Enabling the business-based
Internet of Things and Services

(FP7 257852)

D3.3 Business logic models

Published by the ebbits Consortium

Dissemination Level: Public

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

Document version: 1.1 Submission date: 01.03.2012

Document control page

Document file: D3.3 Business Logic Models.docx
Document version: 1.1
Document owner: TUK

Work package: WP3 – Enterprise Frameworks for Life-Cycle Management
Task: T3.2 Semantic business decision models
Deliverable type: R

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

Document history:

Version	Author(s)	Date	Summary of changes made
0.1	Viliam Vajda (TUK)	2011-04-20	TOC
0.2	Viliam Vajda, Tomas Sabol (TUK)	2011-05-28	Business Process Modelling
0.3	Jan Hreno (TUK)	2011-08-19	Semantic Web Services, SOA, Web Services sources added
0.4	Jan Hreno (TUK)	2011-10-08	Business process modelling guideline
0.5	Sigurjon Bjornsson (TNM)	2011-10-27	Business logic model in the Automotive Manufacturing Domain
0.6	Pietro Cultrona (COMAU)	2011-12-21	Business logic model in the Automotive Manufacturing Domain
0.7	Thomas Janke (SAP)	2012-02-14	Unified Service Description Language description
1.0	Jan Hreno (TUK)	2012-02-19	Version consolidated, Food traceability BPMN models added, version ready for internal review
1.1	Jan Hreno (TUK)	2012-03-01	Final version submitted to the European Commission

Internal review history:

Reviewed by	Date	Summary of comments
Jesper Thestrup (IN-JET)	2012-02-24	Approved
Martin Knechtel (SAP)	2012-02-27	Approved

Index:

1. Executive Summary ... 4
2. Introduction ... 5
2.1 Overview of the ebbits project ... 5
2.2 Purpose, context and scope of this deliverable ... 5
3. Business Process Modelling ... 6
3.1 Introduction ... 6
3.2 Business Process Modelling ... 6
3.2.1 The business process lifecycle ... 6
4. Guideline for Business Process Modelling ... 11
5. Business Logic Model in the Automotive Manufacturing Domain ... 12
5.1 Process identification ... 12
5.2 Process description ... 12
5.2.1 "Body in White" components description ... 12
5.2.2 Devices involved in the production process ... 13
5.2.3 Actors/Stakeholders identification and description ... 13
5.2.4 Data coming from the machine regarding the operating status ... 14
5.2.5 Example of production order coming from the upper supervisor level ... 16
5.2.6 Relevant data and its flow across the process ... 17
5.3 Energy aware process model ... 18
6. Business Logic Model in the Agricultural Domain ... 20
6.1 Process identification ... 20
6.2 Deliver animals to slaughterhouse ... 21
6.3 Slaughtering ... 23
6.4 Preparing meat for retail ... 26
7. Implementation of Business Processes using Web Services ... 28
7.1 Process related services ... 28
7.1.1 Semantic Web Services ... 28
7.1.2 The Unified Service Description Language (USDL) ... 30
8. Conclusions ... 33
9. References ... 34
10. Figures and Tables ... 35
10.1 Figures ... 35
10.2 Tables ... 35

1. Executive Summary

The ebbits project aims at providing semantic interoperability of services, things and people. This is done within existing business processes in selected user domains. Capturing a business process description in a formal way is the basic prerequisite for an efficient process updating. By modelling the business process, it is easy to identify where business activities can be optimised. Thus the business can fulfil its objectives more efficiently.

This deliverable provides a basic overview of the methodology of business modelling. Using this methodology, selected user domain processes are described using selected formalisms.

In Chapter 2 the ebbits project is briefly introduced and the purpose and scope of the deliverable is clarified in context of the ebbits project.

Chapter 3 provides an overview of the business process modelling methodology and business process lifecycle. Business Process Modelling is a basic technology to capture and share business processes. Business Process Lifecycle consists of the Design and Analysis, Configuration, Enactment and Evaluation phases ordered in a cyclic order. Phases describe, how a business process evolves during its administration and stakeholder interaction.

Chapter 4 is a guideline how to describe business processes for the ebbits project. The guideline serves to help users to capture business processes. These business process models are needed for a successful future integration of the ebbits platform within existing business processes. The guideline explains in a simple way, how to capture important information about the processes.

In Chapters 5 and 6, selected business processes from the automotive and the agricultural domains are described respectively. A formal model of these processes is proposed. The manufacturing scenario focuses on a body welding process description. Stakeholders in the process are analysed first. Data and information flows are identified secondly. A production process model is then presented. The agricultural domain focuses on a food traceability case. Process is identified first within interactions of different companies in a food chain of a beef production. Tabular information about actions, data and actors of selected sub processes are then proposed. Selected are the processes of animal delivery to a slaughterhouse, slaughtering the animal and preparing the meat for retail.

In Chapter 7, Web Services and their role in automation of business processes are described. USDL and Semantic Web Services are the selected technologies which are described. These are technologies needed to enable the flexibility of Web Service usage in business processes.

Chapter 8 concludes the deliverable. The conclusion compares the scope of two selected business processes. The automotive domain process focuses on low level production data with the overall description of inter business level decisions, while the agriculture domain process focuses on a business-to-business interaction with a brief overview of the intra business process. The ebbits platform should support both intra business and inter business process integration in the future.

Chapters 9 and 10 list references, figures and tables of the document.

The overall goal of the deliverable was to identify and describe business process elements from both user domains to be involved in the development of the ebbits platform. The deliverable will support the creation of, e.g., the deliverables "D4.7.1 Use-case driven semantic models (M24)", "D3.6 Business modelling concepts (M24)" and will influence architectural and implementation tasks in other WPs. The business processes introduced in this deliverable are further elaborated in the "D3.4 Business framework for online OEEE applications for production and energy optimisation (M12)" and "D3.5 Business framework for online food traceability in life-cycle perspective (M12)" which were prepared in parallel with this deliverable. The next ebbits intermediary SW demonstration will be based on the proposed business processes.

2. Introduction

2.1 Overview of the ebbits project

The ebbits project aims to develop architecture, technologies and processes, which allow businesses to semantically integrate the Internet of Things into mainstream enterprise systems and support interoperable real-world, online end-to-end business applications. It will provide semantic resolution to the Internet of Things and hence present a new bridge between backend enterprise applications, people, services and the physical world, using information generated by tags, sensors, and other devices and performing actions on the real world.

The ebbits platform will support interoperable business applications with context-aware processing of data separated in time and space, information and real-world events (addressing tags, sensor and actuators as services), people and workflows (operator and maintenance crews), optimisation using high-level business rules (energy and cost performance criteria), end-to-end business processes (traceability, lifecycle management), or comprehensive consumer demands (product authentication, trustworthy information, and knowledge sharing).

In the service oriented ebbits architecture providers and requestors of services must be able to communicate business logic information freely with each other despite heterogeneous communication and information infrastructures. To solve this, the ebbits service architecture is based on semantic resolution of services and objects allowing services to semantically discover, configure and communicate with each other services.

2.2 Purpose, context and scope of this deliverable

The purpose of this deliverable is to present technologies for capturing business processes and also to capture business processes for the ebbits project. This deliverable is one of the outcomes of task "T3.2 Semantic business decision models" in WP3.

Selected WP3 objectives that are relevant for this deliverable are

- Creation of vocabularies and description of business logic and business processes across various application domains
- Describe enterprise and management processes in the domain and in specific use cases

The deliverable is structured as follows:

Chapter 2 introduces the Deliverable and the scope in context of the ebbits project.

Chapter 3 provides an overview of the business process modelling methodology and lifecycle.

Chapter 4 is a guideline how to describe business processes for the ebbits project.

In Chapters 5 and 6, selected business processes from the automotive and the agricultural domains are described respectively. A formal model of these processes is proposed.

In Chapter 7, Web Services and their role in business processes automation are described. Selected technologies for enhancement of Web Services including semantic technologies are described.

Chapter 8 concludes the deliverable.

Chapters 9 and 10 list references, figures and tables of the document.

3. Business Process Modelling

3.1 Introduction

For the proper orchestration and execution of business services, a business decision model approach has been adopted. The business decision model is a representation of a specific situation of events and related services. Business decision models describe or mimic reality without dealing with every detail of it and are used by the event management subsystem to analyse a situation by combining strategic business priorities and goals with real-time information about the specific situation.

The business decision models are implemented using business logic, which describes the functional algorithms that handle and prioritise the exchange of information and interaction between services. The business logic provide models of virtual business objects (such as accounts, loans, itineraries) with physical business objects (such as location of physical assets and inventories, real-time energy consumption, machine states, etc.) and prescribes how the business objects services should be orchestrated and interact with one another to meet the overlying strategic business objectives and goals. It also enforces the routes and the methods by which services are authenticated, accessed and executed.

Business logic comprises the business rules that express business policies (such as channels, location, logistics, prices, and products) and workflows that are the ordered tasks of passing documents or data from one participant (a person or a software system) to another. Dynamic SOA based business processes operate on the "publish-find-bind" principle, where business processes may dynamically involve business partners and associated applications. The ebbits platform provides semantic interoperability to handle such dynamic situations involving service brokers to bind enterprises (including economic relationships) that have no prior business relationships between them. The main objective of this deliverable is thus to point to a semantic web based ontological framework, and related semantic service mediators, which allows for the proper orchestration and execution of business services based on the ebbits platform.

3.2 Business Process Modelling

Business Process Modelling (BPM) in systems engineering is the activity of representing processes of an enterprise, so that the current process may be analysed and improved.

In the literature we can find a lot of approaches in the area of business process modelling methodology. Some of the methodologies for business process modelling, especially developed by software companies, are strictly oriented to the system development; some of them are oriented to the top level of management and strategic planning of the company. There are also several. We will describe a technique for Business process modelling, which is a combination of similar proposals by (Aguilar-Saven 2004) and (Weske 2007).

3.2.1 The business process lifecycle

The business process lifecycle is shown in Figure 1. Process lifecycle consists of phases that are related to each other. The phases are organized in a cyclical structure, showing their logical dependencies. These dependencies do not imply a strict temporal ordering in which the phases need to be executed. Many design and development activities are conducted during each of these phases, and incremental and evolutionary approaches involving concurrent activities in multiple phases are not uncommon.

