Enabling the business-based
Internet of Things and Services

(FP7 257852)

D2.8.2 Change request and re-engineering report 2

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 control page

Document file: D2.8.2 Change request and re-engineering report 2 V2.0.doc
Document version: 2.0
Document owner: L. Christiansen (IN-JET)

Work package: WP2 – Requirements engineering and validation
Task: T2.3 – Evolutionary requirements refinement
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	H. Udsen (IN-JET)	2012-09-13	ToC
0.3	M. Jacobsen (TNM), P. Brizzi (ISMB)	2012-09-21	WP8 and WP10 related LL and requirements update
0.4	J. Glova (TUK)	2012-09-22	WP3 related LL and requirements update
0.5	M. Knechtel (SAP)	2012-09-24	WP4 related LL and updated requirements
0.7	M. Ahlsen, P. Kool (CNET), P. Cultrona (COMAU)	2012-09-25	WP7, WP9 and WP10 related LL and updated requirements
0.9	M. Ahlsen, P. Kool (CNET), Y. Martin (SAP), J. Simon (FIT)	2012-09-26	WP5, WP6, WP7 & WP9 related LL and updated requirements
0.91	L. Christiansen	2012-09-27	Editing of all sections
1.0	H. Udsen	2012-10-04	General editing, comments to sections 4 and 5
1.5	M. Ahlsén, J. Simon, P. Brizzi, T. Kjær (TNM), H. Udsen	2012-10-10	Section 7 added, comments addressed, Sections 1+2 updated
2.0	H. Udsen	2012-10-25	Reviewers' comments and suggestions addressed Final version submitted to the European Commission

Internal review history:

Reviewed by	Date	Summary of comments
P. Rosengren (CNET)	2012-10-24	Approved with suggested changes and additions
M. Spirito (ISMB)	2012-10-15	Approved with minor comments and suggestions

Index:

1. Executive Summary

This deliverable reports the results from the second iteration cycle of the work done in *Subtask 2.3.1 Lessons Learned collection and analysis* and contains a complete list of Lessons Learned during this cycle of the ebbits project, organised per work package.

The subsequent analysis of Lessons Learned has identified a number of relevant improvement opportunities for the specification of requirements for the next cycles of the iterative development process.

Also included in this deliverable are the results of *Subtask 2.3.5 Requirements re-engineering*. The resulting new and updated requirements arise from the analysis of Lessons Learned and from the continuous technology, regulatory standards and market watch.

The document combines the content of what was originally planned as two separate deliverables, i.e., *D2.7.2 Lessons Learned and results of usability evaluation 2* and *D2.8.2 Change request and reengineering report 2*, combined under the name of the latter. This measure was taken to reduce duplication and redundancy.

The ensuing document feeds into deliverable *D2.9.2 Updated requirements report 2.*

1.1 Lessons Learned

Section **Error! Reference source not found.** contains all Lessons Learned in cycle 2 and the subsequent analysis, the outcome of which is the identification of a number of improvement opportunities. The ensuing changes in requirements are reported in Section 5.

The Lessons Learned have been collected and reported per work package. Details can be seen in the table below in Section 1.2.

In the second cycle no Lessons Learned have been collected in WP1, WP2, WP11 and WP12.

In total, the second iteration cycle yielded 56 Lessons Learned.

1.2 Requirements Engineering

Section 5 describes the requirement engineering work performed in the second iteration cycle. This has resulted in the creation of 33 new requirements, modification of 24 requirements and deletion of 40 requirements, relative to the complete list reported in *D2.9.1 Updated requirements report 1*. As expected, these changes arise mainly from the efforts of the technical work packages.

Work package	Lessons Learned	New requirements	Updated requirements	Deleted requirements
WP3	2	1	0	0
WP4	6	8	6	36
WP5	9	5	9	0
WP6	4	2	8	0
WP7	6	3	0	0
WP8	20	12	0	2
WP9	2	0	0	0
WP10	7	2	1	2
Total	56	33	24	40

During the second iteration cycle all existing requirements have been converted from the GForge tool to the JIRA Issue Tracker and Requirement Management tool. This change also involved the implementation of a dedicated workflow for processing and managing of the requirements.

1.3 Impact Assessment

At mid-term of the project, the first version of the overall technical ebbits architecture is consolidated. The second platform prototype at M22 implements a subset of the specified components of this architecture, as shown in Figure 1.

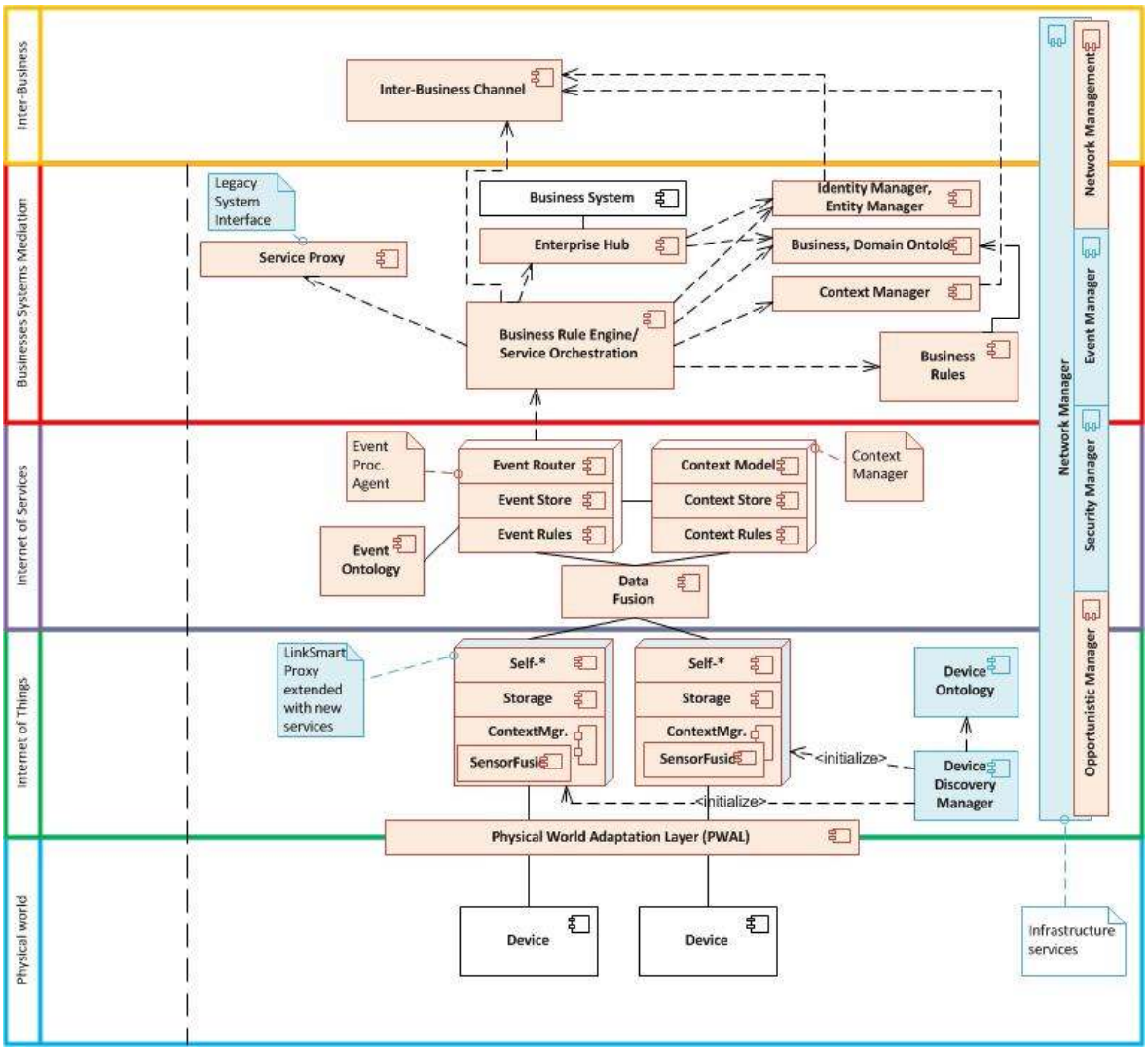

Figure 1: A layered view of ebbits architecture at M24

The application domain analysis and requirements engineering during the second cycle have led to more focused application examples in the Food Traceability and the Automotive Manufacturing scenarios.

