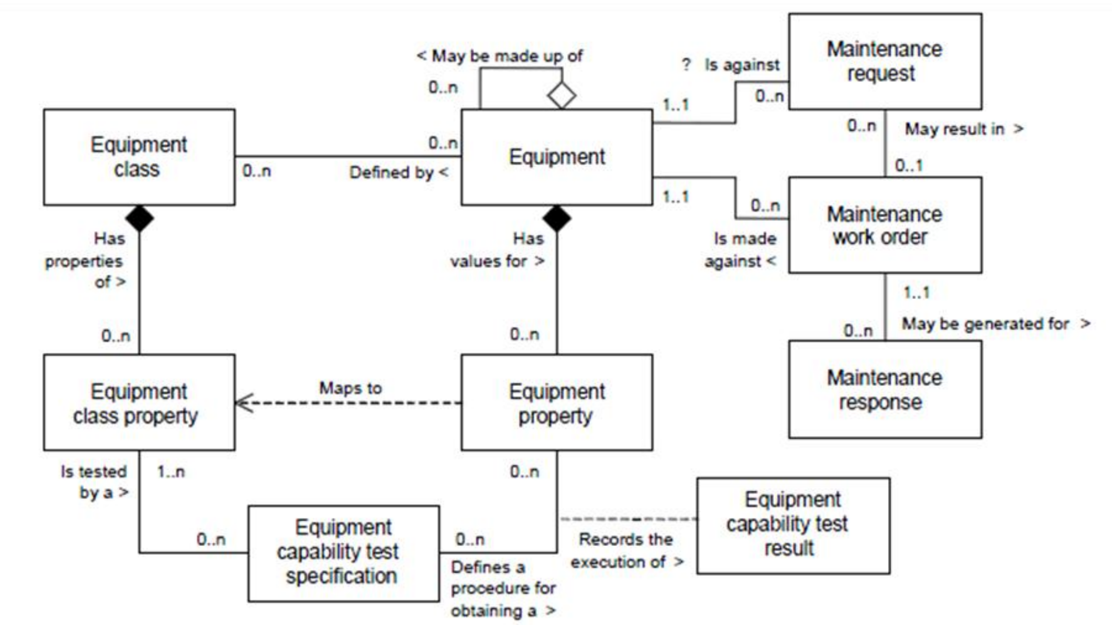


Figure 6: Equipment model (IEC 62264-1)

The standard defines "equipment" in broad sense, meaning it could represent anything that could be represented in the Equipment Hierarchy model in Figure 5. Thus it could be used to model production lines or units as well specific machinery or sensor equipment.

The concept of Equipment class is here understood as any grouping of Equipment objects with similar characteristics with respect planning and scheduling. Any equipment could be a member of 0 or more equipment classes. Equipment objects can also be related in multiple levels, forming equipment aggregates.

Domain specific properties are defined as instances of the Equipment class property and the -equipment property objects in the model. A Property definition is composed of an identifier, a description and a value with possible type constraints.

Properties may also have an associated QoS-like specification consisting of a Capability Test Specification with a corresponding Capability Test Result. In the case of Equipment properties this could be a test certifying some performance property, like sensor battery life.

This approach to extension, by defining new properties as instances to a meta property, means that the base model is not modified, and the logic for processing the extension properties is left to the model implementation and the application. In some model implementations (e.g., XML) this might mean that no typing or validation support is available for new properties.

Modelling material resources

The Materials model is used to describe single material units as well as the properties of collections, or lots, of materials.

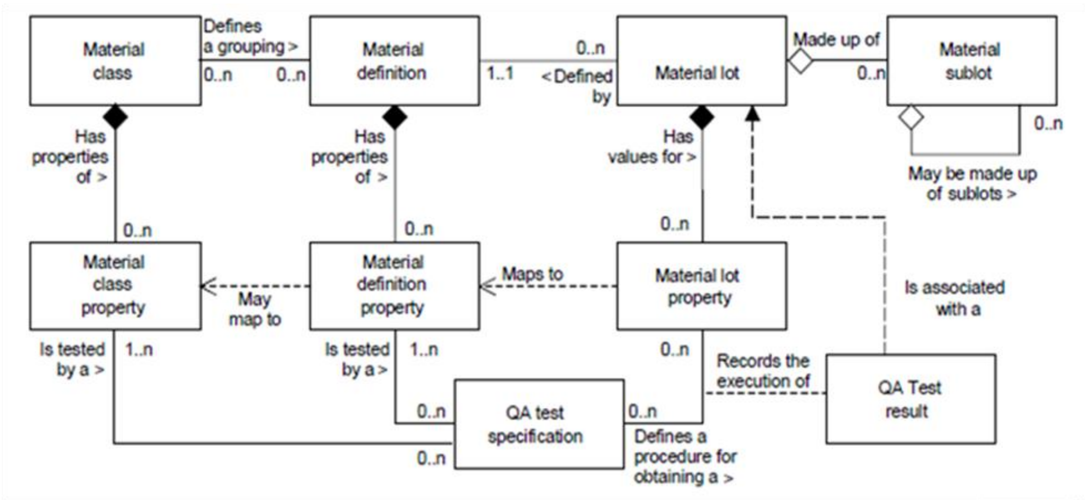


Figure 7: Materials model (IEC 62264-1)

The model distinguishes between class, definition and lot/sub lot, for materials. A material class in food industry could model Pork, a material definition is a specific form and quality of pork, 70% lean with bones, whereas a Material lot would model a certain shipment or storage unit, with slaughter date, quality certification etc. The end product (also a Material) could be Trotters.

A Quality Assurance Test (and results) may be defined for material properties. The same approach to property definition applies as for the Equipment model.

Modelling personnel resources

This sub model describes any staff or human resources relevant for the ERP-MES interface. The model thus distinguishes Person objects from Personnel Class objects.

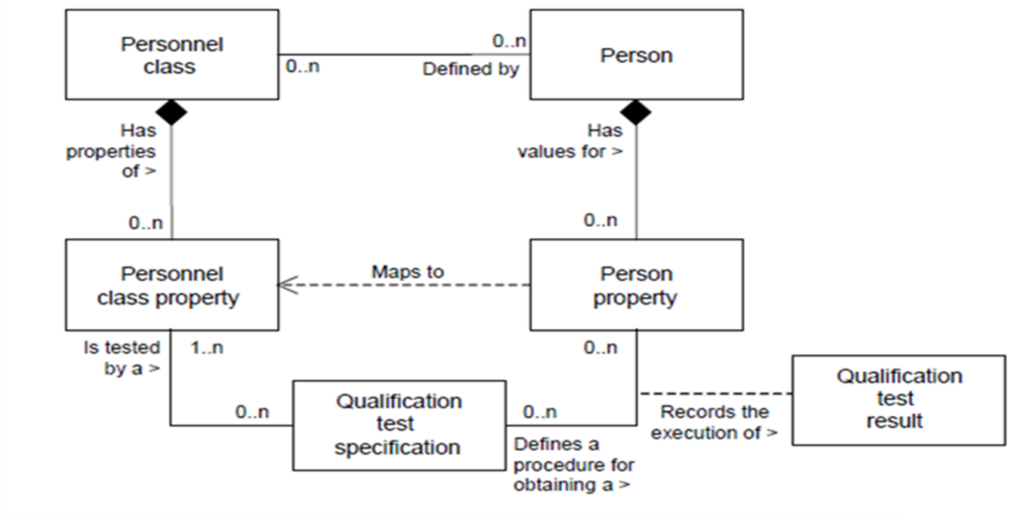


Figure 8: Personnel model (IEC 62264-1)

The Personnel Class should be seen as a personnel grouping or staff category, e.g., representing maintenance staff. A Quality Test Specification could here refer to qualifications for handling certain types equipments. Properties are defined in the same way as for the Equipments and the Materials models.

4.2.4 XML mark-up

B2MML is an XML implementation of the ISA 95 (IEC/ISO 62264) family of standards. B2MML consists of a set of XML schemas based on the W3C XML Schema language (XSD) that defines the vocabulary corresponding to the object models in the ISA95 standard. It is a full implementation of the ISA-95 as a set of XML schemas, one for every object model. This allows, data between ERP and ME systems to be shared by means of XML messages of exchange or on storage.

Below is the XML vocabulary (schema) for the Person Object model (not fully expanded), where the PersonType element defines the xml element type for the Person object. Accordingly, there is also a PersonelClassType definition for the PersonnelClass Object (not shown here).