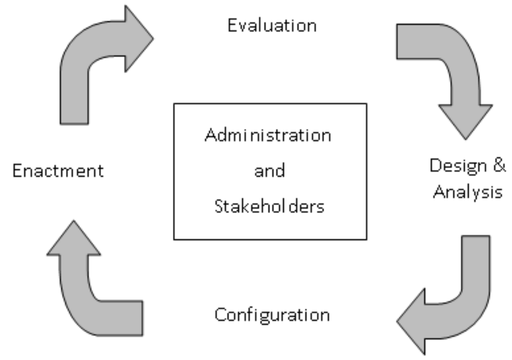

Figure 1: Business process lifecycle

There are numerous artefacts at different levels of abstraction in business process management scenarios that need to be organized and managed. A structured storage and efficient retrieval of artefacts regarding the business process models and information on business process instances as well as the organizational and technical execution environment need to be taken into account. In addition to business processes, knowledge workers with their organizational roles and skills, as well as the information technology landscape of the enterprise, need to be represented properly.

Phase 1 - Design and Analysis

The business process lifecycle starts by the Design and Analysis phase, in which surveys on the business processes and their organizational and technical environment are conducted. Based on these surveys, business processes are identified, reviewed, validated, and represented by business process models.

Explicit business process models expressed in a graphical notation facilitate communication about these processes, so that different stakeholders can communicate efficiently, and refine and improve these models.

Business process modelling techniques as well as validation, simulation, and verification techniques are used during this phase. Business process modelling is the core technical sub-phase during the process design. Based on the survey and the findings of the business process improvement activities, the informal business process description is formalized using a particular business process modelling notation.

The Survey phase is the phase relevant to individual business processes and the projects for realization of these processes. In this phase are defined the project goals, the project team is established, and information on the business process environment is gathered. Empirical studies based on interview techniques, and an analysis of available documentation, are conducted. The development of a domain ontology that provides a common understanding of the terms and concepts in the application domain is essential in this phase.

While the activities in this phase are centred on the business domain, the technical execution environment of the business process is also surveyed, since it might have implications on the realization of business processes.

The overall goals of the Survey phase are collection and organization of information regarding all aspects of business process management in addressed domain. The domain is determined by the organizational business process, of which one or more operational business processes should be identified and realized.

Particular goals are scoping the environment of the information gathering as well as in identifying business processes that can contribute to them. Technical infrastructure of the company should also be taken into account in this early phase in order to identify possible restrictions due to limitations of information systems that need to be integrated to implement the business process.

This phase starts with the setting up of the core team. In order to get an overview of the application process and the persons and organizational units involved, the core team starts with an initial survey of the environment that is addressed. Based on this information, additional persons are identified and invited to participate in the team.

The following activities involve a closer look at both organizational structure and documents that describe the application domain, including legislative regulations and business policies. The main activity of the survey phase is a detailed survey of the organizational and technical aspects. There are several ways to obtain the relevant information; interviews are a costly but effective method to get useful information. The information are documented and aggregated, using different formats, mostly in textual form.

The gathered information are analysed, consolidated, and represented as business process models. Created business process models serve as a communication basis for different stakeholders to improve the processes so that the operational goals can be realized.

Business process improvement not only addresses the actual process, but also the technical and organizational environment in which business processes is enacted. The technical environment can be improved so that service oriented approaches to the integration of external information systems are used that provide more flexibility than traditional enterprise application integration approaches. At the organizational level, new roles that require new skills and competencies might emerge to implement new business processes more efficiently and to provide better service to the customers.

Phase 2 - Configuration

Once the business process model is designed and verified, the business process needs to be implemented. It can be implemented by a set of policies and procedures that the employees of the enterprise need to comply with. In this case, a business process can be implemented without any support by a dedicated business process management system.

In case a dedicated software system is used to realize the business process, an implementation platform is chosen during the configuration phase. The business process model is enhanced with technical information that facilitates the enactment of the process by the business process management system.

The system needs to be configured according to the organizational environment of the enterprise and the business processes enactment of which it should control. This configuration includes interactions of the employees with the system as well as integration of the existing software systems with the business process management system.

The Platform Selection

This phase uses the business process models as well as information on the technical and organizational environments of the business process to select a technological platform on which the business process will be enacted.

A wide variety of platforms might be suitable for implementing business processes, including automated platforms such as enterprise application integration middleware, service-oriented architectures realizing system workflows, or workflow management systems to support human interaction workflows.

Implementation

This phase should involve the development of prototypes, and invite feedback by the knowledge workers on the design of these applications. These aspects are also covered in the configuration phase of the business process lifecycle.

Depending on the particular technology used, concrete data type definitions are set up, as are control flows between activities and the technical realization of the activities, by the integration of existing application systems. An important activity in this phase is concerned with tool integration, i.e., the integration of external applications. Depending on the support provided by the selected system, tool integration may require considerable coding and extensive testing.

The Implementation and Testing phase begins with the implementation of the business process models using the selected platform, for instance, a workflow management system.

Testing

Extensive testing is required to make sure that the technical solution effectively realizes the business process. It is important to also study non-functional aspects, such as performance and robustness, so that problems related to them do not emerge only after the system has been deployed.

Testing comprises the two sub-phases lab simulation and field testing. The overall goal of the testing sub-phase is to obtain information about the technical stability and the usability of the solution in the target environment.

Lab testing also involves simulation, where first the simulation goals are defined, followed by the design of the test scenarios. This includes the definition of the amount of data and workflows as well as the relevant business tasks; time restrictions are also specified in the test scenario.

Deployment

During the Deployment phase, the implementation of the business process is deployed in the target environment. Technical aspects need to be taken into account to make sure that the operations will not suffer during deployment.

Organizational aspects also need to be taken into account, for instance, training of the knowledge workers. Depending on the particular enactment environment of the process and the skills and expertise of the knowledge workers, these activities should start at an earlier stage in time, potentially after the first stable prototype implementation is available.

Phase 3 - Enactment

Once the system configuration phase is completed, business process instances can be enacted. The process enactment phase encompasses the actual run time of the business process. Business process instances are initiated to fulfil the business goals of a company. Initiation of a process instance typically follows a defined event, for instance, the receipt of an order sent by a customer.

The business process management system actively controls the execution of business process instances as defined in the business process model. Process enactment needs to cater to a correct process orchestration, guaranteeing that the process activities are performed according to the execution constraints specified in the process model.

A monitoring component of a business process management system visualizes the status of business process instances. Process monitoring is an important mechanism for providing accurate information on the status of business process instances. This information is valuable, for instance, to respond to a customer request that inquires about the current status of his case.

During the business process enactment, valuable execution data is gathered, typically in some form of log file. These log files consist of ordered sets of log entries, indicating events that have occurred during business processes. Start of activity and end of activity is typical information stored in execution logs. Log information is the basis for evaluation of processes in the next phase of the business process lifecycle.

In the Operation and Controlling phase of the methodology, the business process application runs in the target environment. Valuable execution information is gathered, which is useful in improving the process in an evolutionary way. This phase is associated with the enactment phase of the business process lifecycle.

These directed arcs do not specify a strict sequential ordering. The methodology is iterative and incremental. By gathering knowledge about the business processes and their environment, new questions and issues emerge that need to be taken care of in the next iteration.

The Operation and Controlling phase comprises the sub-phases installation and run time as well as a setup activity, in which the technical environment for the deployment of the workflow application is provided.

Phase 4 - Evaluation

The evaluation phase uses information available to evaluate and improve business process models and their implementations. Execution logs are evaluated using business activity monitoring and process mining techniques. These techniques aim at identifying the quality of business process models and the adequacy of the execution environment.

4. Guideline for Business Process Modelling

In this chapter a guideline for business process capturing in ebbits is presented. User partners together with ICT specialists will describe business processes of both user scenarios in Chapters 5 and 6. These will be based on scenarios proposed in DOW, and single processes will be identified, where the ebbits platform can be used. Business process models of the current processes together with textual description of these are the expected result.

This deliverable will concentrate only on Design and Analysis phase of the process modelling.

Design and Analysis phase

The phase will start with a Survey, where user scenarios and user requirements from the WP2 will be analysed for requirements related to business processes. User partners will provide additional information for the process description as follows:

1. Process identification
    - Description of process elements
        - How the process starts
        - Who initiates it
        - What happens during the process
        - How the process ends
        - Who are the actors (people, their position, departments, external services...)
        - What documents, data are used, how are they sent
        - What systems, SW, HW are used
    - In the processes one should distinguish operational and managerial levels
    - All actors (internal and external) should be in the process. If anything comes from an external business it should be described (suppliers, analysts, systems)
2. Process description (more formal, e.g. bullet style lists)
    - Process participants, actors (persons, systems with detailed description of duties and responsibilities)
    - Events, activities of the process (logically defined steps of the process activities and events)
    - Information and message flow
    - Sequence flow
    - Interaction with external actors (if applicable)
    - Input and output data description (with description of data flow and data storage)
3. Process modelling
    - user partners + developer partners – preferably BPMN (Business Process Modelling Notation) schemes
    - Based on the survey and the findings of the business process improvement activities, the informal business process description is formalized using a particular business process modelling notation

Configuration phase

The configuration phase will be conducted in the implementation tasks WP4 - WP9.

Enactment phase

This phase will be done in WP10 and WP11.

5. Business Logic Model in the Automotive Manufacturing Domain

5.1 Process identification

As already described in Deliverable "D2.1 Scenarios for usage of the ebbits platform" the automated car production inside a plant is normally divided into several areas, each of them is dedicated to specific activities. They can basically be simplified in the following parts of the manufacturing process:

- Power train plant: in this area the engines are machined and assembled.
- Body welding (also called 'body in white shop'): where the body of the car is assembled and welded.
- Painting shop: the area where the body in white is prepared for painting and finally painted and cocked
- Final Assembly: where the painted body is fitted with engine, suspension, trim and all the other parts.

The process chosen to be analysed is the "body welding" because it is an automated process that can be easily simplified, monitored and studied.

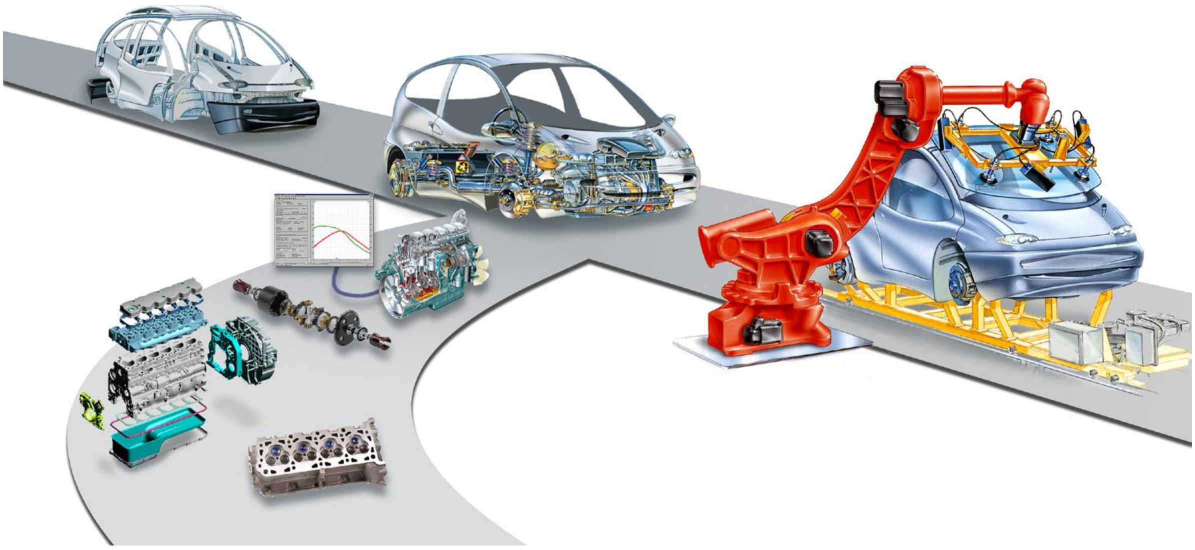

Figure 2: Automated car production

5.2 Process description

The scope of this section is to analyse the "AS-IS" situation regarding the process chosen as representative for the manufacturing environment.