The new and revised requirements will underpin the further design and implementation decisions for the platform, particularly in the areas of Product Life Cycle Management, Business System Mediation, Internet of Services, Internet of Things and the Physical World.

2. Introduction

This deliverable reports outcomes of the second iteration cycle of Task *T2.3 Evolutionary requirements refinement*, more specifically the results of *Subtask 2.3.1 Lessons Learned collection and analysis* and *Subtask 2.3.5 Requirements re-engineering*.

The document combines the content of what was originally planned as two separate deliverables, i.e., *D2.7.2 Lessons Learned and results of usability evaluation 2* and *D2.8.2 Change request and reengineering report 2*, combined under the name of the latter. This measure was taken to reduce duplication and redundancy.

2.1 Purpose, context and scope of this deliverable

This document contains a complete list of Lessons Learned during the second iteration cycle of the ebbits project, organised per work package. These Lessons have been extracted from the joint repository in the GForge Wiki.

The Lessons Learned have subsequently been analysed to elicit relevant improvement opportunities for the specification of requirements for the next cycles of the iterative development process.

This analysis has resulted in the creation of new requirements and updating and deleting of existing requirements. Additional changes to the requirements arise from the continuous technology, regulatory standards and market watch and from verification and validation results.

The content of this document feeds into deliverable *D2.9.2 Updated requirements report 2*.

2.2 Replacement of management tool

During the second iteration cycle it was decided to switch to a different tool for the management and tracking of requirements in the ebbits project. Therefore all existing requirements have been converted from the GForge tool to the JIRA Issue Tracker and Requirement Management tool, which is considered more suitable for ebbits purposes and easier to work with.

At the same time a dedicated JIRA workflow was devised and implemented, involving two additional steps compared to the JIRA default workflow, i.e., the steps 'QC Passed' and 'Part of Specification'. The workflow used in the ebbits project is shown in Appendix 1.

3. Research and Development Methodology

The ebbits project has adopted a human-centred, iterative development process following the guidelines of ISO 9241-210 Human-centred design for interactive systems[1]. A description of the methodology, the software engineering process, the iterative approach, the reengineering of requirements and the ebbits application of Lessons Learned can be found in deliverable *D2.7.1 Lessons Learned and results of usability evaluation 1*.

[1] ISO 9241-210:2010-03 (E). Ergonomics of human-system interaction - Part 210: Human-centred design for interactive systems

4. Lessons Learned in the Second Cycle

This section contains all Lessons Learned in cycle 2 and the subsequent analysis. To facilitate referring to individual Lessons Learned they have been named LL followed by the relevant work package number and Lesson number (as they appear in the Wiki repository), e.g. LL WP3-1. This process results in the identification of a series of improvement opportunities which may lead to new, changed or deleted requirements. The changes in requirements are reported Section 5.

A total of 56 Lessons Learned has been reported in the second iteration cycle.

4.1 Lessons Learned in WP3

The work undertaken in WP3 involves Enterprise frameworks for life cycle management. TUK is the WP leader and two Lessons Learned have been collected and validated from this WP.

Org. No.	Experience and knowledge gained	Lesson Learned	Requirement affected
TUK LL WP3-1	Except the value viewpoint the business process viewpoint is necessary.	The business process viewpoint shows us how a new e-commerce idea can be put into operation (through the outline of operational activities, sequence of performance, inputs and output for activities, and their operational actors).	
TUK LL WP3-2	Value model deconstruction and reconstruction is applicable to find variations on an existing value model.	Deconstruction and reconstruction of the value model help us to find variations on an existing model through reflection of alternative value activities	EBBITS-377

4.1.1 Analysis of Lessons Learned

The Lessons Learned in WP3 deal with the business modelling framework and concepts. Two LL were identified:

- **LL WP3-1:** The LL opens new perspective - business process viewpoint – in business modelling framework. This enables us to understand if a new IoT idea can be put into operation.
- **LL WP3-2:** The LL provides the other perspective of created or existed value models which should be viable to reflect and understand the variation in value created. The new requirement EBBITS-377 has been created to reflect it.

4.2 Lessons Learned in WP4

The work undertaken in WP4 relates to Semantic knowledge infrastructure. SAP is the WP leader and six Lessons Learned have been collected and validated from this WP. Though some revisions have been made in November and December 2011 with the old Requirements Tracker tool GForge still in place, the table refers to the Requirement IDs in the JIRA tool.

Org.	Experience and knowledge gained	Lesson Learned	Requirement affected

No.			(JIRA IDs)
(WP4 Conference Call 18/11/2011) LL WP4-1	Separate conceptual model from semantic storage	We moved EBBITS-288, EBBITS-290 and EBBITS-291 from WP4 to WP5 since the details of the context model are defined in WP5. However the technical semantic storage in the WP4 semantic store has to be done according to the context model defined by WP5. For this reason we newly created EBBITS-360 and EBBITS-363. The merging procedure has been agreed with all WP4 partners in the calls 2011-11-23 and 2011-11-18. It has also been agreed with WP5 lead Ferry Pramudianto, participating in the latter call.	EBBITS-360, EBBITS-288, EBBITS-290, EBBITS-291, EBBITS-363
(WP4 Conference Call 18/11/2011) LL WP4-2	Fit criteria more specific	Fit criterion is no repetition of the requirement, but is a measurable condition determining whether the requirement has been met. This has been agreed with all WP4 partners in the call 2011-11-18.	EBBITS-287, EBBITS-289
(WP4 Conference Call 23/11/2011) LL WP4-3	One concern per requirement	Some requirements involve several aspects. Instead of having one unfocused requirement, create several ones, one for each aspect. For this reason, Requirement EBBITS-299 was split up, the modelling aspects were integrated into EBBITS-363, and EBBITS-365 and EBBITS-366 were newly created. This has been agreed with all WP4 partners in the calls 2011-11-23 and 2011-11-18.	EBBITS-299, EBBITS-363, EBBITS-365, EBBITS-366
(WP4 Conference Call 23/11/2011) LL WP4-4	General Design and Enforcement of an Access Control Policy is out of Scope of WP4	The requirements from WP4 concerning Access Control policy have been merged into a newly created requirement in WP8, namely EBBITS-361. This has been agreed with WP8 lead Maurizio Spirito. The merging procedure has been agreed with all WP4 partners in the calls 2011-11-23 and 2011-11-18. The title, and the fit criterion is "Access Control Policy and enforcement are specified". All original Requirements in WP4 have been closed as Duplicates and their fit criteria have been inserted to the newly created EBBITS-361.	EBBITS-306, EBBITS-307, EBBITS-308, EBBITS-310, EBBITS-311, EBBITS-312, EBBITS-314, EBBITS-315, EBBITS-316, EBBITS-317, EBBITS-318, EBBITS-319, EBBITS-320, EBBITS-321, EBBITS-322, EBBITS-323, EBBITS-324, EBBITS-325,

		In WP4 we newly created EBBITS-367 "ebbits Access Control Policy will be respected by Ontology Manager". The general design and enforcement is not in focus while the enforcement by WP4 components is in focus.	EBBITS-361, EBBITS-367
(WP4 Conference Call 23/11/2011) LL WP4-5	Publish-subscribe mechanism is not WP4 focus.	A general publish-subscribe mechanism is not in the focus of WP4. For this reason the requirement has been moved from WP4 to WP5 as agreed in WP4 call 2011-11-23, with WP5 partners attending.	EBBITS-309
(WP4 Conference Call 23/11/2011) LL WP4-6	One semantic model requirement per domain	Several requirements were identified that can be merged into a single one in the calls 2011-11-23 and 2011-11-18. Requirements EBBITS-288, EBBITS-290, EBBITS-291, EBBITS-292 and EBBITS-293 were merged into EBBITS-362. Requirements EBBITS-294, EBBITS-295, EBBITS-296, EBBITS-297, EBBITS-298, EBBITS-300 and EBBITS-313 were merged into EBBITS-363. Requirements EBBITS-301, EBBITS-302, EBBITS-303 and EBBITS-304 were merged into EBBITS-364. Merged requirements were assigned to WP4.	EBBITS-288, EBBITS-290, EBBITS-291, EBBITS-292, EBBITS-293, EBBITS-294, EBBITS-295, EBBITS-296, EBBITS-297, EBBITS-298, EBBITS-300, EBBITS-301, EBBITS-302, EBBITS-303, EBBITS-304, EBBITS-313, EBBITS-362, EBBITS-363, EBBITS-364

4.2.1 Analysis of Lessons Learned

Several Lessons Learned resulted in a concerted activity of all WP4 partners to revise and re-assign requirements.

