Enabling the business-based
Internet of Things and Services

(FP7 257852)

D2.9.1 Updated requirements report 1

Published by the ebbits Consortium

Dissemination Level: Public

European Commission
Information Society and Media

SEVENTH FRAMEWORK PROGRAMME

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

Document control page

Document file: D2.9.1 Updated requirements report 1
Document version: 1.0
Document owner: Amro Al-Akkad (Fraunhofer FIT)

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	Amro Al-Akkad (FIT)	2011-07-18	Initial Structure
0.2	Ferry Pramudianto (FIT)	2011-10-07	Impact on WP5 (sub chapter 4.2)
0.3	Paolo Brizzi, Mauricio Caceres (ISMB)	2011-10-11	Impact on WP8 (sub chapter 4.5)
0.4	Amro Al-Akkad (FIT)	2011-10-14	Requirements of WP5, WP6, WP7 WP8 to subchapter 3.2, 3.3, 3.4 and 3.5
0.5	Peeter Kool, Matts Ahlsen (CNet)	2011-10-17	Impact on WP7 (sub chapter 4.4)
0.6	Martin Knechtel (SAP)	2011-10-17	Impact on WP4 (sub chapter 4.1)
0.7	Yves Martin (SAP)	2011-10-18	Impact on WP6 (sub chapter 4.3)
0.8	Amro Al-Akkad (FIT)	2011-10-18	Requirements of WP4 to subchapter 3.1 and complete list of open requirements to chapter 6
0.9	Amro Al-Akkad (FIT)	2011-10-19	Executive Summary and Conclusion.
1.0	Amro Al-Akkad (FIT)	2011-10-26	Reviewer comments addressed, editing
		2011-10-31	Final version submitted to the European Commission

Internal review history:

Reviewed by	Date	Summary of comments
Gonzalo Alcaraz (ISMB)	2011-10-21	Approved with minor revisions
Jesper Thestrup (In-Jet)	2011-10-25	Approved with comments

Index:

1. Executive summary

As a tool for the collaborative creation and maintenance of requirements the ebbits project consortium decided to utilize the tracker component of the GForge[1] platform. Project members were asked to create, update or delete requirements based on their findings from user workshops, group discussions, architectural workshops and further.

During the first iteration of the ebbits project several requirements have been created and some have been updated. Currently, most requirements remain in an open status, but now their definition has been consolidated. The main activity for verifying and refining requirements has been the reporting of the LLs resulting from the first iteration cycle and the usability evaluation (see D2.7.1) and the respective change requests (see D2.8.1). In detail, the report, validation and analysis of the LLs yielded for each WP the following changes:

- WP4 created 40 new requirements and updated exactly one,
- WP5 created 14 new requirements and also updated one,
- WP6 created one new requirement,
- WP7 created six new requirements and again updated one, and
- WP8 created 18 new requirements and further updated three.

For the second and next iteration the main target for all technical WPs is to start with implementing the collected and refined set of requirements and where possible to accomplish their implementation, and thus to reduce the number of open requirements.

[1] GForge ® Advanced Server, http://gforge.org/gf/

2. Introduction

2.1 Purpose, context and scope of this deliverable

This document presents the first update of ebbits system requirements, and thus, represents the first version of the D2.9 Updated Requirements Report.

The deliverable gives an overview of the refined set of requirements that will be used within the remaining iterative steps according to the user-centred design approach and methodology adopted in all phases of the project. It takes into account all requirements that have been created or updated since the start of the ebbits project in September 2010 and provides an analysis of the impact that these updates do have on the individual work packages (WPs). Thereby, it provides an outlook on the activities that have to be performed in the next iteration coming in 2011-2012.

2.2 Background

We strongly recommend the reading of the first Lessons Learned (LLs) and results of usability evaluation (D2.7.1), the first change request and reengineering report (D2.8.1), and further as sort of background knowledge the initial requirements report (D2.4) or validation framework (D2.6).

2.3 Structure of this document

The report is structured as follows: Chapter 2 gives an overview of the work described in this document and summarizes the effects on the architecture and WPs. Chapter 3 presents all new and updated requirements since M1. In chapter 4, the impact on each WP is described in detail. Finally, chapter 5 gives a brief summary, and chapter 6 provides the complete list of open requirements.