The process description will be focused on several aspects:

- Body-in-white components description
- Devices involved in the production process
- Actors/Stakeholders identification and description
- Data flow across the process

5.2.1 "Body in White" components description

As already described, the body welding process consists in the assembly of all the metal parts that compose the body of the car. This process is done basically though the spot welding technique.

The main sub-groups that compose the car body are, as shown in the picture below, the front frame, the middle and rear floor, the underbody, the roof, the body sides and the closures.

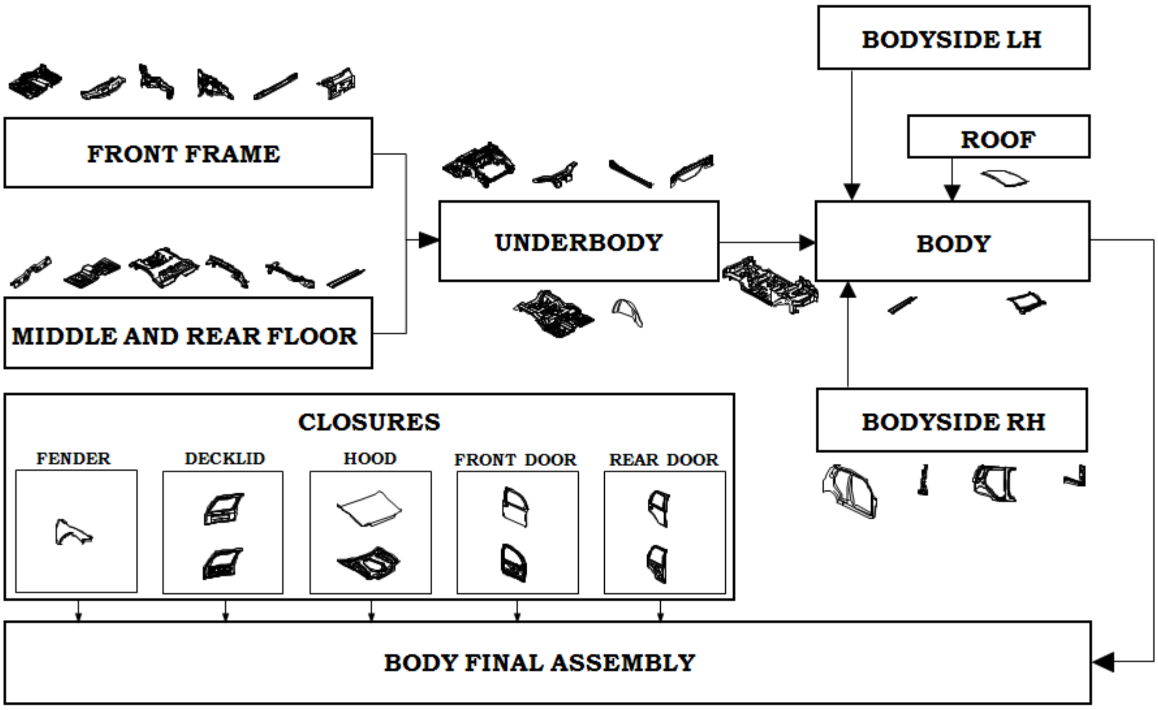

Figure 3: Car body components

5.2.2 Devices involved in the production process

Going more in detail and analysing a single production cell, it is possible to see that the assembly process is done by mechatronic devices controlled by PLCs and running a particular software that consists in a specific sequence of repetitive operations. The main components involved in the production process have been widely described in "D8.4 Integration of physical world in manufacturing" Section 3.2, but they can be summarized as follows:

- PLC, in charge of coordinating all the mechatronic devices.
- Robot, used for handling and welding the elements
- Servo Drives, that interface the servomotors to the PLC and control their movements
- Fieldbus, which is the network that connects all the devices distributed on the field
- Valve packs, composed by one or more electromechanical valves typically used for actuating clamps and locating pins.
- Sensors, used for collecting feedbacks from the field, they are typically inductive, capacitive and photoelectric.
- HMIs, computers used for interfacing the machine with the operator allowing manual movements and displaying the machine status.
- Scada systems, computers used for an high level supervisory control and data management regarding production orders and data logging.

Chyba! Neplatné hypertextové prepojenie.
Figure 4: Body welding

5.2.3 Actors/Stakeholders identification and description

The actors involved in the process have been analysed inside Deliverable "D2.6 Validation framework" in section "2.2 Stakeholder analysis". In the mentioned section they have been classified into four categories:

1. Concept owners that license the right to use the ebbits to industrial enterprises or service providers.

2. Service providers that are organisations that establish the commercial ebbits platform and offer ebbits applications to enterprises or organisations in the forms of Software as a Service, Platform as a Service or Infrastructure as a Service.
3. Business partners that are companies and organisations concretely taking part in the project and benefiting from its outcomes. Their business processes are ubiquitously interwoven and interacting with the aim of optimising and executing the business strategies.
4. End users that are all the people and entities working and interacting with the applications and devices, typically on a daily basis.

The following table is an extraction from "D2.6 Validation framework" and summarizes the classification operated.

Stakeholder Category	**Entities in Automotive Manufacturing**
Concept Owners	Providers (e.g. ebbits partners)
Service Providers	Providers (e.g. ebbits partners)
Business Partners	Manufacturing plant managers* Suppliers Customers Providers (*When considered organisations)
End Users	Machine operators Line supervisors Area supervisors Maintenance crew Manufacturing plant managers** (** When seen as individuals working for the enterprise)

Table 1: Automotive scenario stakeholders

5.2.4 Data coming from the machine regarding the operating status

Data reported by the PLC as "diagnostic standard" are:

- Object Failures-alarms (section, sector, station, machine)
- PLC status of the section (and other requested objects)
- Cycle Time
- Monitoring time counters

Faults / PLC Alarms

The failures/alarm strings will have as first reference the alarm type, and then the link to the line layout (identification of the object) with the identifier of the station to which it belongs (e.g. M/A OP120 fault ....), the word/bit that generated the fault and lastly the description.

<u>IMPORTANT: each alarm must contain the "PLC symbolic" referred to the alarm.</u>

The format of every alarm string must be of type defined in Table 2.

MSG Type	Blank	STATION	Blank	SIMBOLIC	Blank	MSG Description	REF.
3 char	1 char	5 char	1 char	Max 24 char	1 char		[Word].Bit

Table 2: Failure/alarm

M/A ST090 090MCP01-QF12:14 24VDC HMI OK *dbxx.x.x*		
M/A ST090 090MCP01-QF14:14 24VDC ROBOT LATO SINISTRO OK *dbxx.x.x*		
M/A ST090 090MCP01-QF15:14 24VDC ROBOT LATO DESTRO OK *dbxx.x.x*		
ALARM	MSG	COLOR
	TYPE	
Fault	M/A	Red
Time-out	M/T	Magenta
Pre-alarm	M/P	Yellow
Guided operation	M/G	Light blue (cyan)
Parity error	M/E	Red
Emergency	EMG	Red
Voluntary stops	M/V	Orange
Monitor status change	M/M	Production Green, No Loading Yellow, .., ……

Table 3: Alarm example

Status Objects: Section\Station

The monitored object is the section\station. The statuses monitored are listed in priority order from highest to lowest:

- Residual (1)
- Stop/Failure/Breakdown (2)
- Emergency (3)
- Electrodes changing (4)
- Stoppage for check & consistency (5)
- Timeout (movement) (6)
- No loading (7)
- Unloading impossible (8)
- Tooling (*) (9)
- Tooling excess (*) (10)
- Line Side Stop (11)
- Manual (12)
- No cycle started (13)
- No External enabling (14)
- Production (15)

Status Objects: Machine\Component

- Fault (1)
- Production (2)
- Other (3)

Cycle Times

Cycle Time is defined as the time required by an object (station, machine...) to complete the task for CSS or equivalent. The countdown starts at the beginning of the operation and stops whenever the status system is different from Production. It restarts at the restore of Production status, deducted of the load operator time. The cycle time is reset for each piece finished.

The cycle times are counted in a specific area on the PLC for each station-CSS, according to mapping listed in Annex A of this document.

At the end of each cycle a "one-shot" of the last CT value must be copied in the PLC dedicated area.

Counters

Monitoring is the activity that the Line PC connected to the system carries out that allows the time elapsed in a work ship to re-start in the various system operation statuses.

In the case in which a Line PC supervises two or more PLCs for a single line section, the monitoring software will be on only one PLC (called master).

Monitoring also allows the production trend to be known in terms of:

- ELEMENTS PRODUCED BY SPECIALTY APPROVED AND REJECTED
- ELEMENTS PRODUCED BY COLOUR
- STEPS MADE
- ACTUAL CYCLE TIME PERIODS OF EACH STATION

5.2.5 Example of production order coming from the upper supervisor level

The diagram on Figure 5 explains the order-flow from the input in the system to the extraction from the list by the client.