- **LL WP4-1:** Requirements EBBITS-288, EBBITS-290 and EBBITS-291, which describe the details of the context model, were moved to WP5. To define the technical solution for a semantic representation of the model, requirements EBBITS-360 and EBBITS-363 were created.
- **LL WP4-2:** The fit criterion of requirements EBBITS-287 and EBBITS-289 were too unspecific. These requirements were updated with a more specific fit criterion.
- **LL WP4-3:** Requirement EBBITS-299 was too broad and was closed (resolved as Duplicate). The contents were split up into three new requirements, EBBITS-363, EBBITS-365, and EBBITS-366.
- **LL WP4-4:** The requirements EBBITS-306 through EBBITS-308, EBBITS-310 through EBBITS-312 and EBBITS-314 through EBBITS-325 described details of an access control policy, which is not in the scope of WP4. They were closed and summarised in the new requirement EBBITS-361 associated with WP8. The requirement EBBITS-367 was created to reflect the necessary access policy enforcement by WP4 components.

- **LL WP4-5:** The requirement EBBITS-309, describing a general publish-subscribe mechanism, was updated to change its work package from WP4 to WP5.
- **LL WP4-6:** The requirements EBBITS-288, EBBITS-290 through EBBITS-298, EBBITS-300 through EBBITS-304 and EBBITS-313 described the domain models with too much detail. They were closed (resolved as Duplicate) and one requirement per domain was created to describe the semantic models (EBBITS-362 and EBBITS-364).

4.3 Lessons Learned in WP5

The work in WP5 involves Centralised and distributed intelligence. FIT is the WP leader and nine Lessons Learned have been collected and validated from this WP.

Org. No.	Experience and knowledge gained	Lesson Learned	Requirement affected
FIT LL WP5-1	Centralizing context management will not scale if the number of events is high.	Distributing context processing could increase the performance.	EBBITS-332, EBBITS-178, EBBITS-200
FIT LL WP5-2	The overhead of web service introduces latency, which is not acceptable for manufacturing scenario.	Place sensor fusion module as close as possible to the physical devices.	EBBITS-328, EBBITS-327
FIT LL WP5-3	Information about device states are needed for self-* reasoning.	Context of a device need to be monitored and provided to the self-* manager.	EBBITS-196, EBBITS-291
FIT LL WP5-4	Proxy connection to the physical devices could be unstable.	Self-* manager needs to monitor the connection to the physical devices.	EBBITS-250, EBBITS-381. EBBITS-196
FIT LL WP5-5	Sensors causing a lot of problems could affect the stability of the proxy.	Exclude or shut off physical devices from the network if it causes a lot of problem.	EBBITS-382, EBBITS-196
FIT LL WP5-6	Static frequency of sensor reading is not optimal when the network bandwidth fluctuates because it causes congestion when the bandwidth is currently slow.	Adjust data traffic according to the network bandwidth.	EBBITS-332, EBBITS-309, EBBITS-383, EBBITS-385
FIT LL WP5-7	Reset is an easy way to recover the device states.	Reset devices upon problems when no other fix is defined by the developer.	EBBITS-196, EBBITS-382, EBBITS-384
FIT LL WP5-8	Developers need a clear guideline what self-* does, and how to implement self-*	Clear focus and boundary of self-* is required.	EBBITS-381, EBBITS-196
FIT LL WP5-9	Ebbits supports publish/subscribe (EBBITS-309) via Event Manager, but a way to configure the event publication (topic, frequency, data fusion etc.) is needed to support adjusting data traffic according to the network bandwidth, see LL 5-6, and	Device Proxies need a standardized interface that provides control management services for event publication	EBBITS-309, EBBITS-332, EBBITS-383, EBBITS-385

	to support a large number of events (see EBBITS-332)		

4.3.1 Analysis of Lessons Learned

The Lessons Learned in WP5 deal mostly with the interface between Context Manager and Device Proxies:

- **LL WP5-1:** Requirements EBBITS-332, EBBITS-178, EBBITS-200 are updated to reflect the distributed approach.
- **LL WP5-2:** Requirements EBBITS-328 and EBBITS-327 provide the basis for the implementation of this LL and are updated to reflect the new details of placing the sensor fusion modules.
- **LL WP5-3:** The LL makes a link between the diagnostic self-* component (EBBITS-196) and device state information (EBBITS-291). These requirements are linked to each other.
- **LL WP5-4:** In addition to requirements EBBITS-250 and EBBITS-196 which provide the basis for this LL, the new requirement EBBITS-381 is created.
- **LL WP5-5:** The LL describes new functionality of the diagnostic self-* component (EBBITS-196), resulting in the new requirement EBBITS-382.
- **LL WP5-6:** Context management uses the publish-subscribe mechanism described in requirement EBBITS-309. During implementation, it was realized that additional functionality is needed in order to support a high number of sensor events (EBBITS-332), therefore requirement EBBITS-383 was created. This is also related to LL WP5-9, which resulted in the new requirement EBBITS-385.
- **LL WP5-7:** This LL is related to LL WP5-5, offering an alternative approach to a similar problem. Therefore, the new requirement EBBITS-384 is created and requirements EBBITS-196 and EBBITS-382 are updated.
- **LL WP5-8:** The LL describes new functionality of the diagnostic self-* component (EBBITS-196), resulting in the new requirement EBBITS-381.
- **LL WP5-9:** Describes an implementation approach to enable the publish-subscribe mechanism (EBBITS-309) for a large number of events (EBBITS-332) and LL 5-6 (EBBITS-383), resulting in the new requirement EBBITS-385.

4.4 Lessons Learned in WP6

The work in WP6 revolves around Mainstream business systems. SAP is the WP leader and four Lessons Learned have been collected and validated from this WP.

Org. No.	Experience and knowledge gained	Lesson Learned	Requirement affected
FIT LL WP6-1	Experts in EE domain mentioned the need to base energy monitoring on production processes.	Ebbits needs to support process-based energy monitoring.	EBBITS-213, EBBITS-215, EBBITS-232
FIT LL WP6-2	Monitoring and context information need to be transferred into business systems in order to support management decisions.	Interfaces to business systems should take this information into account.	EBBITS- 281, EBBITS-286, EBBITS-330, EBBITS-331, EBBITS-337
SAP	Services have non-technical aspects which	Bring together business,	EBBITS-368

LL WP6-3	should also be described.	operational and technical aspects of service descriptions into one single coherent language.	
SAP LL WP6-4	The important data from web services and background knowledge in form of ontologies should be accessible in combined queries.	SPARQL can be used as query language to ask information from ontologies and get the important data from web services.	EBBITS-369

4.4.1 Analysis of Lessons Learned

Regarding web services, two Lessons were identified:

- LL WP6-3 resulted from the description of a web service of the national Danish Cattle Database which has also non-technical aspects. To accommodate also such aspects, a corresponding service description language should be used. Therefore, requirement EBBITS-368 was created.
- LL WP6-4 resulted from the matching of different web services, and requirement EBBITS-369 was created.