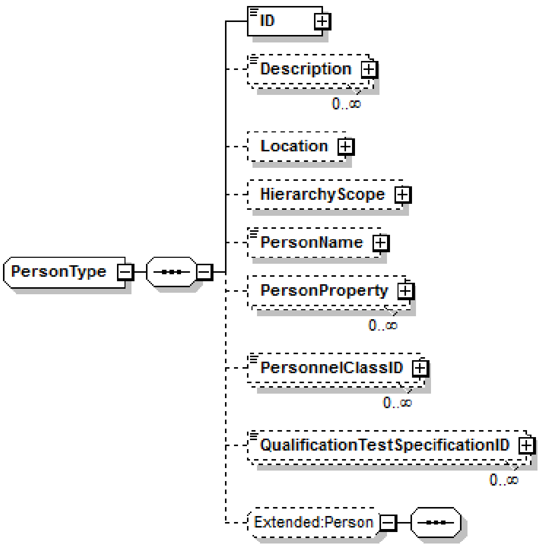


Figure 9: B2MML schema subset

A corresponding XML-instance is shown below, validated against the schema.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<ebbits:Person xmlns:ns2="http://www.wbf.org/xml/B2MML-V05-AllExtensions"
xmlns:ebbits="http://www.wbf.org/xml/B2MML-V05"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://www.wbf.org/xml/B2MML-V05 /Schemas/B2MML-V05-Personnel.xsd">
  <ebbits:ID>ID0</ebbits:ID>
  <ebbits:Description>ebbits staff description</ebbits:Description>
  <ebbits:Location>
    <ebbits:EquipmentID>ebbitsSensorNetwork_</ebbits:EquipmentID>
    <ebbits:EquipmentElementLevel>Site</ebbits:EquipmentElementLevel>
  </ebbits:Location>
  <ebbits:PersonName>John Doe</ebbits:PersonName>
  <ebbits:PersonProperty>
    <ebbits:ID>ID1</ebbits:ID>
    <ebbits:Description>workShift</ebbits:Description>
    <ebbits:Value>
      <ebbits:ValueString>Daytime</ebbits:ValueString>
      <ebbits:DataType>Text</ebbits:DataType>
      <ebbits:UnitOfMeasure>shiftEnumedration</ebbits:UnitOfMeasure>
    </ebbits:Value>
  </ebbits:PersonProperty>
  <ebbits:PersonnelClassID>ebbitsMaintenance_01</ebbits:PersonnelClassID>
</ebbits:Person>
```


Figure 10: Example B2MML xml instance

Additional examples include material definitions relating to food production (ebbits traceability scenario). These are currently under development.

The usage guidelines of B2MML also recommend different extension mechanisms of the schemas to support application domain adaptation.

4.2.5 Domain extensions of the Object Model

The object models may be modified in order to allow for domain specific extensions. The (WBF) B2MML schema architecture recommends a set of different extension methods, including methods without modifying the original schemas, as well as schema modifications.

Examples of extensions in order of their recommended use,

- Addition of properties to objects. New properties are defined as instances of a Class Property element. No schema extension, but also no validation capability.
- Defining parameters, such as parameters for product specific information or process segment information
- Definition of process segments to represent activity sets (Process segments are collections of capabilities needed for a segment of production independent of product).
- User defined Enumerations. E.g. Domain/application specific extensions to standard name lists.
- Use of Substitution groups. An XML schema mechanism that allow (or groups of) elements to be substituted for others in run-time. This is the B2MML recommended method for schema extensions and replaces the use of so called wildcard-elements (i.e., xsd:Any element) supported in previous versions. Allows for validation of extended schemas.
- Modifying original (base) schemas. Although terms of usage do not prohibit this, it is not recommended in the usage guidelines since it may impair on interoperability. Two approaches to schema extensions are possible
  - Extended Namespace, where a user defined domain specific schema imports B2MML schemas through namespace references.

- o The use of wildcard (xsd:any) elements. Where a generic any-element may be used as a placeholder for any schema structure.

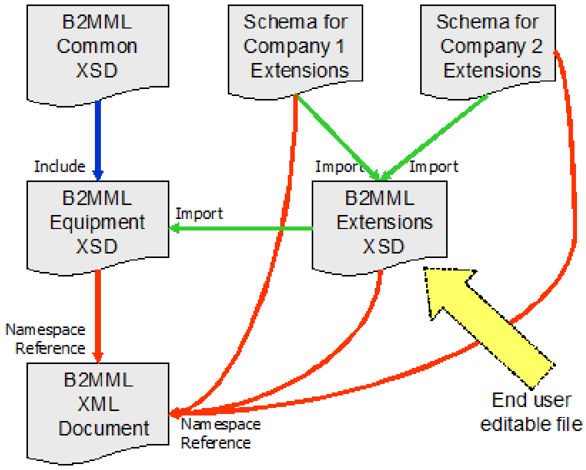


Figure 11: B2MML schema namespace extension (WBF.org)

The choice of extension mechanisms depends on the complexity/size of the required extension. It also depends on whether validation (typing) of the added elements against base schemas is required. This is in principle only possible if schema extension mechanisms are used.

4.3 Ontological support

In the Description of Work for Task 6.6 we find some specification for operational rules and the needed ontologies: "At the process layer we find a comprehensive set of different kinds of operational rules related to specific operative business decision to be executed. These rules often take the form of descriptions of workflow or work processes, operating procedures, etc. The operational rules focus on functions within the enterprise and how the different functions interact, which is described in ontologies."

The execution of operational rules is described below in Section 6. Within the ebbits project, the semantic knowledge infrastructure, which is used to make the above required interaction of different functions and mechanisms possible, is primarily done in Work Package 4. Therefore, in this section only a short description of an ontology specification is given which is amenable to implementation using rule-based technologies as described Section 6. Otherwise, it is referred to the already published Deliverables D4.1 "Analysis of semantic stores and specific ebbits use cases", D4.2 "Knowledge representation formalism analysis" and D4.3 "Coverage and scope definition of a semantic knowledge model" of Work Package 4.

In (Motik, Grau, Horrocks, Wu, Achille, & Lutz, 2009), an OWL 2 Web Ontology Language Profile has been proposed as a W3C recommendation. This profile is called OWL 2 RL and can be used for application "that require scalable reasoning without sacrificing too much expressive power. It is designed to accommodate both OWL 2 applications that can trade the full expressivity of the language for efficiency, and RDF(S) applications that need some added expressivity from OWL 2". As the ontological support within the ebbits project should be based on OWL and RDF(S) and for the business rule execution, an ontology with scalable reasoning and the ability to be encoded in rules is needed, this profile is proposed as a good candidate to combine ontological knowledge with rules in ebbits.

The profile is derived from OWL 2 by restriction and a set of entailment rules (Bishop & Bojanov, 2011), (Motik, Grau, Horrocks, Wu, Achille, & Lutz, 2009):

- Restrictions are placed on OWL 2 Full in the use and position of certain OWL 2 language features to avoid the need to infer the existence of individuals not explicitly present in the knowledge base, and to avoid the need for nondeterministic reasoning.

- As a set of entailment rules to be applied to the RDF serialisation of an OWL ontology, where these rules represent a partial axiomatisation of the complete OWL 2 RDF-Based Semantics. The rules are given as universally quantified first-order implications over a ternary predicate which represents a generalization of RDF triples.

The entailment rules for OWL 2 RL, which are given in the W3C recommendation, are grouped into separate tables for defining the semantics for: equality, property axioms, classes, class axioms, data types and schema vocabulary. They take a variety of forms (Bishop & Bojanov, 2011), (Motik, Grau, Horrocks, Wu, Achille, & Lutz, 2009):

- **Triple Pattern Rules:** For this kind of rule, the rule body and head are made up of atomic formulae representing triples in the RDF graph.
- **Assertional Rules:** Here the "if" part of the rule is empty, in which case they can be considered as being always applicable. An example is as follows:
- **Consistency checks:** The "then" part of these rules contains false only, in which case the initial input RDF graph should be considered inconsistent when the premises of the rule hold.
- **List rules:** These rules make use of an abbreviation for processing RDF collections, e.g. when defining classes as the intersection or union of a closed set of classes.
- **Data-type Rules:** These rules imply a special processing for data-types, for example:

5. Business rules representation

5.1 Candidate languages

Is this section, candidate languages for modelling business processes and rules are described. For the actual execution of the models, the Drools Framework is proposed, which can execute rules and processes. This framework is detailed in Section 6.

5.2 Modelling business processes. BPMN, WS-BPEL, BPEL4People

BPMN (*Business Process Modelling Notation*, http://www.bpmn.org) is a standardised graphical formalism for representing business processes as structured workflow sequences of activities (tasks, sub-processes) performed by involved actors. Since the BPMN outputs are merely graphical schemas, this formalism is primarily dedicated to human users ranging from technical developers responsible for implementing a process-related technology to business analysts and decision makers that can use the models for monitoring and maintenance of the processes. With this respect, BPMN is often employed as an alternative or complement to the UML diagrams, namely to overcome the gap between the design and actual implementation of business processes.

The BPMN formalism includes a specification of business process diagrams in so-called pool and lanes for each of actors, visual appearance of graphical elements for activities, events, gateways, connectors, and artefacts exchanged in the modelled process. The specification also defines rules and constraints for composing well-formed process models. Nowadays, most commonly used BPMN version is still the BPMN 1.2 (2009), which is supported by a wide range of proprietary and open source tools for constructing and maintaining the process models. The most recent version BPMN 2.0 (2011), officially released in January 2011, contains extensions towards collaboration and choreography diagrams, as well as a broader support of exchange formats such as XSD, XMI, and XSLT.

Finally, it is important to stress that BPMN should not be seen as a "true" formalism for expressing business rules, since it deals with business processes on a higher level of abstraction. In particular, it addresses the so-called *abstract business processes*, which "represent the interactions between a private business process and another process or participant" (BPMN 1.2, 2009). The business logic, which includes low-level rules and concrete executable services for abstract BPMN process models, needs to be specified in a separate formalism such as BPEL, WSMO, etc. The BPMN 2.0 specification already contains an execution semantics and means of mapping BPMN models to the executable WS-BPEL. However, the support of these valuable features is still quite rare and limited in currently available business process modelling tools.

BPEL (*Business Process Execution Language*, also known under a short generalised name of WS-BPEL or BPEL4WS) is an OASIS standard for modelling and representation of a low-level business process behaviour (BPEL 2.0, 2007). It provides means for describing business processes, together with their inner logic and rules, in both abstract and executable ways. Both descriptions share the same set constructs and have equal expressive power. Executable models represent an actual behaviour of participants in a business interaction, including a grounding on concrete services and other operational details. Abstract BPEL models, which must be declared explicitly as "abstract", may hide some of required operational details and thus they serve as higher-level descriptions – they may be used to describe observable message exchange behaviour of each of the parties involved, without revealing their internal implementation.

BPEL notation is human-readable and is expressed in the XML format. A BPEL statement consists of three basic components:

- Programming logic: BPEL process model itself;
- Data types: XSD statements;
- Input/Output parameters: WSDL statements, grounding to particular web services.

The major building blocks of BPEL process models are the activity elements such as <receive>, <reply>, <invoke>, <assign>, <throw>, <exit>, etc. There are two types of activities: structured activities may contain other activities and define a business logic between them, while basic activities only perform their intended purpose. An example of the BPEL process structure is depicted in Figure 12.