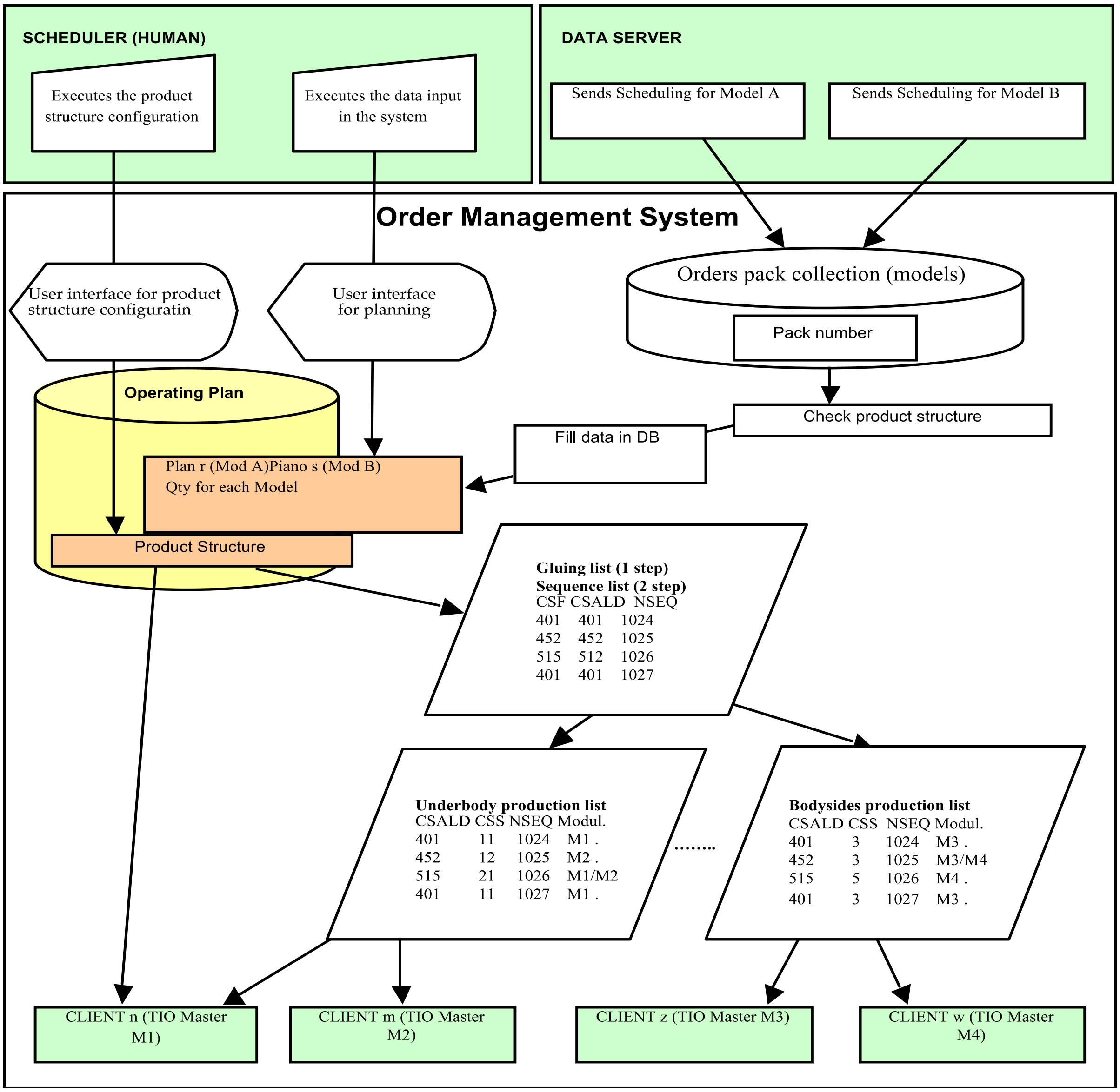

Figure 5: Order flow

There are 4 levels of Users able to access to the system:

Guest: only read permissions inside the application
Scheduler: read/write permission in the scheduling system
Administrator: full access inside the whole system
Operator: only read production lists on the client level

The data format of each order flowing inside the plant is customized on the Customer's request (Car Manufacturer).

An example of data is described in the following table, each element data is composed by 24 WORDS:

Array Detail	
WORD [xx]	**DESCRPTION**
0	Consistency index (Ic)
1	Progressive Shipping number (Ps)
2	Acknoledge request
3	Sequence Number (NSEQ)
4	Code speciality welded body (CSALD)
5	Family Code (CF)
6	Code specialties subgroup (CSS1)
7	Code specialties subgroup (CSS2)
8	Not used and reserved for N_TAG / N_SKID
9	Extraction Target (DE)
10	Extraction Reason (CE)
11	Not used and reserved for FAULT
12	Not used and reserved for LINE OF ORIGIN
13	Not used and reserved for COMBINATION FLAG / PRIORITY FLAG (FA/FP)
14	Not used and reserved for ROBOT EXCLUDED (R1 ÷ R16)
15	Not used and reserved for ROBOT EXCLUDED (R17 ÷ R32)
16	Not used and reserved for FRAME N °
17	Not used and reserved for FRAME N °
18	Not used and reserved for SUBJECT
19 ÷ 22	Reserve
23	Consistency index (Ic')
24	Index previous message (Ic_OLD)

Table 4: Order data format

5.2.6 Relevant data and its flow across the process

The process-related data coming from the field allows controlling and monitoring the manufacturing process.

Control consists of the execution of the production plan. Monitoring involves all those data and information analysis as e.g. in parts produced, destructive tests and not destructive tests, scrap parts, parts produced for measurements and their results (ok or out of tolerance). Another important data usually collected is the cycle time, this parameter is important to plan the production and for the knowledge of the maximum production per shift.

Other important parameters collected in order to improve the manufacturing plant efficiency and to reduce the waste are related to the stops of the machine: part of PLC Program sends to a central remote server all data about the stops distinguishing e.g. not-loads, not-unloads, faults and safety stops.

Not-loads means that the production line is ready for produce, but the elements to be worked are missing.

The not-unloads means that the line is full and it is not able to deliver the elements to the next line for unforeseen problems.

The faults include each process-related problem that does not allow the correct function of the machine.

The safety stops are all the stops due to safety problems for the operator (e.g. safety stop push buttons, doors for accessing the line, light scanners and light curtains, etc.).

All data described are typically made available from the PLC to an upper level controlling device in charge of collecting and storing information from the production plant. The aim of this function is to collect data from the shop floor and track the production trends in order to make production plans and specific maintenance operations when required or when the production is down.

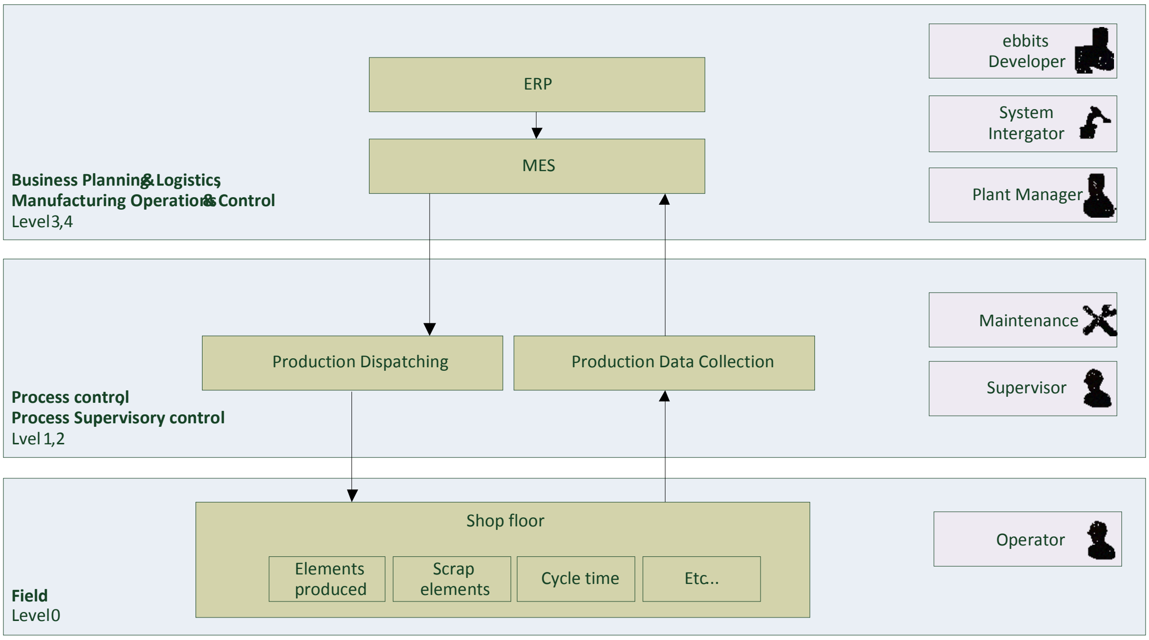

Figure 6: Car manufacturing production process

Important information is actually missing in the previous description and it is the energy consumption related to the production process, e.g. the whole power consumption of the machine or simply the power consumption of the welding circuit and the water used by the cooling system of a single welding robot.

5.3 Energy aware process model

When the plant manager has to prepare the production plan or the developer/system integrator has to build a new plant, the most important parameter considered is the production volume expected in a particular frame-time. This information is collected from the machine, if the production plant is existing, or it is the output coming from a dynamic simulator for a production plant under development.

An important aspect to be analysed is that introducing energy-data is as an additional input inside the simulator, the output provided could be e.g. a "production plan-energy optimised" that considers the different energy costs during the shifts or the days of the week, or that gives the CO2 footprint of each element produced. The following picture introduces the energy concept on the diagram shown previously.

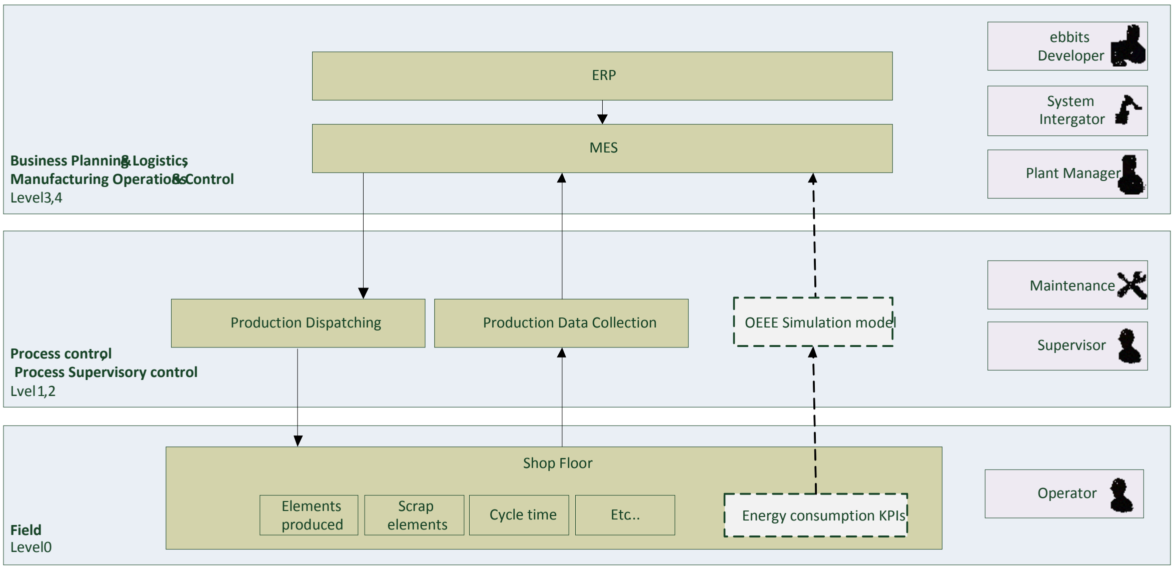

Figure 7: Car manufacturing production process aware of energy consumption.