3. Updated requirements for ebbits

This section contains the condensed list of requirements that have been updated, deleted, or created since the start of the ebbits project within the technical WPs, i.e. WP4, WP5, WP6, WP7 and WP8. Each requirement listed in the following tables has a unique ID that allows referencing. According to the Volere[2] scheme, the requirements are divided into non-functional and functional requirements. Each requirement is assigned to a component, e.g. architecture, context, security etc. The summary of a requirement is a synthetic but clear description of the requirement. The need for fulfilling a particular requirement is prioritized into five levels: 1-high, 2-medium high, 3-medium, 4-low, 5-very low. The rationale gives a reason why this requirement is relevant for the ebbits platform. The fit criterion is needed to measure to which extent a requirement is met, e.g. in percentage. Not least, the status of a requirement indicates if it is open, implemented, ambiguous, rejected, etc.

[2] Volere Homepage: Volere Requirements Resources, http://www.volere.co.uk/

3.1 Requirements of WP4 – Semantic Knowledge Infrastructure

3.1.1 Architecture

ID	Requirement Type	Components	Summary	Priority	Rationale	Fit Criteria	Status
43	Functional	Architecture	Aggregating collected sensor data at a central point	1	The aggregation of collected data is important for analyzing the data.	A framework is provided that aggregates collected sensor data at a central point of an application.	Implemented
138	Constraint: Scope of the product	Architecture	The system should support distributed intelligence on embedded system.	1	We have a need for intelligence (Semantics, reflection etc.). We have a need for supporting embedded systems. This should not conflict	Minimum hardware requirements (which must be supported by all target hardware) are defined and all hardware that meets the specifications is guaranteed to work with LinkSmart.	Open
154	Functional	Architecture	Aggregate data from various data bases and sources	3	Information will be stored in several places, but needs to be combined in some place and assigned to the actual product or entity.	A data aggregation component is available.	Duplicate
451	Functional	Architecture	Sensor fusion algorithm must be added during run-time in a modular and extensible way.	2	Sensor fusion algorithms vary greatly and can't be generalized only in one module.	New sensor fusion algorithms can be added in a pluggable way	Open
452	Non-Functional: Maintainability	Architecture	Sensor fusion algorithms must be realized as a decoupled component.	3	Sensor fusion algorithms can be re-used by several other components.	Unrelated components are able to use sensor fusion modules	Open
456	Functional	Architecture	The system should be able to process a large number of sensor	3	A Manufacturing site has at least 500 sensors	System is able to process at least 500 events / second.	Open

			events				
43	Functional	Architecture	Aggregating collected sensor data at a central point	1	The aggregation of collected data is important for analyzing the data.	A framework is provided that aggregates collected sensor data at a central point of an application.	Implemented

3.1.2 Communication

ID	Requirement Type	Components	Summary	Priority	Rationale	Fit Criteria	Status
52	Functional	Communication	Interoperability needs to be created between various subsystems in the manufacturing area	2	The subsystems in manufacturing environments are currently not interconnected and not able to exchange information.	Three independent subsystems from the manufacturing area can exchange information (show a use case)	Open
123	Functional	Communication	Information needs to be described in a standardised way	3	Enterprises working in the same sector adapt different ways to describe the input, the production processes, and the output; thus it will not be possible to communicate information either to providers or to consumers.	A standard for exchanging information between system components is provided.	Open
163	Functional	Communication	The traceability chain should be computed on demand	3	The traceability chain should be computed on demand and not stored and recalled, due to storage. Thus, the combination of data is the challenge and a handover of identities might be required. A combination of	The traceability chain is computed and not stored.	Open

					searches should be realized by semantics.		