Regarding the energy monitoring, one Lesson was identified:

- LL WP6-1 was proposed by experts in the domain, affecting requirements EBBITS-213, EBBITS-215 and EBBITS-232.

Regarding context information, one Lesson was identified:

- LL WP6-2 could lead to improved management decisions. This affects requirements EBBITS-281, EBBITS-286, EBBITS-330, EBBITS-331 and EBBITS-337.

4.5 Lessons Learned in WP7

The work undertaken in WP7 deals with Event management and service orchestration. It is led by CNET, and for this iteration the following six Lessons Learned have been collected and validated.

Org. No.	Experience and knowledge gained	Lesson Learned	Requirement affected
CNET LL WP7-1	The revised event processing architecture with the EPAs (Event Processing Agents) provides a good basis for scaling and distribution of event processing capabilities.	Any software component in the ebbits architecture could be provided with event processing capability, i.e. be realized as a specialization of a generic EPA.	EBBITS-336 EBBITS-337
CNET LL WP7-2	The LinkSmart Event Manager was extended to provide better resilience and reliability.	The LS event manager should be kept as a stand-alone component, providing a general communication channel based on the publish & subscribe mechanism.	
CNET	The Event Processing Agent needs the entity structure for	Writing rules involving complex entities, for instance entities that	EBBITS-378

LL WP7-3	writing rules over complex entities	are composed from sub-entities, requires knowledge of the entity structure. (NB! ObjectID in the event model maps to the entity ids).	
CNET LL WP7-4	The Event Processing Agent needs to be able to find relating entities from a given entity id.	Writing rules that involve entities at different abstraction levels needs to be able to refer to relationships in-between entities. For instance given the id of an engine it must be possible to find out the id of the car where it is mounted.	EBBITS-379
CNET LL WP7-5	Knowledge of the process model can be useful for efficient filtering of events.	Some events are only interesting in certain production steps.	EBBITS-380
CNET LL WP7-6	Events need to be able to refer to the process model to manage rules that define the process step where they are created.	Writing rules that take into account the process need the process model to be able to enhance the events with process IDs.	

4.5.1 Analysis of Lessons Learned

These LLs refer to the re-design and implementation of the new event processing model, and the need for entity and process identities.

- WP7-1 and WP7-2 stem from our design and implementation work on the revised Event Processing architecture. Event processing agents are intended for the rule-based filtering and processing of semantically enhanced events, whereas the LinkSmart event manager is available as an infrastructure component generally available as a communication mechanism in the ebbits architecture.
- WP7-3 refers to the need of having access to the structure of entities managed in the system (application objects and system objects) in order to enhance the expressiveness in rules and procedures processing events. A new requirement EBBITS-378: *Entity structure accessible from event rules* has for this issue.
- WP7-4 is related to the previous one, but on the entity instance level, resulting in new requirement EBBITS-379: *Navigable entity IDs.*
- WP7-5 and WP7-6 has to do with the possibility of making use of process descriptions (models) in event and data management. These LLs result in the new requirement EBBITS-380: *Using Process IDs in Events.*

4.6 Lessons Learned in WP8

The work in WP8 takes care of Physical world sensors and networks. ISMB is the WP leader and 20 Lessons Learned have been collected and validated from this WP, as reported in the table.

Org. No.	Experience and knowledge gained	Lesson Learned	Requirement affected
FIT LL WP8-1	It is hard for companies to integrate smartcards into their system and they are not likely to do it unless highly supported.	We cannot build on smartcards	
FIT LL WP8-2	Companies do not understand security risks	Hard to get security requirements from	EBBITS-387

		user partners	
FIT LL WP8-3	Out of channel reporting of network events is preferred to automatic reaction to problems	There is no trust in IT. Out of channel alarm mechanisms are needed	EBBITS-373
FIT LL WP8-4	Companies have restricted policies for information sharing and network traffic management	Policy and rule based procedures are required	EBBITS-387
FIT LL WP8-5	Companies build on configurable logging in their systems for checking operation	ebbits has to support logging mechanisms	EBBITS-388
FIT LL WP8-6	Some information in traceability may miss or be corrupted	ways of reporting available information is necessary	EBBITS-393
FIT LL WP8-7	Stakeholders could try to withhold data to make meat more valuable	Provided information has to be double checked or provided by trust entity	
FIT LL WP8-8	Information validity cannot be ensured inside a plant	Some authority has to control farms, etc regularly	
IS LL WP8-9	Along with globally unique and persistent entity IDs, the existing identification schemes and possible future schemes should be supported.	Interoperability of applications, which use different identification schemes is required.	EBBITS-374
IS LL WP8-10	Associations between identifiers should be recorded and queried.	Mapping and relation service should be part of the Entity Manager.	EBBITS-374
ISMB LL WP8-11	LinkSmart does not support proxy authentication through NTML	Middleware deployments should take into account the possible strict security policies in enterprise networks	EBBITS-389
ISMB LL WP8-12	Robot Controller server does not support multiple clients	RC server should be updated to support connections from multiple clients	EBBITS-375
ISMB LL WP8-13	Robot Controller server freezes when client interrupts communication abruptly	RC should handle properly sudden communication interruptions	EBBITS-376
ISMB LL WP8-14	PWAL framework does not scale with the support of new drivers	PWAL framework project should be separated from the drivers logic	EBBITS-372

ISMB LL WP8-15	Developing of new PWAL drivers requires always to modify core PWAL framework	PWAL framework should be independent of the number of supported PWAL drivers	EBBITS-372
ISMB LL WP8-16	PWAL does not handle multiple drivers of the same type	PWAL handling of drivers should allow multiple drivers, even of the same type	EBBITS-372
ISMB LL WP8-17	PWAL PLC driver does not support a recursive discovery of symbols	PLC driver should be updated to include a recursive discovery of symbols	EBBITS-390
ISMB LL WP8-18	PWAL RC driver does not support an automatic map of parIDs to variables	RC driver should be updated to implement some mechanism to receive/retrieve a table with the semantic information regarding the available parIDs in the RC	EBBITS-392
ISMB LL WP8-19	PWAL RC driver does not support an automatic discovery of available axes	RC driver should be updated to parse information regarding available axes in the RC	EBBITS-391
FIT LL WP8-20	PWAL design is already very complex, yet lacks some of the requirements defined for it, meaning that the scope according to the requirements is too broad.	The scope of the PWAL is restricted to providing drivers for device proxies. Mapping the PWAL attributes to the ontology is shifted to WP4 tools. Data fusion should be provided by higher-level services, yet in the device proxy	EBBITS-348, EBBITS-351

4.6.1 Analysis of Lessons Learned

Regarding software development experience issues, two Lessons were identified, both regarding the Physical World Application Layer (PWAL) design and development:

- **LL WP8-14:** PWAL framework should support the execution of multiple drivers at runtime without affecting its performance significantly; the core framework currently should provide methods to divide PWAL workload if needed. To accomplish this requirement EBBITS-372 was created.

- **LL WP8-15:** PWAL framework should support the addition of multiple drivers without modification of its core design; the core framework should not have any dependencies with respect to the logic which manages devices drivers. The development of new drivers supporting new devices or subsystems would benefit from better generalization of the PWAL core, avoiding on the need to update drivers for legacy PWAL drivers when its core design is changed. This is also covered by EBBITS-372.

Also a more general research Lesson concerning the PWAL design was identified:

- **LL WP8-20:** The PWAL entails a complex development workflow for drivers because of the broad initial PWAL scope (according to already defined requirements): in order to simplify development, it has been considered necessary to better separate some PWAL-ebbits features which were partially overlapping with other ebbits components (i.e. by moving to upper layer some more complex capabilities). This has led to the closing of EBBITS-348 and EBBITS-351 as being Out of Scope, thus easing integration of new devices.