6. Business Logic Model in the Agricultural Domain

6.1 Process identification

Figure 8 is taken from "D2.1 Scenarios for usage of the ebbits platform" and shows the overall processes in the traceability scenario. The process of producing crops and animals for consumption is long and can often take up to several years for the larger animals. There are also many actors and many steps it this whole process. The processes also vary by countries, companies and what type of animal is processed.

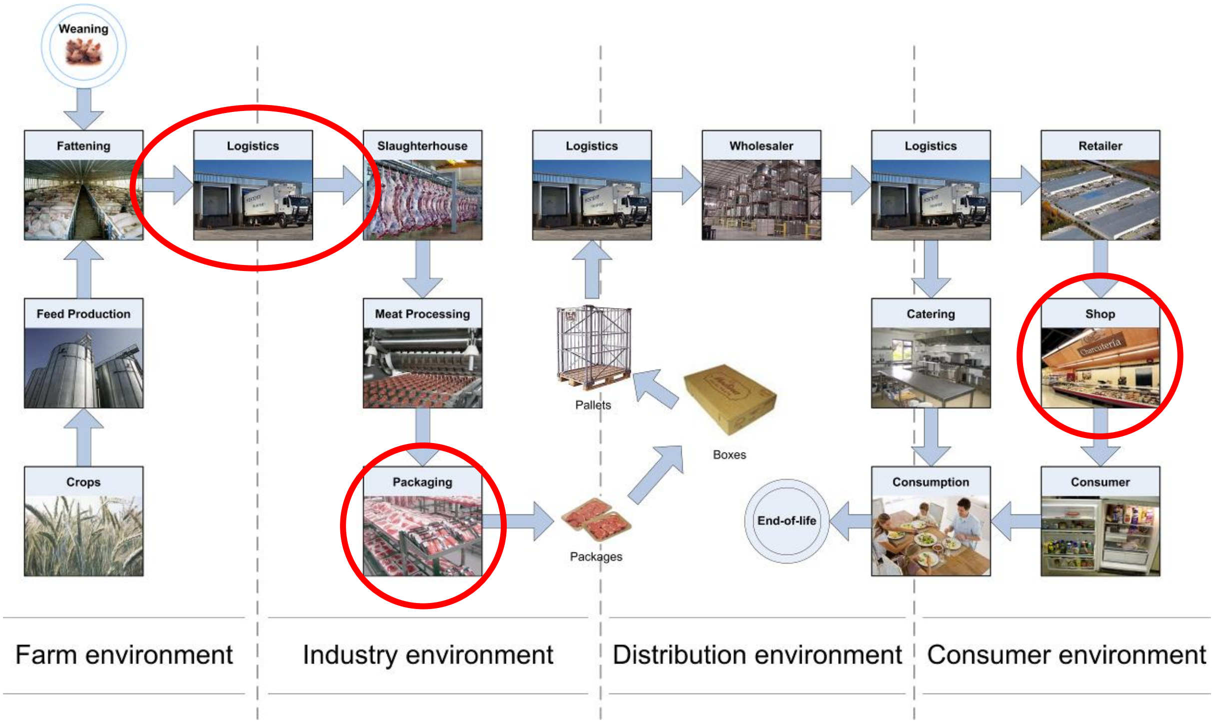

Figure 8: From farm to fork

The processes chosen to be described here are marked with red circles and were chosen because of their importance for the traceability scenario. The process description provided here is typical for handling of cattle/beef in Denmark and provides some specifics to support traceability. The first process is the delivery to the slaughterhouse and shows the transition of information between two important actors. The second process handles the slaughtering of the animal, it is in the slaughterhouse many of the quality information are generated. The third process describes the packaging of meat, a step where the traceability is usually lost.

6.2 Deliver animals to slaughterhouse

This is the final process for the farmer. When he has finished fattening a portion of animals they are sold to the slaughterhouse for slaughtering. This means that they are to be delivered from the farmer to the slaughterhouse at a time when the slaughterhouse is prepared to receive them.

<table>
<tr><td>Actors:</td><td>Farmer, Transport Company, Slaughterhouse.</td></tr>
<tr><td>Input data:</td><td>Requests from butchers and slaughterhouse</td></tr>
<tr><td>Initiated by:</td><td>Farmer.</td></tr>
<tr><td>Actions:</td><td>
<table>
<tr><td></td><td>Purpose:</td><td>Action:</td></tr>
<tr><td>1</td><td>The farmer informs the slaughterhouse about the number of animals he expects to deliver.</td><td>The farmer using feedback from the slaughterhouse and the butchers identifies the animals ready for slaughtering. The feedback from the slaughterhouse is usually requests for number of animals, specific breeds needed and historical data for optimal weight. The animals are identified by unique numbers on ear tags.</td></tr>
<tr><td>2</td><td>Selection of animals that have the size to be delivered.</td><td>Animals selected for slaughtering are taken out of the pen and gathered in a delivery pen.
When the transportation vehicle arrives the animals are taken from the delivery pen and herded to a truck.
Animals are manually counted during loading.</td></tr>
<tr><td>3</td><td>Transfer from truck to slaughter</td><td>The animals are unloaded into a waiting area at the slaughterhouse
If there is no delivery statement the animals are not sold as branded meat but as normal meat.</td></tr>
</table>
</td></tr>
<tr><td>Output data:</td><td>Number of animals, animals IDs</td></tr>
</table>

Table 5: Deliver animals to slaughterhouse - process description

Submitter	**Receiver**	**Transmission method**	**Information details**
Farmer	Slaughterhouse	Telephone	Notifying the slaughterhouse of an incoming delivery. • Number of animals • Type of animals • Race • Organic or not • Estimated weight • Age
Slaughterhouse	Farmer	Email	Notifying the farmer of expected delivery time. • Date and time for receiving animals
Slaughterhouse	Transport company	Telephone	Ordering transportation for animals. • Date and time for picking up the animals • Number and type of animals • Delivery location
Farmer	Slaughterhouse	Paper	Delivery statement. • List of Animal ID numbers

Table 6: Deliver animals to slaughterhouse - information flow

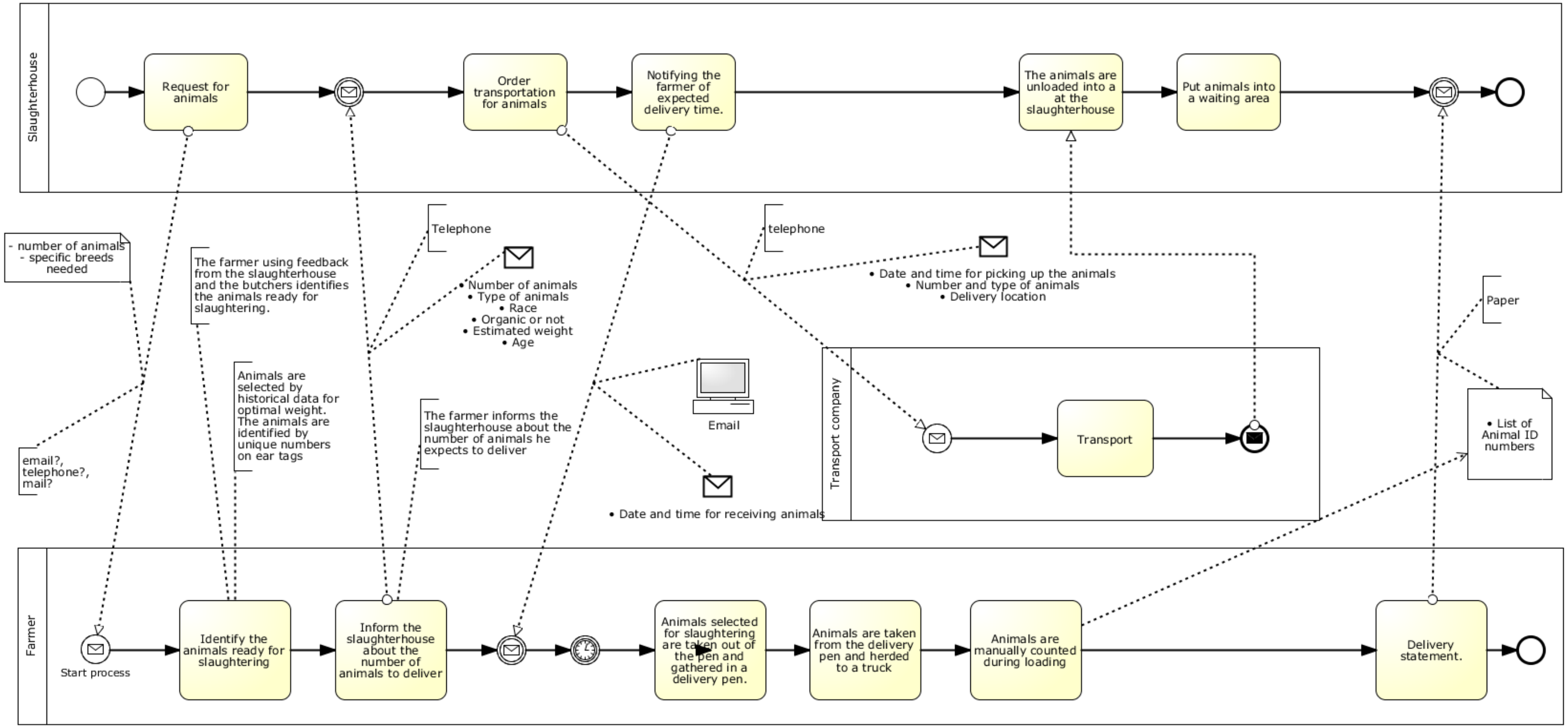

Figure 9: Deliver animals to slaughterhouse process model

6.3 Slaughtering

The process describes the processing of live cattle into prime cuts. The animals arrive from the farmer and the prime cuts are large pieces delivered to retail stores and butchers for consumer packaging. In this process many of the information about the quality of the meat is generated.