```
<process name="..", targetNamespace=".." > //global settings, schemas
 <partnerLinks>..</partnerLinks> //references to wsdl of used services
 <variables>..</variables> //declaration of local variables
 <faultHandlers>..</faultHandlers>
 <sequence> //start of workflow sequence
  <assign>..</assign> //set up variables, if needed
  <receive service1_attribs>..<receive> //read an input from service1
  <flow> //start of a sub-process
   <sequence>
    <receive attribs>..</receive>
    <invoke service2_attribs>..</receive> //invoke a service2
   </sequence>
  </flow>
 </sequence>
</process>
```

Figure 12: A structure of BPEL process model

BPEL provides capabilities to model providing and consuming web services, to define a structure of the process logic, repetitive activities, parallel processing, manipulations with data, and many additional advanced concepts.

Connection with semantically described web services is usually accomplished by means of the OWL-S and/or SA-WSDL standards (cf. D4.1, sections 3.11.1-2). The service orchestration can be handled by OWL-S, while the choreography is described in BPEL (Martinek & Szikora, 2008). The semantically enriched web services may run on an OWL-S Virtual Machine, while the complex processes, described in BPEL are operated on a BPEL run-time. A suite of available BPEL run-time engines includes, for example, open source engines of JBoss or Apache, or proprietary solutions provided by Oracle, Parasoft, etc.

It may happen that the services of a workflow are not available in a form of automatically executable web services. Instead, a human interaction is required and/or such a service may be invoked and executed offline (Furdik et al, 2011). These services, referenced as so-called "human tasks", require an extension of BPEL, which is standardised by the OASIS consortium as the WS-BPEL Extension for People (*BPEL4People*, (Agrawal, A. et al, 2007)). The BPEL4People extension is defined in a way that it is layered on top of WS-BPEL 2.0. It introduces a set of elements (e.g. <b4p: humanInteractions>, <b4p: peopleAssignments>, <b4p: peopleActivity>, etc.), which extend the standard BPEL elements and introduce the modelling of human interactions.

5.3 WSMO framework and BPMO

WSMO (*Web Service Modelling Ontology*, http://www.wsmo.org) is a framework specifically designed for semantic description and maintenance of web services. The conceptual model of WSMO ontologies, consisting of four major top-level elements of ontologies, goals, web services, and mediators (cf. section 3.11.3 in D4.2), was published in 2005 as the W3C Member Submission (de Bruijn et al, 2005). The whole WSMO framework, which is based on the specified WSMO ontologies, includes the declarative WSML mark-up language (http://www.wsmo.org/wsml/), the WSMX execution environment (http://www.wsmx.org), the wsmo4j Java API (http://wsmo4j.sourceforge.net) and the WSMO Studio toolkit (http://www.wsmostudio.org).

Particular functional components of WSMO are available as separate Eclipse plug-ins or as integrated into the stand-alone application of WSMO Studio toolkit (Dimitrov et al, 2007). In the WSMO Studio toolkit, the functionality is grouped into so-called perspectives that enable a manipulation with WSMO ontologies, ontology repositories (i.e. local ORDI or external IRS-III, Sesame, etc.), semantic descriptions of web services (SA-WSDL), and business process models. The toolkit and the related

WSMO framework is provided under the LGPL license, while some of external plug-ins have their own license models such as Apache Software License, Eclipse Public License, and MIT License.

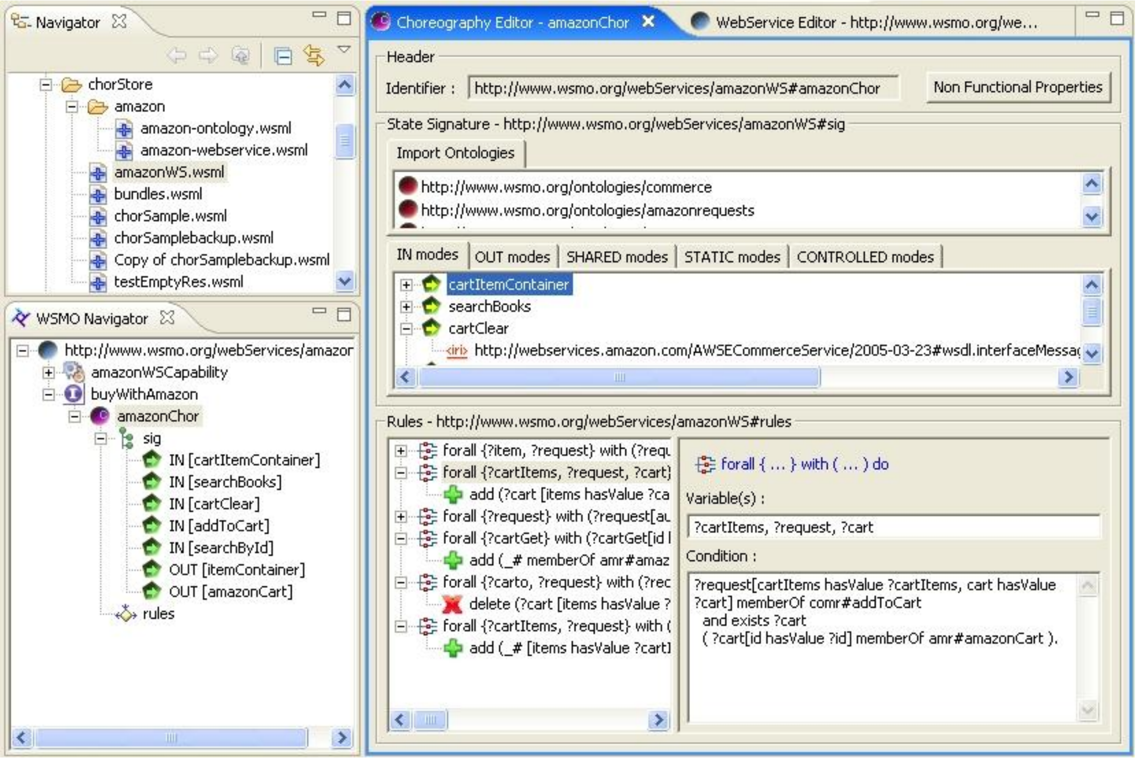


Figure 13: Service choreography rules created in the WSMO Studio editor (Dimitrov et al, 2007)

Business rules and processes are supported in WSMO on two levels. On the lower level of services, the embedded Choreography Editor tool of WSMO Studio provides functionality for creating WSMO-based choreographies of semantically enhanced and grounded web services (Roman et al, 2006). The choreography is based on state signatures and transition rules, formally expressed in WSML. The editor enables specifying the Capability, Interface, and Choreography parts of the web service interfaces, as it is depicted in Figure 13.

On the level of business processes, the modelling and composition of semantically annotated web services into workflow sequences is supported in WSMO by means of BPMO ontologies (Hepp & Roman, 2007). Namely, the BPMO perspective of WSMO Studio enables a visual creation of process models, as it is presented in the figure below.

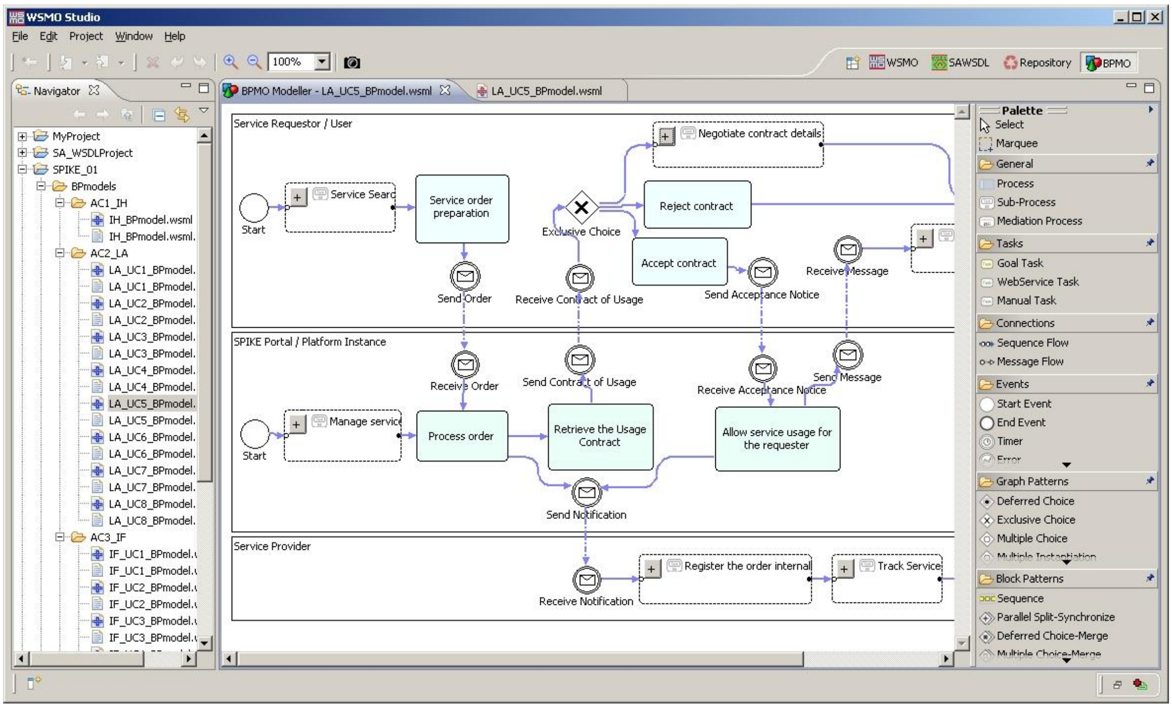


Figure 14: BPMO representation of a business process in WSMO Studio toolkit

The palette of BPMO components in WSMO Studio roughly corresponds to the graphical elements of BPMN 1.2. The tasks in the workflow may represent web services, WSMO goals (which are subsequently grounded to particular web services), or manual activities. The tasks and sub-processes can be described by IOPE characteristics, i.e. by required inputs, preconditions, generated outputs, and post-conditions / effects. All these properties can be specified as references to an existing WSMO ontology.

The semantic business process modelling in WSMO implements the methodology that was developed within the FP6 integrated project SUPER (http://www.ip-super.org). Namely, the BPMO editor of WSMO Studio employs the following SUPER ontologies:

- the *Upper Process Ontology* (UPO, ver. 2.1.3), which defines top-level concepts such as task, goal and condition;
- the *Business Process Modelling Ontology* (BPMO, ver. 2.0.1), which extends the UPO into a full process ontology. It provides abstractions over standardised business process modelling notations such as BPMN and EPC (Scheer et al, 2005);
- *sBPMN*, *sEPC* and *sBPEL* / *BPEL20*, which are ontologised versions of subsets of the BPMN, EPC and WS-BPEL specifications respectively, all designed as extensions of the main BPMO ontology. In addition, sBPEL is enriched with the BPEL20 ontology containing WSMO representations of BPEL 2.0 for goal-oriented discovery, mediation and execution of services.

It implies that a business process model produced in the BPMO editor of WSMO Studio is internally represented as a set of instances of BPMO concepts such as Process, SubProcess, Workflow, StartEvent, ReceiveMessageEvent, SendMessageEvent, etc. A transformation of such a process model to the corresponding executable workflow (and vice-versa) can be accomplished by the BPMO2sBPEL translation plug-in (Norton, Cabral & Nitzsche, 2009), which employs F-logic-like rules within WSML-Flight axioms that infer instances in the target ontology from those in the source.

Inner structures of BPMO and WSMO are represented in WSML, which is a specific declarative formal language for ontologies, rules, business logic, and other semantic data. To enable a wider compatibility with other ontology languages, a translation mechanism between WSML and RDF(S) / OWL was proposed in (de Bruijn & Kopecky, 2008) and (de Bruijn, Polleres, Lara & Fensel, 2005).

The corresponding import/export wizards are included in WSMO Studio as well; however, according to our experience, the correctness of translation is rather limited for larger or more complex ontology structures.

To summarise, the WSMO framework is a comprehensive toolkit for designing and maintaining semantic business rules and process models, which are grounded to web services. However, a lack of effective transformation of native WSML format to/from RDF(S) or OWL representations can be considered as a serious drawback for ebbits. In addition, development activities around the WSMO framework are nowadays somehow suppressed, since the last code update at http://sourceforge.net/projects/wsmostudio/files/ is dated in December 2008.

5.4 RuleML

RuleML is an effort to standardize a normalized mark-up for expressing, translating and interchanging rules in semantic web. It is a family of sublanguages whose root allows access to the language as a whole and whose members allow to identify customized subsets of the language. It supports interchanging rules among platform-specific and various standards of rule language ranging from Derivation Rules that extends SQL views to Reaction Rules which extends SQL triggers.

Derivation Rules is specified via XML Schemas (XSDs) that correspond to the expressive class of a specific RuleML sub language. e.g.: SWRL for ontology(Horrocks, Patel-Schneider et al. 2004), FOL RuleML for First order logic, OO RuleML for all sublanguages, and SWSL(Battle, Bernstein et al. 2005) for web services. However, RuleML uses Datalog(Abiteboul and Vianu 1991) as the Kernel of its family of sublanguages. Datalog constitutes intersection of SQL and Prolog. The facts in Datalog are defined correspond to explicit rows of relational tables and rules belong to the tables defined by views. Datalog is able to represent relational information for the columns that contain natural language phrases E.g.:

Phrase:
"A customer is premium if their spending has been min 5000 euro in the previous year."

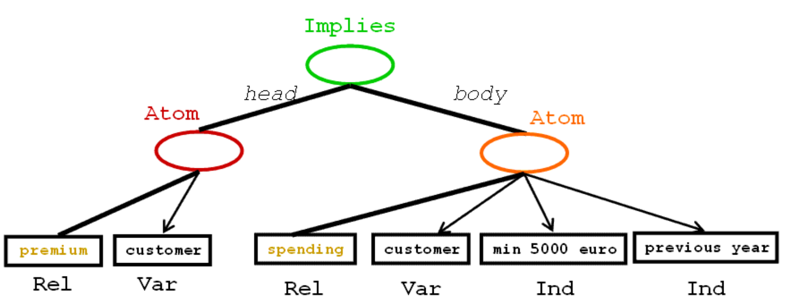
Graph	Mark-up
	*<Implies> <head> <Atom> <Rel>premium</Rel> <Var>customer</Var> </Atom> </head> <body> <Atom> <Rel>spending</Rel> <Var>customer</Var> <Ind>min 5000 euro</Ind> <Ind>previous year</Ind> </Atom> </body> </Implies>*

Figure 15: Datalog representation(Boley, Grosof et al. 2005)

RuleML contains a hierarchy of rules including(Boley, Grosof et al. 2002):
Reaction rules (event-condition-action rules), which can only be applied in the forward direction.

Transformation rules (functional-equational rules) that prefers to use backward direction

Derivation rules (implicational-inference rules) that can be applied either in forward or backward direction. Backward directions can decompose the proof of a goal to proofs of its subgoals.

Specialized to facts ('premiseless' derivation rules), which has no notion of an application direction.

Queries ('conclusionless' derivation rules) that proofs top down goals using backward direction and also can use forward direction via 'goal-directed' bottom-up processing.

Integrity-constraints (consistency-maintenance rules), which are usually forward-oriented. But they can instead be backward-oriented, trying to show consistency by fulfilling certain conditions

Related to RDF, RuleML rules can be represented in RDF through a translator. This can be implemented through an XSLT stylesheet translating RuleML rules to RDF. There exists also an implementation, FRODO rdf2java tool (DFKI 2011) that uses Java classes instead of RDF Schema to generate textual horn clauses as well as XML RuleML syntax. RuleML Lite is a derivation of RuleML built on datamodel that integrate data models of XML and RDF. It supports unary and binary predicates.