Regarding Network oriented Lessons, four topics have been highlighted:

- **LL WP8-3:** In case of IT-related failures, it is not always possible to programmatically react and correct at runtime problems or errors. In order to avoid introduction of more complexity, at the moment it is preferred to simply notify errors (e.g., to network administrators), relying on "out of channel" communication. As a consequence requirement EBBITS-373 was created.
- Two issue are related to applications interoperability:
    - **LL WP8-9:** Applications use different identification schemes. ebbits should support both its own existing identification schemes and other possible future identity management schemes. As a result requirement EBBITS-374 was created.
    - **LL WP8-10:** Elaborating on **LL WP8-9,** a need is foreseen to support an association between two different types of identifier (while storing and querying them), which should be managed by the Entity Manager. This is also covered by EBBITS-374.
- **LL WP8-12:** It has been evaluated that the Robot Controller server does not support multiple concurrent clients to avoid degradation of performance of the control loop. As a consequence, the Robot Controller could be updated to fill this gap, or, alternatively, its PWAL driver should manage the issue by queuing uncritical commands. This led to the creation of EBBITS-375.

Six Lessons Learned have to do with security issues:

- Four of them pertain to production and traceability cycles:
    - **LL WP8-1:** It is difficult to rely on smartcard technologies, because companies refuse to integrate smartcards into their infrastructures unless they become highly supported.
    - **LL WP8-2:** Companies do not know/understand security risks and so it is hard to get security requirements from user partners. This resulted in creation of requirement EBBITS-387.
    - **LL WP8-4:** Companies usually have constrained policies to share information and manage network, so policy and rule based procedures should be properly defined. This is covered by EBBITS-387.
    - **LL WP8-7:** Stakeholders may withhold data in order to create further revenue (e.g. in the meat traceability field): to prevent this it is necessary to ensure data origin correctness.
    - **LL WP8-8:** External authority body has to control farms or production site in order to control information sources, because sometimes information validity cannot be ensured.
- While at software level:

- **LL WP8-11:** LinkSmart/ebbits deployments should take into account the possible strict security policies within already existing enterprise networks, because today LinkSmart does not support proxy authentication through NTML. As a consequence requirement EBBITS-389 was created.

Finally, seven Lessons Learned are related to the overall process.

- Two Lessons concern information propagation and control:
  - **LL WP8-5:** ebbits must support configurable and flexible logging mechanisms in order to support companies during verification/checking operations. This resulted in creation of requirement EBBITS-388.
  - **LL WP8-6:** It must be possible to detect, store and report meta-information about presence or lack of traceability information while it is being collected, in order to diagnose, control and eventually prevent loss of information from the field. This resulted in creation of requirement EBBITS-393.
- **LL WP8-13:** In case the client abruptly interrupts communications, the Robot Controller should handle properly those eventual interruptions without freezing. This resulted in creation of requirement EBBITS-376.
- Regarding the PWAL:
  - **LL WP8-16:** As already mentioned, PWAL should handle multiple drivers at runtime, even if different drivers belong to the same class. This is also covered by EBBITS-372.
  - **LL WP8-17:** PLC developers often rely on nested structures to organise variables and signals; the PLC PWAL driver should be updated to support recursive symbols discovery. This resulted in creation of requirement EBBITS-390.
  - **LL WP8-18:** The Robot Controller itself does not support an automatic map of parIDs to variables and so a mechanism to receive and retrieve a list with the semantic information regarding the available parIDs is needed within the PWAL Robot controller driver. This led to the creation of requirement EBBITS-392.
  - **LL WP8-19:** PWAL Robot Controller driver should be updated to parse information regarding available axes in order to provide support to automatic axis discovery. This resulted in creation of requirement EBBITS-391.

4.7 Lessons Learned in WP9

The work in WP9 involves Platform integration and deployment. CNET is the WP leader and two Lessons Learned have been collected and validated from this WP.

Org. No.	Experience and knowledge gained	Lesson Learned	Requirement affected
CNET LL WP9-1	The project should consider setting up a server permanently for the backend deployment and where everybody can have access to deploy their server side software.	Having a deployment server that is accessible from outside helps when integrating clients with servers. CNet allocated a server for use of the back ends that is accessible for CNET	

		and SAP.	
CNET LL WP9-2	The project should consider buying a cloud server that can be used for deployment, such as an Amazon server.	Setting up a deployment server accessible for everybody involves a lot of work if it is contained in one organisation. The reason for this is that it necessary to create logins to the VPN in order not for the server to be vulnerable from attacks.	

4.7.1 Analysis of Lessons Learned

The Lessons Learned in WP9 is of an organisational nature. The proposed actions will be discussed among the technical partners and thus will not create any formal change requests.

4.8 Lessons Learned in WP10

The work undertaken in WP10 involves End-to-end business applications. COMAU is the WP leader and 7 Lessons Learned have been collected and validated from this WP.

Org. No.	Experience and knowledge gained	Lesson Learned	Requirement affected
TNM LL WP10-1	As part of the research for our traceability prototype we visited secondary processors (butchers) and learned just how old and non-modern their current equipment is. The main point here is that we cannot expect the local equipment to support any kind of QR code printing to the stickers of the meat packages. Likewise we cannot expect to gather secondary information regarding the meat products (weight, best before date, type of product produced, etc.) from these pieces of equipment.	Instead this information must be gathered through the interface of the application during the work process.	EBBITS-166
TNM LL WP10-2	During our visit to secondary processors, we learned that the butchers have a very limited budget for equipment. They are ready to invest, but only if they can see a real benefit. It could therefore be beneficial if the programs are able to run on a range of different equipment. For example a product that ideally runs on an industrial computer with a coupled QR code scanner is also able to run on a simple smartphone, scanning QR-codes with the camera. In that way the butcher	The software for secondary processors must be able to be executed on a range of different equipment types.	EBBITS-166

	can see the benefit of the program without initially having to invest in any additional equipment		
TNM LL WP10-3	In a modern butcher shop a lot of the staff is actually temporary workers. So you cannot expect them to know different pieces of meat cutting from each other by simply looking at them. Also they lack other kinds of domain knowledge	Since these temporary workers will be doing a lot of the packaging of the meat, it is important, that the prototype application for registering meat products can be operated with a minimum of domain knowledge	EBBITS-371
TNM LL WP10-4	Secondary processors do a lot of administrative paper work. Some are to satisfy legislation and ensure public health. Some are to safeguard against fraud, for example fraud with country of origin, whether the meat is organic and so forth. Currently almost all of this documentation (at least in Denmark) is handled manually and kept physically. If the butcher misplaces or otherwise lose any documents, he could very well be heavily fined or find himself unemployed. We have learned that there could be great advantages for both processors and authorities if the data was instead kept digital. The processors would no longer have the responsibility of handling all the paperwork and the authorities could improve inspection on many levels	We should ensure an interface for both of these user groups.	EBBITS-370
TNM LL WP10-5	In this traceability scenario you are not able to control the data sources. You will have to work with whatever the data owner presents to you. This can present a problem. And we have learned that even big successful companies can have not so brilliant IT-systems. A wide range of legacy systems still live long and healthy lives in the real world and 'databases' are not always structured in a way normally recognized as worthy of the tag database.	Data sources are messy not only on small farms but also within large corporations.	EBBITS-371
TNM LL WP10-6	Basic data in real life is often generated by humans and are therefore, due to a lot of different reasons, often error prone. Concrete examples include animals that are registered as butchered twice, missing identification and some data simply missing due to unforeseen circumstances.	It is therefore important that data import functions to the ebbits system are rugged and are aware of and takes care of these issues. For example by	EBBITS-371

		providing feedback to data generators about possible errors or otherwise try to remedy the effects of erroneous or missing data.	
COMAU LL WP10-7	The main issue highlighted during the setup for the demo server is that the industrial environment of enterprises such as COMAU has a very complex IT management behind it.	It is not so easy to setup ingoing connections to computers inside the enterprise intranet because of internal policies.	