3.1.3 Context

ID	Requirement Type	Components	Summary	Priority	Rationale	Fit Criteria	Status
64	Functional	Context	Historical data should be recorded persistently.	1	Quality is very important inside an assembly line as it is the essential parameter used for force tests or lack tests. Furthermore, if failures are detected lately when a car is already in the market, but shows some lack, the production history can be traced to find the devil in the detail.	Quality related information is logged inside a proper carrier medium.	Open
380	Non-Functional: Operational	Context	Monitored/sensed data should be contextualized (timestamp, geotag, type, etc).	3	It is important to know when and where data were sensed/monitored.	Semantic store with knowledge model for sensor readings.	Open
382	Non-Functional: Operational	Context	Alerts should be contextualized (timestamp, geotag, type, message, warning level, etc).	3	Generated messages and alerts need to be traceable and provide rich information about the event detected.	Alerting and messaging policy.	Open

3.1.4 Devices

ID	Requirement Type	Components	Summary	Priority	Rationale	Fit Criteria	Status
132	Functional	Devices	Device and service exception handling	3	The development and run-time environment should support exception handling constructs that the developer can employ to manage service	Exception handling constructs that the developer can use to specify exception responses with a success rate of 9/10.	Open

					and device availability and malfunctioning, isolated from the main application logic.		

3.1.5 Interface

ID	Requirement Type	Components	Summary	Priority	Rationale	Fit Criteria	Status
405	Non-Functional: Operational	Interface	Feed Provider should transfer delivery information about sent feedstuff.	2	The traceability relays on the successful exchange of information about the monitored processes linked to the tracked product.	Interface/procedure for request/receive/exchange of relevant information about deliveries.	Open
415	Non-Functional: Operational	Interface	Accounting Management System should store orders, and have access to Supplier Management System (to send/receive orders/acks/invoices).	4	The ebbits paradigm can be exploited also for improving the efficiency of accounting task.	Access policies to Supplier MS.	Open
416	Non-Functional: Operational	Interface	Accounting Management System should have access to bank's management system for sending/receiving payment orders/confirms.	4	The ebbits paradigm can be exploited also for improving the efficiency of accounting task.	Access policies to Banks' MS.	Open
466	Non-Functional: Usability	Interface	Query the ontologies conveniently	3	Easy query possibility	SPARQL is used as query language, or an abstraction is used if this turns out to be more usable.	Open

3.1.6 Modelling

ID	Requirement Type	Components	Summary	Priority	Rationale	Fit Criteria	Status
42	Functional	Modelling	semantic relationships between data	1	Currently, any data is stored in a simple database. Hence, data	Data can be queried and inferred in order.	Open

					is available, but cannot be interrelated intelligently.		
71	Functional	Modelling	Definition of smallest unit can be traced or uniquely identified	2	Small parts of products / cheap parts of products cannot be identified anymore because of physical limitation of tags . Price of tags is too expensive to tag cheap parts.	Clear definition in both domains of what can be tagged.	Open
136	Functional	Modelling	Handling of different device versions in device ontology	3	The device ontology should be able to handle different versions of a device.	The device ontology can maintain at minimum 2 versions of any single device	Open
142	Functional	Modelling	Download and harmonisation of third party device ontologies	1	Device ontological models describing devices, which will be provided by manufacturers or third parties, should be automatically downloaded (updated) and harmonised to ensure the same ontological view. Formal definition of ontologies should be realised using the world wide accepted formats, recommended by W3C, such as RDF, OWL, OWL-S.	Ontologies from different manufacturers can be used if they are in RDF, OWL or OWL-S.	Open
147	Functional	Modelling	waste of energy act definitions	3	Some users are wasting energy without realizing/being conscious that there are better alternatives.	Energy wasting behaviours are modelled	Open
151	Functional	Modelling	Common structure of information is needed	3	Relevant data to describe specific situations of a pig needs to be determined and put into a common structure such as an	Common data structure to describe situations of arbitrary entities is available	Open

					ontology, a common definition of data.		
152	Functional	Modelling	The system should allow the correlation of information emerging from several sources	3	In order to easily analyse information, the system should allow for the correlation of information from different sources on a farm or enterprise	Acquired information is timestamped.	Open
381	Non-Functional: Operational	Modelling	Monitored/sensed data should be annotated (semantically) in local server/repo/store.	3	Information relationships should be available as soon as data enters the ebbits system.	Data acquisition, annotation and storing policy.	Open
383	Non-Functional: Operational	Modelling	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).	3	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.	Semantic store with knowledge model for devices. There is a unique mapping between physical devices and their semantic reflections.	Open
384	Non-Functional: Operational	Modelling	Production should be annotated or modelled in order to calculate OEE and get number of orders/elements/products requested/delivered/in-progress/faulty.	2	Real-time traceability of produced goods/services is achieved by properly annotating their status and metrics during the manufacturing process.	Semantic store with knowledge model for production.	Open
385	Non-Functional: Operational	Modelling	Logistic should be annotated or modelled in order to get information about element and consumables (present, in transit from supplier, ordered, etc).	3	The ebbits platform could provide also a system for efficient management of consumables and logistic aspects needed in (not only) manufacturing domains.	Semantic store with knowledge model for logistic.	Open
386	Non-Functional: Operational	Modelling	Feedstuff should be annotated with origin, genetics, treatment,	2	A detailed annotation of feedstuff is required to the reasoning	Semantic store with knowledge model for feedstuff.	Open