Actors:	Slaughterhouse.		
Input data:	List of Animal IDs.		
Initiated by:	Slaughterhouse.		
Actions:		Purpose:	Action:
	1	Slaughtering the animals	After being in a holding pen. The animals are led to the slaughter station. They are stunned, hung upside down and a vein cut open to let the animal bleed to death. The animal ID is read from it ear tag and registered in the ERP system. A tag is also placed on the carcass with identification information. If animals are not on the delivery statement from the farmer they are not sold as branded meat.
	2	Removing non meat parts	When the animal is dead the head, internals, feet and hide are removed
	3	Veterinarian inspection	Until the veterinarian inspects the carcass all its parts are kept together. The internals are kept in a tray near the carcass. Organs and other aspects of the animal are inspected and based on this information and observations from when the animal was alive are used to identify sick animals that are discarded.
	4	Cooling	The carcasses are cooled and stored before further processing.
	5	Quality inspection	The meat is rated based on colour, form and marbling. The meat is also tasted and only meat that gets high enough score on all tests is sold as traceable meat. This information is stored in the slaughterhouse ERP system. Meat that does not pass the quality tests is sold as unbranded meat.
	5	Prime cuts (optional)	The slaughterhouse cuts the carcass into prime cuts.
	6	Whole sale packaging	The prime cuts are packaged so that the whole or halve carcasses are packaged individually. This insures that the traceability available for the butcher. The package is then labelled with the animal number. With the package the slaughterhouse sends stacks of labels with animal ID, so the butcher can place it on the retail packages so the customer can trace their meat.
Output data:	Weight, Meat percentage, animal condition, price, Quality measures (colour, form, marbling)		

Table 7: Slaughtering - process description

Submitter	**Receiver**	**Transmission method**	**Information details**
Slaughterhouse	Farmer	Paper/Mail	Feedback on state of animals and invoice Number of animals Slaughter weight Base prices Meat percentage

			Quality metrics Extra price based on quality
Slaughterhouse	Registration authority	Web interface	Updating the central database with information on date of death Slaughter date of animal Animal ID

Table 8: Slaughtering - information flow

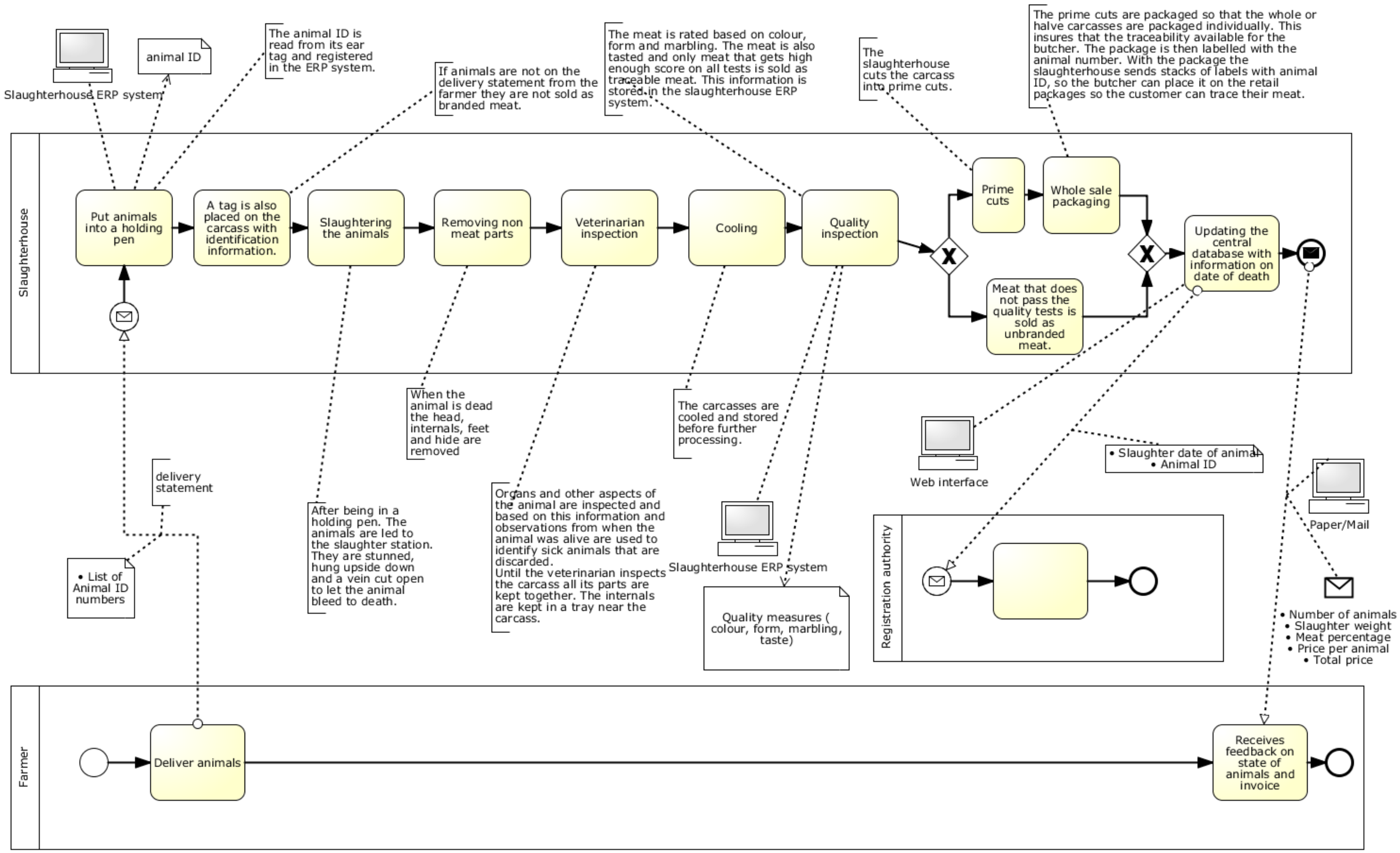

Figure 10: Slaughtering - process model

6.4 Preparing meat for retail

The slaughterhouse process ends with halve or whole carcases packaged individually after having been cut into prime cuts. These pieces are too large for consumer packages. They are cut into smaller pieces at the butcher or in the retail store. It is important that the animal ID gets through to the package so the consumer can get information about the meat he is buying. In Denmark the butcher often works in the retail store, but the process remains the same.

Actors:	Slaughterhouse, Retail store, Butcher.		
Input data:	Orders for packages of specific meat cuts.		
Initiated by:	Retail store or Butcher.		
Actions:		Purpose:	Action:
	1	Order meat from butcher	The retail store orders the meat from the butcher they plan to sell.
	2	Order meat from slaughterhouse	The butcher orders the required number of carcasses from the slaughterhouse to obtain the amount of meat he is to deliver to the retail store.
	3	Slaughterhouse send the carcasses	The slaughterhouse delivers the carcasses to the butcher with enough ID tags for every carcass so the butcher can put a the ID label on every package
	4	Verify the carcasses and IDs match	The butcher receives the carcasses or larger pieces and with every carcass is a set if ID tags prepared by the slaughterhouse to be put on the consumer package. The carcass is usually marked with the animal ID which has to mark the tags received.
	5	Cutting the meat into pieces for selling	The larger pieces of meat received are cut the pieces that the butcher needs to deliver. In this stage the ID tag needs to follow the individual pieces to the next processing stage.
	6	Packaging the meat for retail selling	The cuts are usually put into plastic trays and sealed. The ID of the meat is put on the package so consumers can read the ID.
	7	Deliver meat to store	The packaged meat is delivered to the retail store.
Output data:	Number of packages, weight, price.		

Table 9: Preparing meat for retail - process description

Submitter	**Receiver**	**Transmission method**	**Information details**
Retail store	Butcher	Email/Order	Order for meat pieces and quantities Meat pieces Quantities
Butcher	Slaughterhouse	Email/Order	Order the carcasses needed for retail order What pieces to order Number of pieces
Slaughterhouse	Butcher	Paper	Delivery note Delivered pieces with IDs ID tags Price
Butcher	Retail store	Paper	Delivery note Delivered pieces with IDs Price

Table 10: Preparing meat for retail - information flow

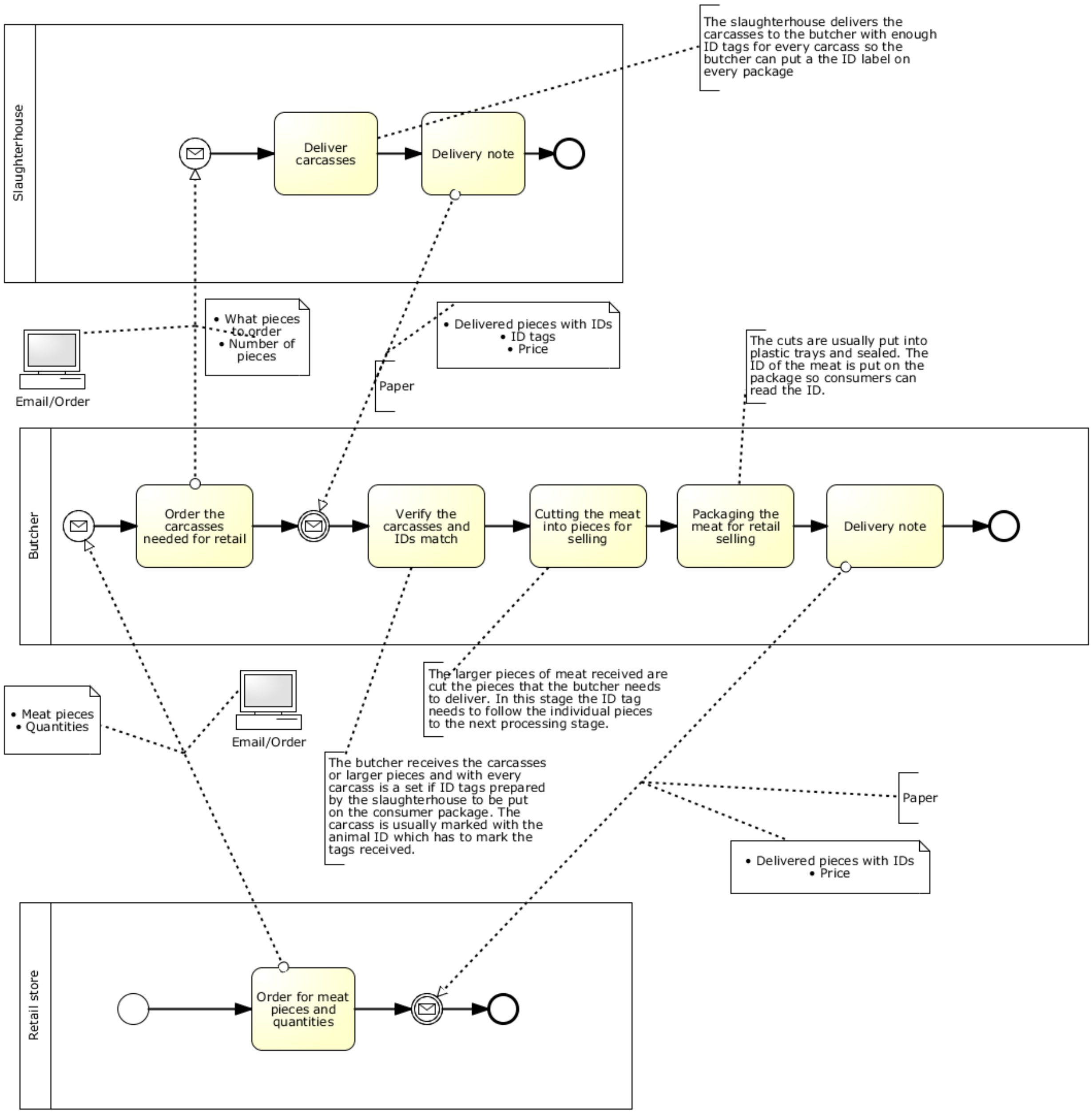

Figure 11: Preparing meat for retail - process model

7. Implementation of Business Processes using Web Services

Splitting a business process into steps that are implemented as independent Web Services allows an organization to be agile in changing its business processes. These services are loosely coupled, making it relatively easy to add new steps or remove obsolete ones from the process. Ideally, new business processes can be implemented by reusing existing Web Services as steps of the process, and the implementation of one step can be replaced with a newer Web Service—even a service provided by a third party—without having to modify and retest the whole process (Hueppi et al. 2008).