4.8.1 Analysis of Lessons Learned

Six Lessons Learned were reported in the Food Traceability environment:

- The most important Lessons Learned in the Food Traceability environment are that the real world regarding foodstuffs and traceability is not perfect. There is a lot of old legacy hardware and software in this line of business. We will have to adjust to the fact that the processors have dated equipment and very little room for investment. And with regards to data, the available sources are not always up to the standard we normally work with. There are Access databases, non-existing web services, lacking and error prone data and lots of non-digitized paperwork out there in the real world. This is the basis for Lessons Learned LL WP10-3, LL WP10-5 and LL WP10-6 and to encompass this, requirement EBBITS-371 was created.
- Likewise we may foresee problems when trying to introduce new technology to the field. Some employees might not be ready to embrace a new way of working. We may also encounter problems if a change in technology requires more than a basic investment. This is reflected in Lessons Learned LL WP10-1 and LL WP10-2 and references requirement EBBITS-166.
- We have also registered an opportunity to broaden the scope of the framework. The ebbits core could help secondary processors by storing information that would normally be handled as paperwork. Authorities could then have an interface to these data, making inspection more effective. This is reflected in Lesson LL WP10-4, which resulted in the creation of requirement EBBITS-370.

The Automotive Manufacturing environment yielded one Lesson:

- **LL WP10-7:** the server has been setup properly but only with outgoing connections, it is not allowed to give access from the external to computers inside the intranet network.

4.9 Other Work Packages

Work packages WP1, WP2 and WP12 have collected no Lessons Learned in the second cycle, largely due to their non-technical nature. The work in WP11 has not started yet.

5. Requirements Engineering in the Second Cycle

A total of 56 Lessons Learned has been reported, validated and analysed in Section 4. Relative to the set of requirements listed in deliverable *D2.9.1 Updated requirements report 1*, this has resulted in 26 new requirements, 24 changed requirements and 40 deleted requirements originating from different work packages as detailed below.

5.1 Change request and reengineering originating from WP3

We have to focus on incorporation of all e3-value methodology viewpoints, i.e. the value viewpoint, the business process viewpoint and the information system viewpoint.

5.1.1 Lessons Learned

Two Lessons Learned have been reported, validated and analysed in the second cycle. The analysis has resulted in the creation of one new requirement.

5.1.2 New requirements

Key	Summary	Rationale
EBBITS-377	Deconstruction and reconstruction of created or existed value models should be viable to reflect and understand the variation in value created	Using deconstruction we are able to split a value model into smaller parts and reconstruction composes these parts in different ways. That enables us to reflect the possible variations in value created

5.1.3 Updated requirements

During the second cycle no requirement was updated.

5.1.4 Deleted requirements

No requirements were deleted.

5.2 Change request and reengineering originating from WP4

5.2.1 Lessons Learned

Six Lessons Learned have been reported, validated and analysed in the second cycle. The analysis has resulted in the following changes to the requirements.

5.2.2 New requirements

Key	Summary	Rationale
EBBITS-360	Measured data and the context model (as defined by WP5) are consistent	Consistency between abstract context model and concrete measured data
EBBITS-361	Fine grained access control policies have to be definable in the ebbits system	Merged from Requirements #398-#417, except #401+#405
EBBITS-	Semantic model for the manufacturing domain is	It is important to know when and where data were sensed/monitored. Generated messages and alerts need to be

362	created	traceable and provide rich information about the event detected
EBBITS-363	Semantic model for the traceability domain is created	A detailed annotation of feedstuff is required to the reasoning processes devised. Proper identification of animals and logging the most relevant information about their lives is vital for the traceability and quality control proposed in ebbits
EBBITS-364	Semantic model for the business process domain is created	ebbits platform can be exploited also for generic enterprise processes. Information exchanged between stakeholders could be exploited for some reasoning, thus it is convenient to model such exchange semantically and to forward or keep linked its metadata through the different stakeholders
EBBITS-365	The ebbits platform offers performant ad-hoc reasoning	Applications request information that needs to be inferred from the ontology and other knowledge sources through reasoning/information processing
EBBITS-366	The ebbits platform offers ad-hoc aggregation of knowledge	Knowledge is distributed in the ebbits network, and must be aggregated when a query requires so, transparently for the application
EBBITS-367	ebbits Access Control Policy will be respected by Ontology Manager	WP4 components need to respect WP8's access control policies.

5.2.3 Updated requirements

Key	Summary	Rationale
EBBITS-288	Monitored/sensed data should be contextualized (timestamp, geotag, type, etc)	The ebbits system annotates/tags each measurement according to the defined semantic models for sensed data which includes relevant classification and spatial-temporal metadata and relationships
EBBITS-290	Alerts should be contextualized (timestamp, geotag, type, message, warning level, etc)	Generated messages and alerts need to be traceable and provide rich information about the event detected
EBBITS-291	Devices should be annotated with id, type, name, location, and current/historical data (status, work in progress, consumables levels, quality record, energy consumption, energy profile, planned/unplanned intervention/maintenance, fault info, etc)	Another added value that ebbits could introduce in enterprise domains is efficiency tracking, which requires a monitoring and log of several metrics in devices/tools/machinery and resources in general
EBBITS-287	All stakeholders should be annotated with unique Id, type, name and relevant info	The ebbits system includes one or more directories of stakeholders or identity managers, including for each stakeholder annotations about id, type, name and relevant info
EBBITS-289	Monitored/sensed data should be (semantically) annotated in local server/repo/store.	ebbits data acquisition devices/proxies annotates sensed data according to metadata models locally available or requested to semantic stores.
EBBITS-309	ebbits platform should have a publish-subscribe system	The different monitored processes in ebbits should generate alerts and send them to the interested subsystems or stakeholders

5.2.4 Deleted requirements

36 requirements have been deleted.

Requirements EBBITS-288, EBBITS-290, EBBITS-291, EBBITS-292, EBBITS-293, EBBITS-294, EBBITS-295, EBBITS-296, EBBITS-297, EBBITS-298, EBBITS-299, EBBITS-300, EBBITS-301, EBBITS-302, EBBITS-303, EBBITS-304, EBBITS-305, EBBITS-306, EBBITS-307, EBBITS-308, EBBITS-310, EBBITS-311, EBBITS-312, EBBITS-313, EBBITS-314, EBBITS-315, EBBITS-316, EBBITS-317, EBBITS-318, EBBITS-319, EBBITS-320, EBBITS-321, EBBITS-322, EBBITS-323, EBBITS-324 and EBBITS-325 have all been closed as being Duplicates.

5.3 Change request and reengineering originating from WP5

5.3.1 Lessons Learned

Nine Lessons Learned have been reported, validated and analysed in the second cycle. The analysis has resulted in the following changes to the requirements.

5.3.2 New requirements

Key	Summary	Rationale
EBBITS-381	Self-* manager needs to monitor the connection to the physical devices	Proxy connection to the physical devices could be unstable
EBBITS-382	Device proxies can shut off a physical device from the network if it causes a lot of problem	Sensors causing a lot of problems could affect the stability of the proxy.
EBBITS-383	Device proxies adjust event publishing frequency according to the network bandwidth	Static frequency of sensor reading is not optimal when the network bandwidth fluctuates because it causes congestion when the bandwidth is currently slow
EBBITS-384	Device proxies reset devices upon problems when no other fix is defined by the developer	Sensors causing a lot of problems could affect the stability of the proxy; Reset is an easy way to recover the device states
EBBITS-385	Device Proxies need a standardized interface that provides control management services for event publication	A way to configure the event publication (topic, frequency, data fusion etc.) is needed to support adjusting data traffic according to the network

5.3.3 Updated requirements

Key	Summary	Rationale
EBBITS-178	Aggregating collected sensor data at a central point	The aggregation of collected data is important for analyzing the data
EBBITS-196	Diagnostic component to detect and correct malfunctions	If a malfunction has slipped in the plant it should be corrected ASAP. In fact, if possible any fault behaviour should be prevented at all
EBBITS-200	Distributed data can be referenced in data fusion and context management	Data is spread across several instances of ebbits, even several enterprises. In context management, data fusion, and also semantic querying, distributed data needs to be referenced
EBBITS-250	Support runtime reconfiguration	To supporting monitoring leading to adaptation, the architecture should be dynamic in the sense that components/services should be connectable at

		runtime
EBBITS-291	Devices should be annotated with id, type, name, location, and current/historical data (status, work in progress, consumables levels, quality record, energy consumption, energy profile, planned/unplanned intervention/maintenance, fault info, etc)	Another added value that ebbits could introduce in enterprise domains is efficiency tracking, which requires a monitoring and log of several metrics in devices/tools/machinery and resources in general
EBBITS-309	ebbits platform should have a publish-subscribe system	The different monitored processes in ebbits should generate alerts and send them to the interested subsystems or stakeholders
EBBITS-327	Sensor fusion algorithm must be added during run-time in a modular and extensible way	Sensor fusion algorithms vary greatly and can't be generalized only in one module
EBBITS-328	Sensor fusion algorithms must be realized as a decoupled component	Sensor fusion algorithms can be re-used by several other components
EBBITS-332	Context management should be able to process a large number of sensor events	A Manufacturing site has at least 500 sensors, where each one raises about 1 event per second. All these events have to be processed by context management

5.3.4 Deleted requirements

No requirements have been deleted.