5.5 Windows Workflow Framework

Windows Workflow Framework (WF) is a workflow framework from Microsoft consisting of workflow designer and engine. The advantage to isolate business rules and logic in a separate workflow is that business rules can be changed at runtime without having to redesign the whole implementation(Scribner 2007). A workflow consists of activities that are going to be executed by the WF runtime that provides mechanisms for persisting a workflow's state, tracking, and etc. These activities are expressed in graphical representation that is similar to control structure of a programming language, which then translated into XAML(MacVittie 2006) by the workflow designer. WF supports integration with web service technology such as soap and rest. One is able to call web service inside activities as well as expose workflows as a web service.

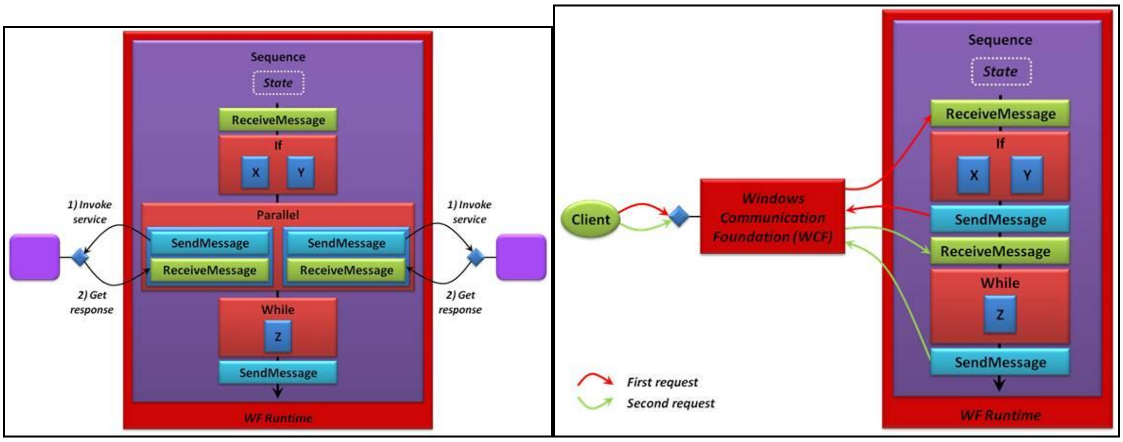


Figure 16: (Left) Workflow consuming webservices, (Right) Workflow exposed as a webservice(Chappell and Associates 2009)

Workflow can be hosted in a simple .NET executable process, however Microsoft offers an automatic monitoring of the workflow runtime (e.g.: automatically restart a failure) and SQL-Server based persistence integration when workflow is hosted in a worker process inside Internet Information System (IIS) with Dublin extension(Chappell and Associates 2009).

WF also offers a seamless integration of rules capabilities that extend from simple conditions that drive activity execution behavior all the way up to complex RuleSets executed by a full-featured forward chaining rules engine. RuleSets can be applied at any point in the workflow which allows developers to make decision whether to apply their logic in the workflow model, rules, or code without having to worry about the integration implications of their decision(Willis 2008).

The rule engine supports the following expression:

- Equal ("==" or "=")
- Greater than (">")
- Greater than or equal (">=")
- Less than ("<")
- Divide ("/")
- Modulus ("MOD")
- AND ("AND", "&&")
- OR ("OR", "||")

- Less than or equal ("<=")
- Add ("+")
- Subtract ("-")
- Multiply ("*")
- NOT ("NOT", "!")
- Bitwise and ("&")
- Bitwise or ("|"

If rules are implemented as condition on workflow activities, it can be defined in a policy activity file that encapsulates the definition and execution priority of the rules. This policy file can also be updated at runtime.

The evaluation mechanism can be conceptually described with the following procedure:

- Start with the list of active rules.
- Find the highest priority rule.
- Evaluate the rule and execute its Then/Else actions as appropriate.
- If the actions of a rule update a field/property that is used by a previous rule in the list (one with a higher priority), reevaluate that previous rule and execute its actions as appropriate. Note that only those rules with a specific dependency are reevaluated.
- Continue the process until all rules in the RuleSet have been evaluated.

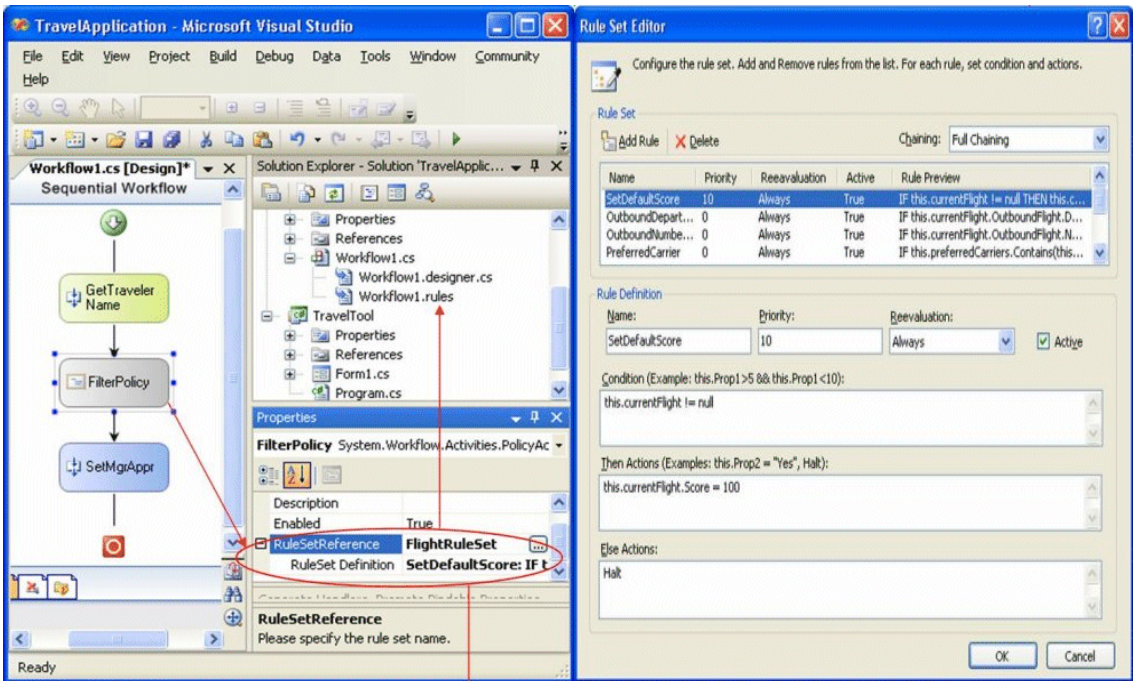


Figure 17: User interface to define rules(Willis 2008)

In forward chaining Rules the dependencies among rules are identified automatically by the rule engine based on the properties each rule read in its condition and write in its action. For more complex scenario (e.g.: aiming at increasing performance, preventing runaway loops), the developers are also allowed to change the chaining behavior. In such case, the developers must choose between sequential chaining or defining the chaining explicitly and whether the reevaluation is always done due to action of a rule or not. When RuleSet is executed, it can be traced and tracked through events which also can be logged in a log file for further analysis.

6. Business rules execution

6.1 Reasoning and inferencing

For the actual reasoning and execution of the business rules, but also for complex event processing and workflow execution we propose to use the Drools Business Logic integration Platform within the ebbits project. Drools is written in Java, but able to run on Java and .NET. Drools is now split into 5 modules, each with their own manual - Guvnor (BRMS/BPMS), Expert (Rules), Fusion (CEP), Flow (Process/Workflow) and Planner. Guvnor is our web based governance system, traditionally referred to in the rules world as a BRMS. Expert is the traditional rules engine. Fusion is responsible for the event processing side. Flow is the workflow module. The last module will not be discussed in this deliverable.

6.1.1 Drools Expert Rule Engine

The background of this systems lies in the research field Artificial Intelligence and more specifically in the field Knowledge Representation. In this field, one is concerned with formally representing knowledge in a knowledge base and to do reasoning with this knowledge base. This means, that we want to infer conclusions from the facts stored in the knowledge base and in this way, make implicit knowledge explicit. Drools is a Rule Engine that uses the rule-based approach to implement this reasoning procedure (Proctor, et al., Drools Expert User Guide, 2011).