Service oriented architecture is an approach to have software resources in an enterprise available and discoverable on a network as well defined services. Each service would achieve a predefined business objective and perform discrete units of work. The services are independent and do not depend on the context or state of the other services. They work within distributed systems architecture. Earlier SOA used COM or ORB based on CORBA specifications and recent SOA stress on Web Services using standard description (WSDL), discovery (UDDI) and messaging (SOAP). Service oriented architecture may or may not use Web Services but yes Web Services provide a simple way towards service oriented architecture albeit with the age old security and reliability limitations (Goel 2006).

"You don't need Web Services to build SOA!" These are words you'll hear many say prior to explaining service-oriented architecture. However, this statement is typically followed by something equivalent to "...but using Web services to build SOA is a darn good idea..." (Erl 2005).

7.1 Process related services

Web Services based on the service-oriented architecture framework provide a suitable technical foundation for making business processes accessible within enterprises and across enterprises. But to appropriately support dynamic business processes and their management, more is needed, namely, the ability to prescribe how Web Services are used to implement activities within a business process, how business processes are represented as Web Services, and also which business partners perform what parts of the actual business process.

An architecture that supports Web Services covers the following aspects.

- The dynamic discovery of registered services. This includes searching for services that meet certain criteria, especially business criteria such as delivery time, price, etc.
- The organization of services, so that one can easily understand what a service offers
- The description of services, so that a service can be properly invoked. This includes formats and protocols for invoking the Web Service (Leymann 2002).

7.1.1 Semantic Web Services

Semantic Web Services promise to add automation and dynamics to current Web Service technologies, considerably reducing the effort required to integrate applications, businesses and customers. One of the key tasks in the integration process is to locate services that can fulfil the application, business or customer needs. With current Web Service technologies this is done mainly manually, which reduces the accuracy of the search and requires considerable effort. Semantic Web Services aim at providing formal descriptions of requests and Web Services that can be exploited to automate several tasks in the Web Services usage process, including dynamic discovery of services (Keller 2005).

The architecture for Web Service discovery through UDDI, or other Web Services registries such as ebXML, breaks down interaction stages into distinct roles, including Service Provider, Registry, and Client. To add semantics to these interactions, no change is required for the UDDI Registry, so roles in the semantic approach can be broken down into Provider and Client. Of the two, the Provider has the greatest burden, providing more detailed information about the meaning of the service. The developer on the client side must also change his or her workflow to use the semantic approach. No changes are needed to the discovery process are needed, but with the use of a semantic description, the client can discover the service semantically, and then apply transformations to adapt the interface of the service to the interface expected by using existing Client software (Fox,Borenstein 2003).

The automatic discovery and composition of Web Services rely on the service description which is annotating of the service functionalities semantically. There are 4 types of semantics in Web Services: data semantics, functional semantics, non-functional semantics, and execution semantics.

Data semantics are the formal definition of data in input and output messages of a Web Service. They are used in service discovery and interoperability between Web Services.

Functional semantics are the formal definition of the capabilities of a Web Service. They can help the discovery and composition of Web Services.

Non-functional semantics are the formal definition of quantitative or non-quantitative constraints like QoS (Quality of Service) requirements like minimum cost and policy requirements like message encryption. They are used in discovery, composition and interoperability of Web Services.

Execution Semantics are the formal definition of the execution or flow of services in a process or of operations within a service. They are used in process verification and exception handling (Yoo 2010).

According to (Fensel, Bussler 2012), **Mediator** describes elements that resolve interoperability problems between different elements, e.g., between two ontologies or two services.

For an open and flexible environment such as web-based computing, adapters are an essential means to cope with the inherent heterogeneity. This heterogeneity can wear many clothes:

- Mediation of **data structures**. A Web Service may provide an input for a second one, however, not in the format it is expecting.
- Mediation of **business logics**. Two Web Services provide complementary functionality and could be linked together in principle (one is a shopping agent and one is a provider of the searched goods), however, their interaction patterns do not fit.
- Mediation of **message exchange protocols**. SOAP over http is unreliable requiring trading partners to implement transport level acknowledgments as well as time-out, retry, upper resend limits as well as duplicate detection in order to guarantee exactly once semantics. Web Services may differ in the way they achieve such a reliability layer.
- Mediation of dynamic **service invocation**. A Web Service may invoke other Web Services to provide its functionality. This can be done in a hard-wired manner; however, it can also be done more flexibly by just referring to certain (sub-) goals. During execution other services can be invoked dynamically. (Fensel, Bussler 2002)

There are 4 types of mediator according to the purpose of mediation.

O-O Mediators (ontology to ontology) resolve mismatches between source ontology and target ontology.
G- G Mediators (goal to goal) connect goals to one another and can create the new goal from existing goals.
W- G Mediators (Web Service to goal) links a Web Service to a goal and resolves terminological mismatches.
W-W Mediators (Web Service to Web Service) are used to establish interoperability between Web Services.

Web Service Modelling Ontology mediators aim to resolve heterogeneity problems at the data, process, and protocol levels i.e., in order to resolve mismatches between different used terminologies (data level), in how to communicate between Web Services (protocol level) and on the level of combining Web Services (process level) (Fensel, Bussler 2002).

7.1.2 The Unified Service Description Language (USDL)

The Unified Service Description Language (USDL) is a platform-independent language for describing services. It was mainly developed by SAP Research bringing together and consolidating the research results and experience gained in various research and customer projects. As a result, there are contributions from various experts in different fields like, software developers, security experts, SLA experts, business economists, legal scientists, exports in semantic technologies etc.

USDL aims at unifying all relevant information about services into one single coherent language. While technical aspects are captured quite well by existing service description languages (e.g. WSDL, SAWSDL etc.), USDL explicitly enables to express business and operational characteristics set by an organization. In that sense, USDL brings together business, operational and technical aspects of services, covering the description of the service providers themselves, price plans, general terms and conditions but also dependencies of one service to other services and instructions on how to combine a particular service. Moreover, technical data is described as for example the concrete interface description (e.g. WSDL) and the provided access protocols (see Figure 12). As a result, USDL helps to minimize the need for practitioners to retain mental references to various documents each describing particular facets of one single service.

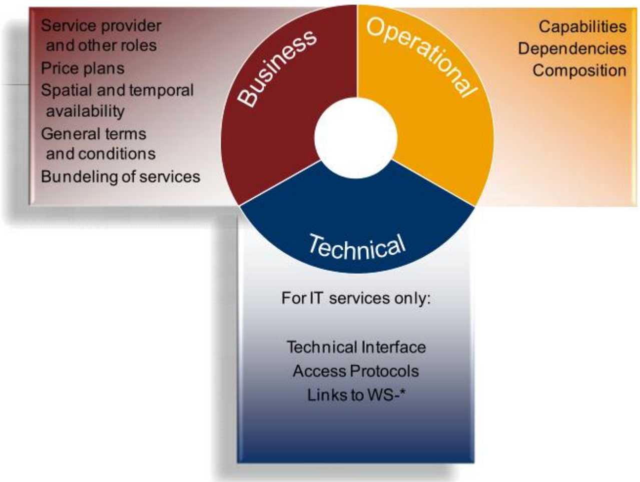

Figure 12: USDL covers information about business, operational and technical aspects

In order to meet the objective of USDL to provide a common service description model, a shared conceptualisation of the service domain need to be defined. The current USDL specification, which is version 3.0, is based on UML. More specifically, the Ecore meta-modelling specification of the Eclipse Modelling Framework (EMF) was chosen. Whereas the distinction between business, operation and technical aspects of USDL helps to introduce the main goal of USDL, it proved to be too coarse-grained for an adequate structuring. Therefore, the USDL specification has been split into 9 packages (according to UML terminology). The resulting modules and their dependencies are depicted in Figure 13.

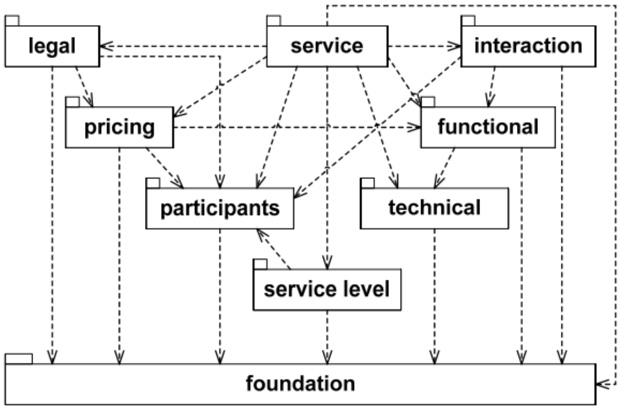

Figure 13: USDL packages and their dependencies

The modular nature of USDL allows using only parts that are needed in order to describe the services of a particular domain. The basis of all packages is the "foundation" module. It assembles concepts that are common to all other packages and is therefore referenced by each and every package. The "service level" package in turn captures concepts that cover quality of service aspects claimed or expected by different actors involved in provisioning or consumption of services. The actors, e.g. stakeholder, provider, intermediary and consumer, are described using the concepts contained in the "participants" package. Pricing and legal information are captured in the corresponding packages. The "service" package captures central service concepts, e.g. service and service bundle, as well as their relations. It depends on the "interaction" package which incorporates concepts that outline sequences of interactions between a consumer and a service needed to successfully complete service execution. Functional details of a service as for example function names, parameter lists or potential faults are captured in the "functional" package. The last package describes "technical" details and thus contains concepts that describe available means to access a service, e.g. interface and access protocols. A comprehensive description of all packages and their content can be obtained from the USDL website[1].