5.4 Change request and reengineering originating from WP6

5.4.1 Lessons Learned

Four Lessons Learned have been reported, validated and analysed in the second cycle. The analysis has resulted in the following changes to the requirements.

5.4.2 New requirements

Key	Summary	Rationale
EBBITS-368	Service descriptions include operational and business aspects	Bring together business, operational and technical aspects of services into one single coherent language
EBBITS-369	Query the main data in web services and the ontologies in a convenient, combined way	It should be possible to pose combined queries to the background knowledge in form of ontologies and data coming from web services

5.4.3 Updated requirements

Key	Summary	Rationale
EBBITS-213	System should show Energy Cost for different granularity of production processes	Energy cost at different levels is needed to do benchmarking of operational processes
EBBITS-215	Adjust production processes according to energy price policies	Reduce production cost by taking into account energy price policy from energy provider
EBBITS-232	Recognition of energy wasting behaviours	Help decision makers to optimize energy usage

EBBITS-281	Explicit model of context	It must be possible to trace events and data items across processes and workflows, context management is one of the mechanisms to support this
EBBITS-286	Events mapped to (business) rules	Events and services are basic mechanisms for the implementation of the (business) rules logic in the ebbits architecture
EBBITS-330	Applications can monitor the state of devices and context entities	Continuous monitoring of context entities (e.g., pigs, welding guns) can be used to detect anomalies (e.g.: ill Pigs, overheated welding gun)
EBBITS-331	System needs to trigger business events based on changes of devices and entities states	Enterprise applications have to be notified when the process starts and finishes, and further how much resources have been consumed for the process
EBBITS-337	Semantic event processing	It must be possible to interpret events in the context of the different layers in the architecture (from PWAL to a business rules layer)

5.4.4 Deleted requirements

No requirements were deleted.

5.5 Change request and reengineering originating from WP7

5.5.1 Lessons Learned

Six Lessons Learned have been reported, validated and analysed in the second cycle. The analysis has resulted in the following changes to the requirements.

5.5.2 New requirements

Key	Summary	Rationale
EBBITS-378	Entity structure accessible from event rules	The Event Processing Agent needs the entity structure for writing rules over complex entities
EBBITS-379	Navigable entity IDs	The Event Processing Agent needs to be able to find related entities instances from a given entity id
EBBITS-380	Using Process IDs in Events	The Event model processID needs to be mapped to the corresponding process model.

5.5.3 Updated requirements

No requirements are changed with respect to their content.

5.5.4 Deleted requirements

No requirements were rejected.

5.6 Change request and reengineering originating from WP8

5.6.1 Lessons Learned

Twenty Lessons Learned have been reported, validated and analysed in the second cycle. The analysis has resulted in the following changes to the requirements.

5.6.2 New requirements

Key	Summary	Rationale
EBBITS-372	PWAL framework must scale efficiently when multiple drivers are executed at runtime	Multiple PWAL drivers might be running inside the same PWAL instances; the coexistence of multiple drivers should not affect performance. The core framework currently should provide methods to divide PWAL workload if needed
EBBITS-373	Out of channel event management is needed	Due to poor IT reliability, it is not always possible to programmatically react to channel problems or errors. It is supposed to just trigger channel events. Out of channel reporting of network events is preferred to automatic reaction to problems. Out of channel alarm mechanisms are needed
EBBITS-374	Support for multiple identification schemes	Application uses different identification schemes. In this way ebbits should support both the existing own identification scheme and other possible future identity management schemes. It is necessary to support an association between identifiers (while storing and querying them), which should be managed by the Entity Manager
EBBITS-375	Robot Controller PWAL driver must support multiple clients	The Robot Controller server does not support multiple clients. As a consequence, the Robot Controller PWAL driver should provide mechanisms to manage requests from multiple clients e.g. by supporting queuing
EBBITS-376	Robot Controller must not freeze when clients interrupts communication	Robot Controller must not freeze when clients interrupts communication; more specifically, it may happen that the Robot Controller PWAL driver is interrupted or restarted for maintenance
EBBITS-387	Support to describe company security policies and rules must be provided	Sometimes companies have restricted policies for information sharing and network traffic management. ebbits should provide means to describe such policies and rules programmatically, so that they can be enforced or at least understood automatically by ebbits security framework
EBBITS-388	Provide support to logging of information from physical-world devices and sub-systems	Logging mechanisms are used by companies to check correct functioning of devices and subsystems. Since logging at device level generates a great amount of information which is not needed continuously, configurable and flexible methods should be provided to control logging features. Among the logged information, additional meta-data about the lack or corruption of information from the field should also be included
EBBITS-389	Support to proxy authentication through NTML	LinkSmart should support strict security policies in enterprise networks
EBBITS-390	PLC driver supports symbols recursive discovery	PLC developers often rely on nested structures to organize variables and signals; the PLC PWAL driver should be updated to support recursive symbols discovery
EBBITS-391	PWAL Robot Controller driver should support automatic discovery of available axes	Information about available axis is needed from the Robot controller to support autonomous discovery of robot capabilities
EBBITS-392	Robot Controller PWAL driver must support the automatic parIDs mapping into variables	The Robot Controller itself does not support an automatic map of parIDs to variables and so a mechanism to receive and retrieve a list with the semantic information regarding the available parIDs is needed within the PWAL Robot controller driver
EBBITS-	Meta-data about missing	Sometimes field information is missing or corrupted e.g. because the

393	or corrupted information should be managed	user has not entered it correctly or because of malicious behaviour. ebbits should be aware and able to report that some information is missing or corrupted

In addition requirement **EBBITS-361** has been created, see Section 5.2.2 under WP4.

5.6.3 Updated requirements

No requirements have been updated.

5.6.4 Deleted requirements

EBBITS-348 and EBBITS-351 have been closed as being Out of Scope.

5.7 Change request and reengineering originating from WP9

5.7.1 Lessons Learned

Two Lessons Learned have been reported, validated and analysed in the second cycle. The analysis has not resulted in any changes to the requirements.

However, we foresee a revision and update of requirements as the work package gets closer to the SDK (Software Development Kit) phase.

5.7.2 New/updated/deleted requirements

No requirements have been added, updated or deleted.

5.8 Change request and reengineering originating from WP10

5.8.1 Lessons Learned

Seven Lessons Learned have been reported, validated and analysed. This resulted in the creation of two new requirements, update of one requirement and the deletion of two requirements.

5.8.2 New requirements

Key	Summary	Rationale
EBBITS-370	Access to traceability data from legislative actors	A lot of administrative work can be avoided if public control instances can access the traceability data
EBBITS-371	Data sources are not perfect	Data are typically at some point entered by hand. Also databases found at slaughterhouses feature legacy systems with non-optimal structure and possible inconsistencies

5.8.3 Updated requirements

Key	Summary	Rationale
EBBITS-166	Product labels / tags for the traceability should be readable by legacy system of the stakeholders	The stakeholders do not want to invest for buying new label / tag readers

5.8.4 Deleted requirements

Two requirements have been resolved; EBBITS-170 because it is Out of Scope and EBBITS-175 because it cannot be implemented.