Drools rule system is Turing complete, with a focus on knowledge representation to express propositional and first order logic in a concise, non-ambiguous and declarative manner (Proctor, et al., Drools Expert User Guide, 2011). Its main component is an inference engine that is able to scale to a large number of rules and facts. This engine matches facts and data against rules to infer conclusions which result in actions. This matching of new or existing facts against rules is called pattern matching. A rule has two-parts. In both parts first order logic is used as representation language. The structure of a rule is as follows:

when <conditions>

then <actions>;

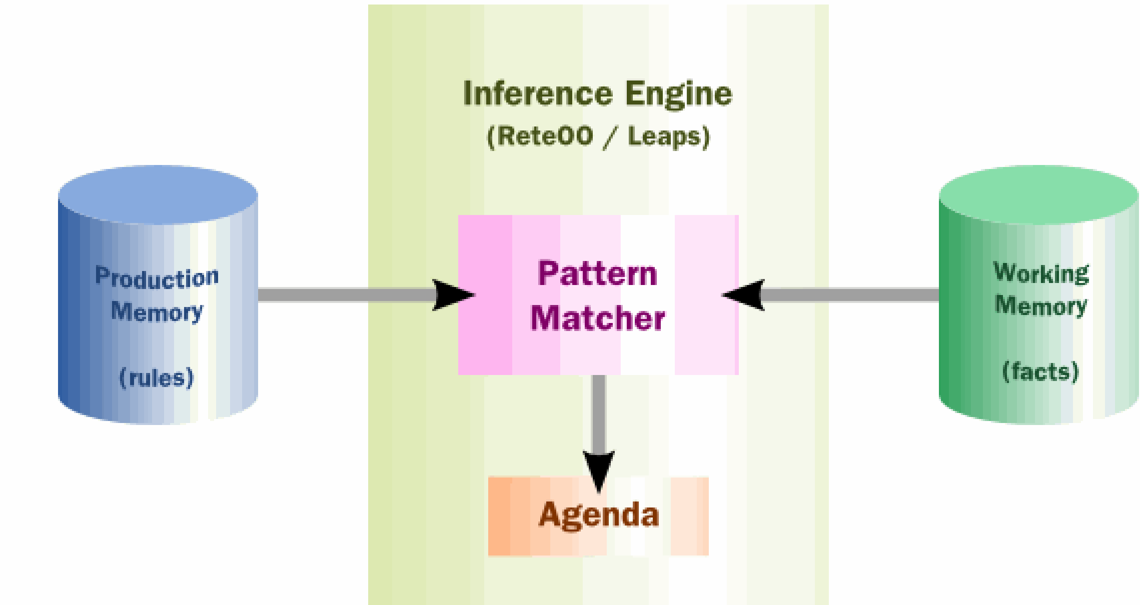


Figure 18: High-level view of a Rule Engine (Proctor, et al., Drools Expert User Guide, 2011)

The rules in Drools are stored in the so-called production memory as it is depicted in Figure 18. Facts instead are asserted into the so-called working memory where they may then be modified or retracted. The inference engine takes the facts from the working memory and does pattern matching against the rules in the production memory. In this process, it can happen, especially for a system with a large number of rules and facts, that many rules being true for the same fact assertion; these rules are then said to be in conflict. Another component of the inference engine, the Agenda manages the execution order of these conflicting rules using a conflict resolution strategy (Proctor, et al., Drools Expert User Guide, 2011).

There are two working modes for the inference engine in Drools. One is goal-oriented and it is called backward chaining, which is also the method the programming language Prolog is using. For this method of execution, a goal is tried to be proven. If this is not possible, the goal is broken down into subgoals which now in turn are tried to be satisfied. So far, Drools has only limited, experimental support for backward chaining (Proctor, et al., Drools Introduction and General User Guide, 2011). The other method of execution is "data-driven" and it is called forward chaining. For this method, which is reactionary, facts are being asserted into the working memory. This normally leads to one or more rules being concurrently true and firering. Which of the rules are then scheduled for execution is determined by the agenda component. The whole process is also shown in Figure 19 and be summarized as: its starts with a fact, which is then propagated and ends in a conclusion. At the moment, Drools is mainly a forward chaining engine (Proctor, et al., Drools Expert User Guide, 2011).

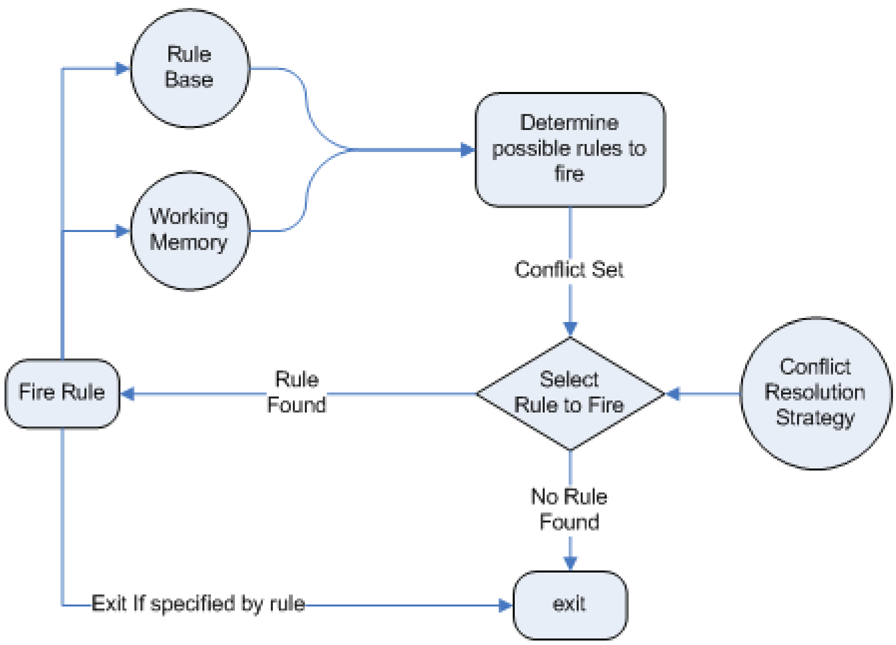


Figure 19: Forward Chaining (Proctor, et al., Drools Expert User Guide, 2011)

As there are other forms of knowledge representation and reasoning techniques, the question arises, why we should use a rule engine like Drools in ebbits. A rule engine has many advantages, the following mentions the most important ones (Proctor, et al., Drools Expert User Guide, 2011):

- Declarative Programming: Rules specify in a natural und understandable way, what should be computed. The question of how something is then computed is hidden inside the system.
- Logic and Data Separation: The data is in the domain objects, the logic is in the rules. This probably leads to a much easier maintenance and update of the logic as the general world knowledge is separated from specific data instances. This proposition can be especially true if the logic is cross-domain or it is a multi-domain logic.
- Speed and Scalability: The used algorithm provides very efficient ways of matching rule patterns to the domain object data. This holds especially, when the datasets only change in small portions as the rule engine can remember past matches.

- Explanation Facility / Understandable Rules: Drools Expert effectively provides an "explanation facility" by being able to log the decisions made by the rule engine along with why the decisions were made. Additionally, Drools allows for rules that are very close to natural language.

6.1.2 Drools Fusion Complex Event Processing

In the ebbits project, as it was already shown for the COMAU use case, where events from a water pump were recorded and aggregated in demo, there is a necessity to process events. The Drools Fusion module is able to process a single event, multiple atomic events or even hierarchies of correlated events, where a single event is defined as a significant change of state in the applications domain. Drools Fusion module is tightly integrated with the Drools Expert module.

Events are also handled by rules and are considered as facts with the following special semantics (Proctor, et al., Drools Fusion User Guide, 2011):

- Usually Immutable: Events are a record of a state change, i.e., a record of something that already occurred in the past which can't be changed in the present.
- Strong Temporal Constraints: Rules involving events usually require the correlation of multiple events and temporal correlations where events are said to happen at some point in time relative to other events.
- Managed Lifecycle: Events usually will only match other events and facts during a limited window of time, which is then automatically taken into consideration by the engine to manage the lifecycle of the events automatically.
- Use of Sliding Windows: All events have timestamps associated to them, therefore, it is possible to define and use sliding windows over them. Such a window allows for the creation of rules on aggregations of values over a period of time, e.g., average of an event value over 60 minutes.