Apart from the ability to describe services and their various facets, USDL has been also designed to integrate with existing organizational systems and services. As an example, the interaction module allows interlinking the service sequence with *existing workflows and business processes descriptions* such as WS-BPEL or BPMN resources. In doing so, those artefacts can seamlessly be reused. Already existing artefacts or models can be reused. This is of special importance taking into consideration that services and business processes share a lot of similarities and, depending on the application, are even regarded synonymously. In that sense, business processes are composed as a "pipeline" of actions, with assigned resources (human or automated) for each, operating on business operations, applications and business resources. Their focus is on the internal details of organizations and their systems, i.e., "how" requests, actions and responses are processed to fulfil consumer goals. Services, in turn, represent more the external interface consumers may access. Usually, the interface exposes several capabilities which can be consumed by external parties. Where business process activities are orchestrated, service capabilities are delivered.

Given the number of packages and the broad scope of USDL, covering different aspects of service description, it can be seen that the modelling of services using USDL may become quite complex. In order to mitigate this situation, USDL provides a graphical editor also freely available from the official website.

Driven by the vision of the Semantic Web, the paradigm of Linked Data has been continuously adopted to the Web. The basic idea of Linked Data is to publish structured data in a way that allows interlinking data sets by using standard Web technologies like unique identifiers for resources (URIs) and HTTP as the default transport protocol. In order to foster a broader adoption of USDL for the Web, a Linked Data version of USDL has been proposed. Instead of UML, it uses RDF Schema to

[1] http://www.internet-of-services.com

model the concepts needed to describe services. Moreover, it has been decided not to cover all modules of USDL but rather provide a lightweight semantic representation. The rationale behind that was to simplify the interlinking of USDL services with existing data sets as well as better reusability of existing vocabularies available on the Web (Friend of a Friend, Dublin Core, GoodRelations and others).

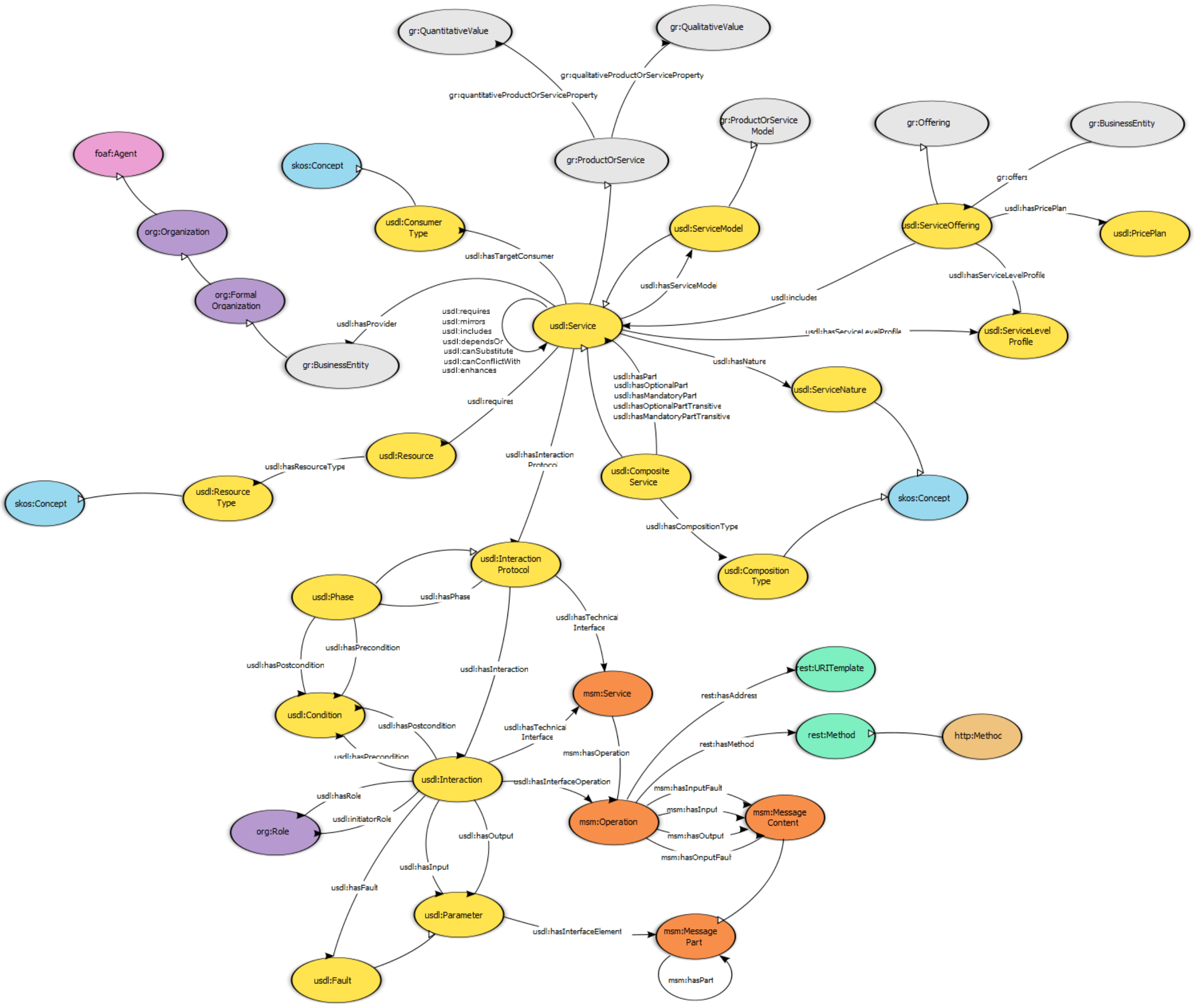

Figure 14: High-level overview of the USDL core ontology

The resulting USDL ontology (see Figure 14 for a high-level overview) can be explored in detail as part of the latest "Unified Service Description Language XG Final Report"[2] of the USDL W3C Incubator Group.

---

[2] http://www.w3.org/2005/Incubator/usdl/XGR-usdl/

8. Conclusions

The purpose of this deliverable was to document the work in task "T3.2 –Semantic business decision models".

Business processes described in Chapters 5 and 6 show a significant difference in their scope and their level of granularity even if both used the same modelling methodology proposed in Chapter 4 of this deliverable. The automotive domain process focuses on low level production data with the overall description of inter business level decisions. The agriculture domain process focuses on a business-to-business interaction with a brief overview of the intra business process. Both processes provide valuable inputs for business vocabulary terms to be understood and supported by the ebbits application. The ebbits solutions should be applicable on several levels of the organizational processes from the production level to the business-to-business level.

Based on these processes, a usage scenario, the business framework and metrics for the ebbits application will be described in deliverables "D3.4 Business framework for online OEEE applications for production and energy optimisation" and "D3.5 Business framework for online food traceability in lifecycle perspective". Vertical and Horizontal vocabularies developed within the "D3.2 Vertical and horizontal business vocabularies" will be updated according to these processes. Business process related ontologies will be delivered in WP4 based on WP3 analysis results. Business process models from this deliverable will be used for the identification of technical requirements in development tasks of the project.

9. References

(Aguilar-Saven 2004)	Aguilar-Savén, R.S.: Business process modelling: Review and framework, International Journal of Production Economics 90 (2), pp. 129-149
(Erl 2005)	Erl, T: Service-Oriented Architecture: Concepts, Technology, and Design, ISBN 0-13-185858-0, Prentice Hall, 2005
(Fensel, Bussler 2002)	Fensel, D. – Bussler, C. - The Web Service Modeling Framework (WSMF), Electronic Commerce Research and Applications, 1(2), 2002
(Fox, Borenstein 2003)	Fox, J. – Borenstein, J.: Semantic Discovery for Web Services, SOA World Magazine, March 27, 2003, http://soa.sys-con.com/node/39718
(Goel 2006)	Goel, A.: Enterprise Integration EAI vs. SOA vs. ESB, ITtoolbox Emerging Technologies, 2006, http://hosteddocs.ittoolbox.com/Enterprise%20Integration%20-%20SOA%20vs%20EAI%20vs%20ESB.pdf
(Hueppi et al. 2008)	Hueppi, F – Wrage, L – Lewis, G.A.: T-Check in Technologies for Interoperability: Business Process Management in a Web Services Context, Carnegie Mellon University, 2008
(Keller 2005)	Keller, U. et al.: Semantic Web Service Discovery, WSMX Deliverable, D10 v0.2, WSMX Working Draft, 2005, http://www.wsmo.org/TR/d10/v0.2/d10.pdf
(Leymann 2002)	Leymann, F. – Roller, D. – Schmidt, M.-T.: Web Services and business process management, IBM Systems journal, VOL 41, No 2, 2002
(Weske 2007)	Weske, M.: Business Process Management: Concepts, Languages, Architectures., ISBN: 978-3-540-73521-2 Springer-Verlag Berlin Heidelberg, 2007
(Yoo 2010)	Yoo, H. et al.: Framework of Semantic Web Services and Service Mediation, WMSCI 2010, http://www.iiis.org/CDs2010/CD2010SCI/SCI_2010/PapersPdf/SA487TV.pdf
(ISA 95)	Lean Validation, ISA-95, http://www.leanvalidation.eu/, 2010

10. Figures and Tables

10.1 Figures

Figure 1: Business process lifecycle 7
Figure 2: Automated car production 12
Figure 3: Car body components 13
Figure 4: Body welding 13
Figure 5: Order flow 16
Figure 6: Car manufacturing production process 18
Figure 7: Car manufacturing production process aware of energy consumption. 19
Figure 8: From farm to fork 20
Figure 9: Deliver animals to slaughterhouse process model 22
Figure 10: Slaughtering - process model 25
Figure 11: Preparing meat for retail - process model 27
Figure 12: USDL covers information about business, operational and technical aspects 30
Figure 13: USDL packages and their dependencies 31
Figure 14: High-level overview of the USDL core ontology 32

10.2 Tables

Table 1: Automotive scenario stakeholders 14
Table 2: Failure/alarm 14
Table 3: Alarm example 15
Table 4: Order data format 17
Table 5: Deliver animals to slaughterhouse - process description 21
Table 6: Deliver animals to slaughterhouse - information flow 21
Table 7: Slaughtering - process description 23
Table 8: Slaughtering - information flow 24
Table 9: Preparing meat for retail - process description 26
Table 10: Preparing meat for retail - information flow 27