6. Validation Results

6.1 Summary of verification results

No verification results have been reported by the technical work packages.

6.2 Summary of validation results

The applications are still at an early prototype level, and no end users have so far been involved in validation testing.

6.3 Summary of results from usability testing

No usability testing has been done on the prototypes at this stage.

6.4 Summary of outcomes of field trials

No field trials have been performed in the second development cycle.

7. Impact Assessment

7.1 Impact on overall architecture

At mid-term of the project, the first version of the overall technical ebbits architecture is consolidated. The second platform prototype at M22 (following the proof-of-concept prototype at M12) implements a subset of the specified components of this architecture. The (application) domain analysis and requirements engineering during the second cycle have led to more focused application examples in the Food Traceability and the Automotive Manufacturing scenarios (M24 demos).

Revisiting the project's original objective of promoting the ebbits platform as an enabling technology in product life cycle management, functionality and components supporting this wider perspective still remain to be implemented and to be made visible.

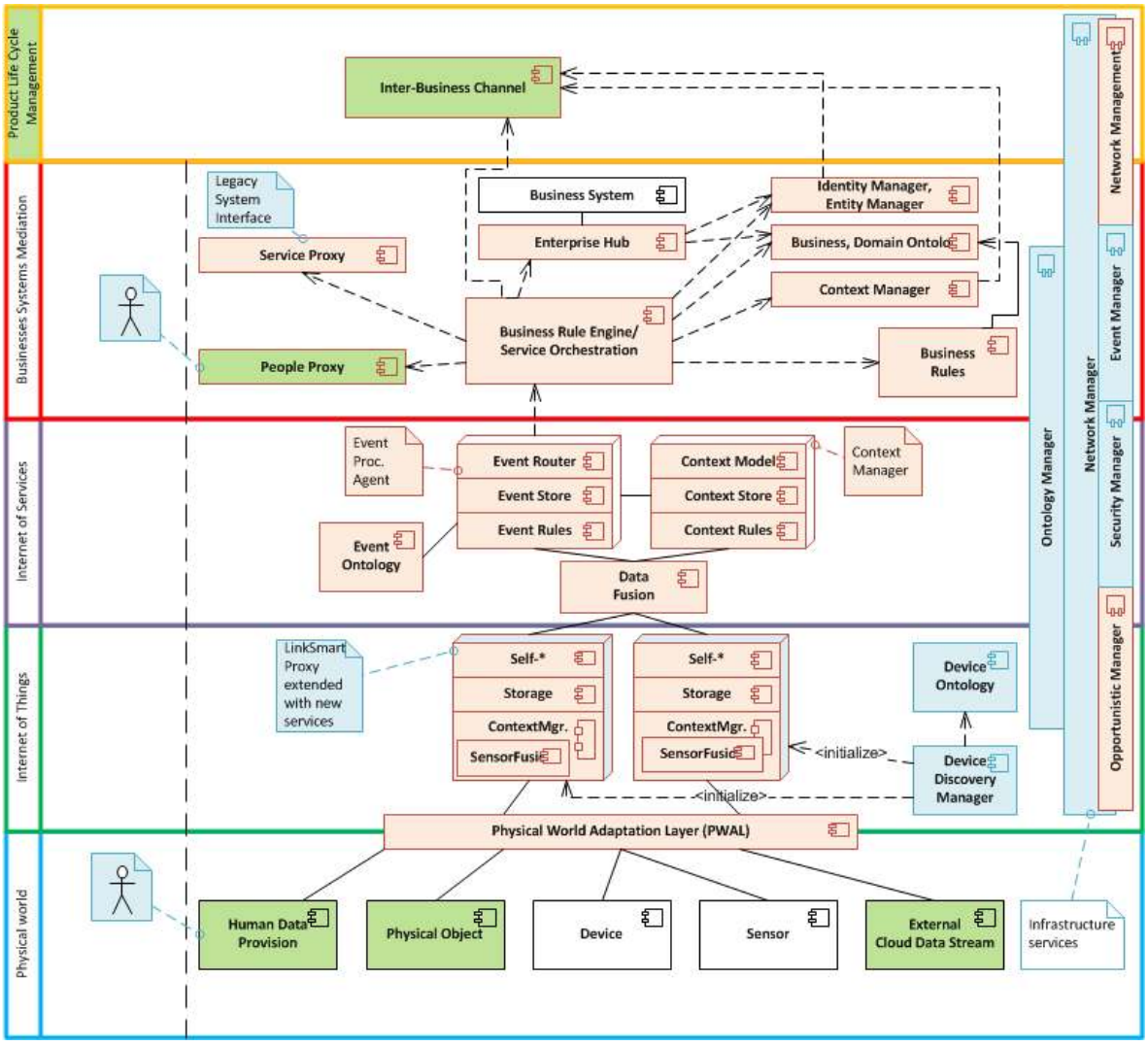

Figure 2: Envisioned architecture for Year 3 iteration, with green boxes representing new modules

At the end of this iteration, Lessons Learned and the new or revised requirements will underpin further design and implementation decisions with respect to the platform. For the Year 3 iteration this will involve the areas identified below:

- *Product Life Cycle Management* – Knowledge of process semantics and structure can be exploited for value creation in a business analysis phase as well as in the encoding of those business and operational rules that control the execution of the applications

- *Business System Mediation* – Technical and non-technical aspects need to be covered. This aspect was addressed at the end of this iteration by the analysis and introduction of a higher-level description framework and tools (USDL[2] from SAP). Orchestration involving the Things and People in the IoPTS should be investigated
- *Internet of Services* – Device context and distributed context are issues that will be further pursued in the third iteration, in order to make first-line event processing scalable. The design of the current architecture for both the context management and for event processing is corroborated by the Lessons Learned and the requirements from the two application domains. Data fusion mechanisms need further development.
- *Internet of Things* – This layer is the most developed and well understood since it largely builds on the LinkSmart middleware. It provides mechanisms so that the underlying devices, sensors, and external services become services that can be used by platform applications. Data capture needs to be done close to sources. The work on self-* manager needs further elaboration and testing in real-world situations. New IoT proxy objects need to be defined and developed for the new entities (people, real-world objects, external cloud streams) provided by PWAL.
- *Physical World* – The PWAL needs to model objects (or entities) in the real world such as product items and not only devices and sensors and provide a uniform interface for application developers to use these objects in their applications. Moreover, solutions for integrating "people" into the platform need to be developed. Furthermore, the rapid development in cloud-based sensor and data services needs to be supported, allowing the ebbits platform to connect to and make use of available data streams from services like Cosm[3].

7.2 Impact on architecture for Automotive Manufacturing and Food Traceability

Both application areas would benefit from making the ebbits platform infrastructure more visible. This is will help to better target the overall objective of life cycle management, making visible process, people, things and timeline.

For the next prototypes, this could be done by demonstrating the capabilities provided by the different components in the platform architecture, such as

- The process perspective should be elaborated and made more visible in the application design
- To a greater extent consider the P(eople) in the IoPTS world of ebbits
- The use of naming and identity schemes should be made explicit
- The SDK functionality, i.e., what developer functionality and tools are needed in order to efficiently design and implement this type of applications and others

Considering traceability, the project could even further emphasize the possibilities of improved resolution in food traceability, where traceability datasets are targeting properties of the specific product and the context and properties of the individual animal (organism). This is an improvement to current traceability systems which are focused on the properties of a batch of products. For the manufacturing application, the T(hings) in the IoPTS could be made more explicit.

7.3 Impact on individual work packages

A consolidated design and implementation path for process representation and accessibility would be beneficial. This would at least include work packages 3, 6 and 7, coordinated by WP3. Furthermore a discussion with regard to the Lessons Learned in WP9 should be held with the other technical work packages.

[2] Unified Service Description Language
[3] https://cosm.com

8. Appendix 1 – JIRA workflow for ebbits