```
declare S1Passed
    @role(event)
    @timestamp(timestamp)
end

declare S2Passed
    @role(event)
    @timestamp(timestamp)
end

/**
* Avaiblable operators are: after, before, during, coincides, includes, overlaps...
*/
rule "Passing from S1 to S2"
    when
        $s1 : S1Passed();
        $s2 : S2Passed( this after $s1 );
    then
        System.out.println("From S1 to S2, Speed '"
        + speedInMeterPerSecond($s1.getTimestamp(),$s2.getTimestamp())+"' m/s");
end
```

Figure 20:Example Event Processing Scenario for Ebbits

We have already devised a small, simple example scenario for the COMAU use case in Drools Fusion. Event processing is applied to a situation where a white body, which will be assembled to a car, is moving on a conveyer belt. At two places at the belt there is a light barrier, which fires an event, when the object on the belt passes by. Now we want to measure to speed and direction of car bodies moving on this belt in order to adjust to the most energy-efficient value. In Fusion, depicted in Figure 20 and very easy to understand, we declare the events and define rules, which determine the order of events and calculate the velocity. If the event `S1` is now followed after one second by the event `S2`, the expected result is given by the inference engine:

```
From S1 to S2, Speed '2.5' m/s
```

6.1.3 Drools Flow Workflow / Process Engine

Drools Flow is a workflow / process engine (Proctor, et al., Drools Flow User Guide, 2011). It is tightly integrated with the rules and the events module of Drools, not only system-wise, but also the API and the tooling support is unified. One of the tools, which is also integrated in the Eclipse Platform is shown in Figure 21. One can see in this screenshot a comfortable rule flow editor, which contains the graphical representation of the process and should allow also for non-technical people to easily model their process. The graphical model can also be processed in the underlying XML. The defined process model is automatically translated into a Java program, which can be executed like any other Java program and executes the specified process. The integration to rules and events happens in special nodes of the process, called rule task (in the picture in yellow) nodes. These nodes represent a set of rules / events, which are evaluated when the execution flow reaches one of these nodes. If one wants to include other, already defined workflows, the Drools Flow engine is generic enough such that different business process languages are supported, among them WS-BPEL, which was already described in Section 5.

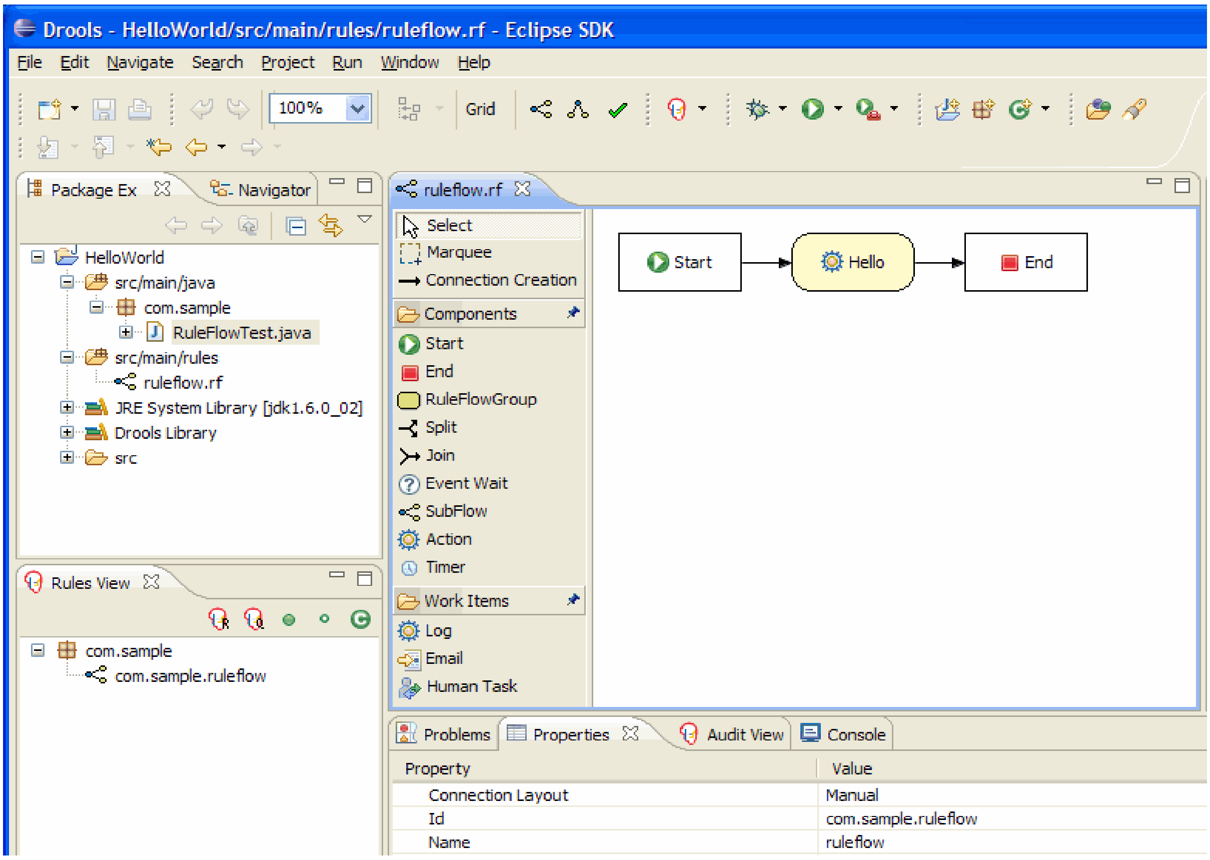


Figure 21:Graphical Tool Support for the Workflow Engine (Proctor, et al., Drools Flow User Guide, 2011)

6.1.4 Rules Repository

At the current stage of the project, we foresee more than one rule repository. As depicted Figure 24 in each device gateway, data fusion gateway and central ebbits node will have its own rule engine with a smaller and bigger number of rules. Accordingly, there should be also smaller or bigger rule repositories attached to each of these engines.

For the implementation of the rule repositories, we propose to use the Guvnor module of the Drools Business Logic integration Platform. This module is a business rule manager with user friendly interfaces which provides for multiple users of different skill levels to access and edit rules and processes (Proctor, et al., Guvnor Manual, 2011).

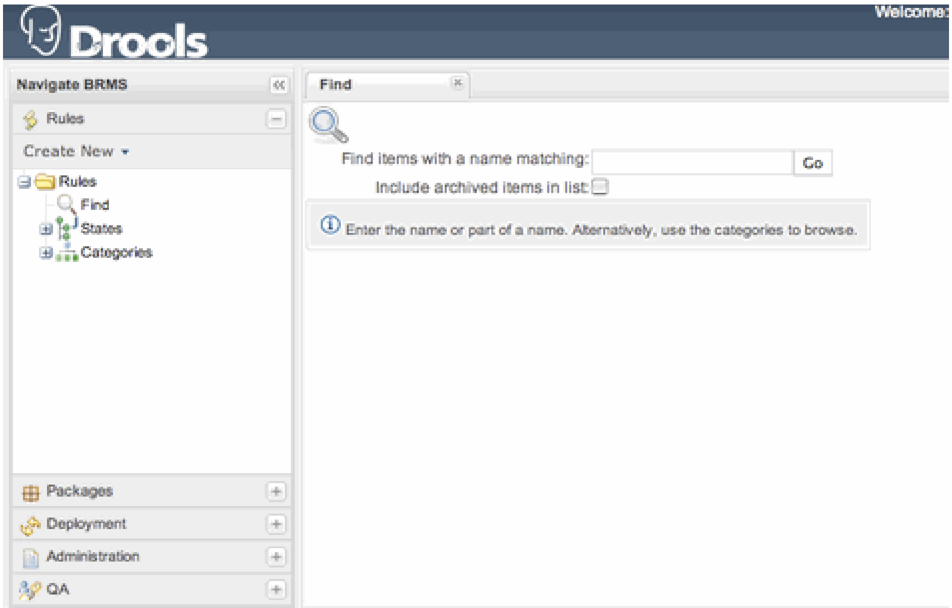


Figure 22: Drools Guvnor – a Business Rules Manager (Proctor, et al., Guvnor Manual, 2011)

Guvnor provides a WebDAV API such that the repositories can be accessed over http. Proper authentication will be required and therefore, access can be restricted to a predefined group of people within the ebbits project. Even a more fine-grained access mechanism can be selected, where for example domain experts, who do not program, can see and edit only with selected features. Guvnor allows using standard databases together with this business rule manager. Additionally, it has a version control system which provides for an undo capability (Proctor, et al., Guvnor Manual, 2011). Search in Guvnor is possible in two ways, one can use the hierachical structure of the packages or give own labels and use these tags for navigation.

For domain experts, which are not used to think in rules, the Drools Guvnor guided editor provides a wizard like way to create and edit rules through a graphical interface, which is depicted in Figure 23

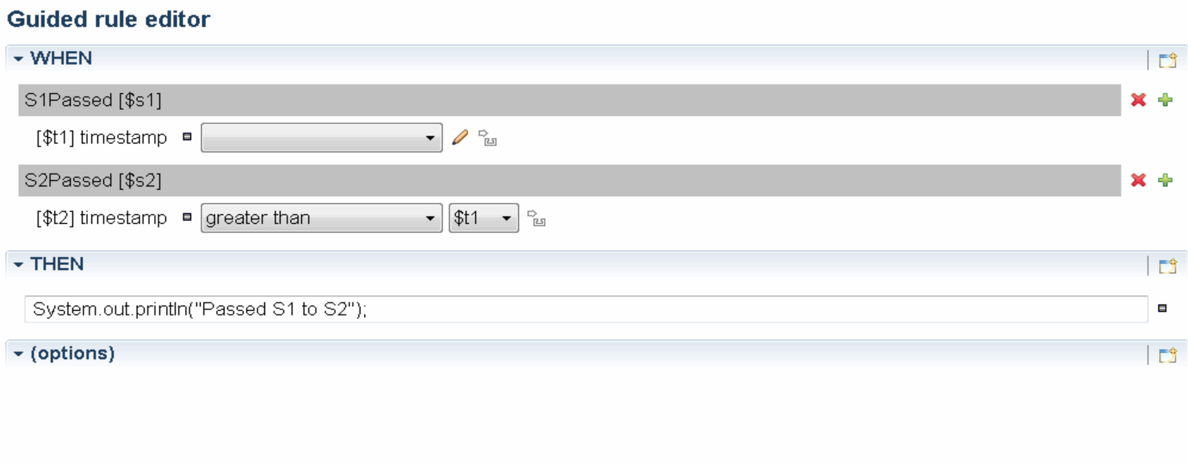


Figure 23: Graphical Rule Editor used for the Ebbits Scenario

6.2 Mapping to events and services

The initial event and data fusion architecture for ebbits is shown below. The model shows the flow of events and fusion of data through the main architectural components in the architecture.

In relationship to the ISA-95/IEC62264 framework, the enterprise – control boundary would reside between the ebbits central node and the device layer.

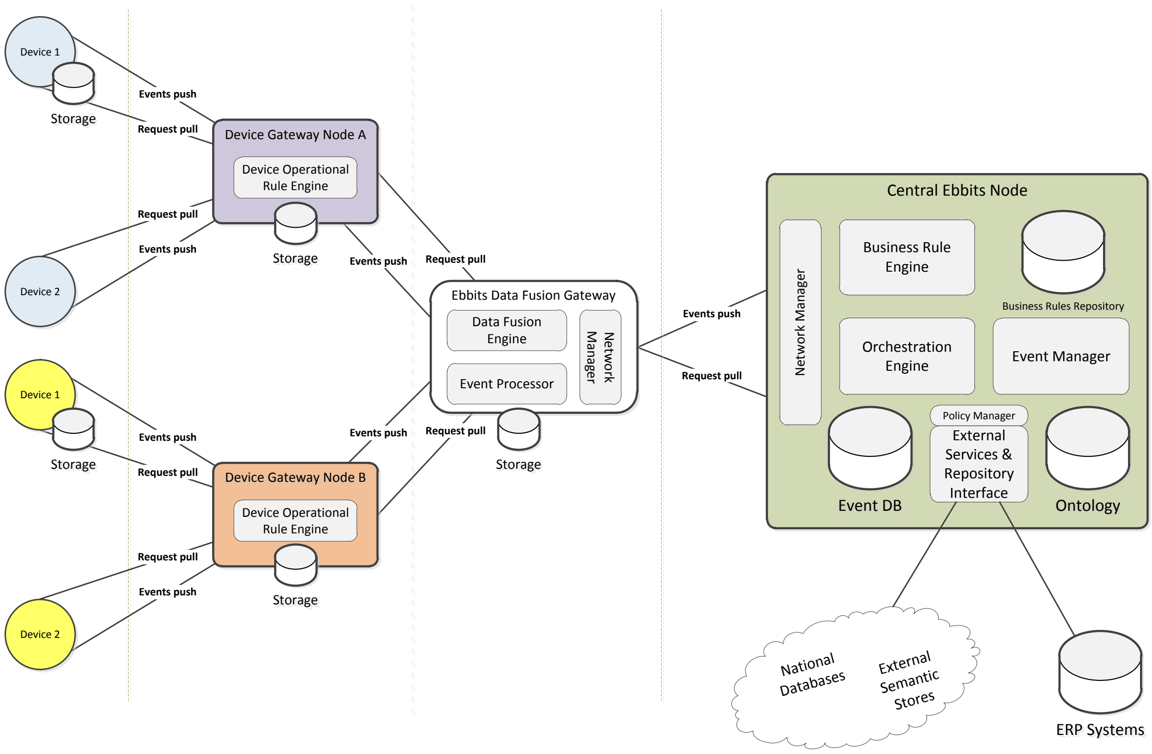


Figure 24: The initial ebbits Event and data fusion architecture

The business rule engine process a set of business rules defined by the ebbits platform user and describes the intended work flow organisation of the specific domain. Business rules are mapped to services via the orchestration engine. The rule engine uses its own repository. The business rule engine combines one or more application events into a business event which is forwarded to external business system. Business rules are maintained in a specific Business Rules Repository.

Services invocation is performed via the Orchestration Engine. The orchestration engine performs tasks on the request-pulls over the ebbits network. This can partly be configured by the business rule engine and partly manually through an interface. The engine will make use of the Device Ontology maintained on the ebbits central node.

The ebbits event manager handles events that are broadcasted throughout the network architecture. It deals with events processed on all levels in the ebbits platform and provides event management to external parties. The Event database store events in chronological order, and stores event logs for different event producers.

7. Conclusions and future work

The issue of investigating a suitable terminology for rules definition and interoperability in ebbits, was suggested in the Description of Work as a starting point, with reference to the ISA-95 approach. There are a number of questions to consider when evaluating applicability of a framework like ISA-95 to ebbits, e.g.,

- In general, how does this approach compare with the notion and vision of the IoPTS?
- How can the manufacturing and the traceability scenarios benefit or be supported?
- How can the sensor and monitoring perspective be supported? For example, energy consumption monitoring for manufacturing control function, and product tracing?
- And also, how could such a standards framework benefit from results of ebbits?

ISA-95 is a high-level model framework, developed long-before the emergence of the IoPTS. In this task and deliverable we have only looked at the first two parts of ISA, defining generic terminology and models for the ERP-MES interfaces. Being generic, it could be applied in ebbits, but in order to provide added value it would have to be specialized for the two scenarios, specifically for the traceability scenario. Further, the notion of an ERP-IoPTS interface is (naturally) not part of this framework, but could possibly be proposed as a result of the future ebbits developments.

The initial approach to rules representation and execution is based on the Drools framework, among the main arguments are that this framework provides us rich functionality, is well-established and supports multiple platforms and also multiple business processes modelling languages. The rules framework is to be used in combination with the event management and the service orchestration functionality built on the Hydra middleware (the latter provided in WP7).

Among the relevant issues to consider in the future design and implementation in WP6 are,

- The scalability of rule engines and repositories.
- The semantics and granularity of rules
- Storage and rule execution for constrained devices and gateways in the ebbits network.
- Further analysis, selection and specialization of the terminology for ebbits business rules management. This point is further studied in closed cooperation with work package 3.

8. References

Abiteboul, S. and V. Vianu (1991). "Datalog extensions for database queries and updates." Journal of Computer and System Sciences **43**(1): 62-124.

Battle, S., A. Bernstein, et al. (2005). "Semantic web services language (SWSL)." W3C Member submission **9**.

Bishop, B., & Bojanov, S. (2011). Implementing OWL 2 RL and OWL 2 Ql rule-sets for OWLIM. *OWLED 2011.* San Francisco.

Boley, H., B. Grosof, et al. (2002). "RuleML Design." from http://ruleml.org/indesign.html#The RuleML Design.

Boley, H., B. Grosof, et al. (2005). "RuleML Tutorial." Retrieved 10 June 2011, from http://ruleml.org/papers/tutorial-ruleml-20050513.html.

Chappell, D. and Associates. (2009). "The Workflow Way: Understanding Windows Workflow Foundation." Retrieved 10 June 2011, from http://msdn.microsoft.com/en-us/library/dd851337.aspx.

de Bruijn, J. et al (2005). Web Service Modeling Ontology (WSMO). W3C Member Submission. World Wide Web Consortium. Retrieved 10 June 2011, from http://www.w3.org/Submission/WSMO/.

de Bruijn, J. & Kopecky, J. (2008). D32v1.0 WSML/RDF. WSML Final Draft 2008-08-08. Retrieved 10 June 2011, from http://www.wsmo.org/TR/d32/v1.0/.

de Bruijn, J., Polleres, A., Lara, R., Fensel, D. (2005) WSML Deliverable D20.1 v0.2. OWL-. WSML Working Draft, May 15, 2005. Retrieved 10 June 2011, from http://www.wsmo.org/TR/d20/d20.1/v0.2/.

Dimitrov, M., Simov, A., Konstantinov, M., Cekov, L., Momtchev, V. (2007). WSMO Studio Users Guide, v. 1.27. Retrieved 10 June 2011, from http://www.wsmostudio.org/doc/wsmo-studio-ug.pdf.

DFKI. (2011). "rdf2java." Retrieved 10 June 2011, from http://rdf2java.opendfki.de/.

Furdik, K., Sabol, T., Hreno, J., Bednar, G., Lukac, G., Mach, M. (2011). A Platform for Semantically Enhanced Business Collaboration of Networked Enterprises. In: G. Fung (ed.), Introduction to the Semantic Web:Concepts, Technologies and Applications. Chapter 7. iConcept Press Ltd, Annerley, Australia.

Hepp, M., Roman, D. (2007). An Ontology Framework for Semantic Business Process Management. In: Proceedings of Wirtschaftsinformatik (1) 2007, pp. 423-440. University of Karlsruhe.

Horrocks, I., P. F. Patel-Schneider, et al. (2004). "SWRL: A semantic web rule language combining OWL and RuleML." W3C Member submission **21**: 79.

IEC/ISO (2003). IEC 62264-1 Enterprise-control system integration - Part 1: Models and terminology, IEC.

IEC/ISO (2004). IEC 62264-2 Enterprise-control system integration - Part 2: Object moel attributes, IEC.

Indihar-Stemberger, (2003) Indihar-Stemberger M., Popovic A., Vesna Bosilj-Vuksic V. (2003): Simulation and Information Systems Modelling: A Framework for Business Process Change. Proceedings 15th European Simulation Symposium

Martinek, P. & Szikora, B. (2008). Semantic Execution of BPEL Processes. In: Advances in Information Systems Development, pp. 361-367. Springer US.

Motik, B., Grau, B., Horrocks, I., Wu, Z., Achille, F., & Lutz, C. (2009). *OWL 2 Web Ontology Language* . Retrieved 06 03, 2011, from http://www.w3.org/TR/owl2-profiles/

Norton, B., Cabral, L., Nitzsche, J. (2009) Ontology-based Translation of Business Process Models. In: Proceedings of the 2009 Fourth International Conference on Internet and Web Applications and Services (ICIW '09), pp. 481-486. IEEE Computer Society, Washington, DC, USA.

(Perumpalath 2005) Perumpalath B. P., Labib A. W., N. (2005): Modelling Business Process: An Integrated Approach. Publication of the Portsmouth Business School

Proctor, M., Verlaenen, K., Tirelli, E., Rikkola, T., Lui, J., Anstis, M., et al. (2011). *Drools Expert User Guide*. Retrieved 05 31, 2011, from http://docs.jboss.org/drools/release/5.2.0.CR1/drools-expert-docs/html_single/index.html

Proctor, M., Verlaenen, K., Tirelli, E., Rikkola, T., Lui, J., Anstis, M., et al. (2011). *Drools Flow User Guide*. Retrieved 06 01, 2011, from http://downloads.jboss.com/drools/docs/5.1.1.34858.FINAL/drools-flow/html_single/index.html

Proctor, M., Verlaenen, K., Tirelli, E., Rikkola, T., Lui, J., Anstis, M., et al. (2011). *Drools Fusion User Guide*. Retrieved 06 01, 2011, from http://docs.jboss.org/drools/release/5.2.0.CR1/drools-fusion-docs/html_single/index.html

Proctor, M., Verlaenen, K., Tirelli, E., Rikkola, T., Lui, J., Anstis, M., et al. (2011). *Drools Introduction and General User Guide*. Retrieved May 30th, 2011, from http://docs.jboss.org/drools/release/5.2.0.CR1/droolsjbpm-introduction-docs/html_single/index.html#d0e47

Proctor, M., Verlaenen, K., Tirelli, E., Rikkola, T., Lui, J., Anstis, M., et al. (2011). *Guvnor Manual*. Retrieved 06 01, 2011, from http://docs.jboss.org/drools/release/5.2.0.CR1/drools-guvnor-docs/html_single/index.html

Roman, D., Scicluna, J., Fensel, D., Polleres, A., de Bruijn, J. (2006). D14: Ontology-based choreography of WSMO services. WSMO working draft, DERI, May 2006. Retrieved 10 June 2011, from http://www.wsmo.org/TR/d14/v0.4/.

Scheer, A. W., Thomas, O., Adam, O. (2005). Process modeling using Event-driven Process Chains. In: M. Dumas, W. M. van der Aalst, and A. H. ter Hofstede (eds.), Process-Aware Information Systems: Bridging People and Software Through Process Technology. Chapter 6. Wiley, October 2005.

Scribner, K. (2007). "Microsoft® windows® workflow foundation step by step."

Vittie, L. A. (2006). XAML in a Nutshell, O'Reilly Media, Inc.

Willis, J. (2008). "Introduction to the Windows Workflow Foundation Rules Engine." Retrieved 10 June 2011, from http://msdn.microsoft.com/en-us/library/dd554919.aspx.