			storage conditions and transport/delivery info (batch number, silo id, amount, timestamp).		processes devised.		
387	Non-Functional: Operational	Modelling	Animals should be annotated with RFID tag, weight, genetics, birth date, and current/historical (timestamped) data (growth/weight, location/movements, consumed feed, water, weaning, insemination, heat during pregnancy, born piglets, anomalies, vaccines).	2	Proper identification of animals and logging the most relevant information about their lives is vital for the traceability and quality control proposed in ebbits.	Semantic store with knowledge model for animals. Logging policy.	Open
388	Non-Functional: Operational	Modelling	Meat packages should be annotated with ID of animal (for trace).	3	Meat traceability is one of the main added values of the ebbits platform in the agricultural domain.	Semantic store with knowledge model for animals.	Open
389	Non-Functional: Operational	Modelling	Farm's soil/fields should be annotated with location, laboratory analysis info (date, sample field source, lab id/name, results, etc), current/historical data (types of crops, grain maturity, soil nutrients, quality of products grown, etc).	3	Different reasoning applications devised in ebbits for tracking the soil efficiency require a detailed annotation and logging of farms' soil.	Semantic store with knowledge model for farm soil/fields.	Open
391	Non-Functional: Operational	Modelling	Sow farm production should be annotated or modelled in order to allow tracking the number of piglets, pigs at fertile age, pigs ready to slaughter, maximum capacity, etc.	2	By proper reasoning and processing, the ebbits platform can exploit the knowledge in the network and extract aggregated information required in real time.	Reasoning algorithms for production tracking.	Open
392	Non-Functional: Operational	Modelling	Halves of slaughtered pigs should be annotated with id, id of slaughtered pig, weight, fat thickness,	2	Traceability of meat requires proper tracking of pigs since birth to stores, thus the	Semantic store with knowledge model for halves of slaughtered pigs.	Open

			date of slaughter, price, etc.		information after its slaughter is very relevant.		
393	Non-Functional: Operational	Modelling	Order should be annotated with type/ID of good/service, amount, price, dates(issue, expiry, delivery, etc).	4	ebbits platform can be exploited also for generic enterprise processes, like account management.	Semantic store with knowledge model for orders.	Open
394	Non-Functional: Operational	Modelling	Invoices should be annotated with supplier's info (name, id, bank account, contacts, etc), goods or services info (type/id, amount, price, dates, etc).	4	ebbits platform can be exploited also for generic enterprise processes, like account management.	Semantic store with knowledge model for invoices.	Open
395	Non-Functional: Operational	Modelling	Payment reports should be annotated with responsible id/name, bank, order number, status, etc.	4	ebbits platform can be exploited also for generic enterprise processes, like account management.	Semantic store with knowledge model for payments.	Open
396	Non-Functional: Operational	Modelling	Generic Information should be annotated with requester, sender, content (price, capacity, dates), etc.	4	Information exchanged between stakeholders could be exploited for some reasoning, thus it is convenient to model it semantically.	Knowledge model for information exchange.	Open
402	Non-Functional: Operational	Modelling	Manufacturing Monitor System must have read access to internal and external environment data.	3	The reasoning processes devised by ebbits for the manufacturing domain require environmental monitoring.	Internal/external sensors collected and annotated in respective repositories.	Open
464	Functional	Modelling	OWL-lite will be used to model ontologies	3	Appropriate knowledge representation formalism is needed to avoid the high complexity in the knowledge manipulation process.	The knowledge representation formalisms must be as small and easy as possible, but as expressive as necessary for our scenarios.	Open
465	Functional	Modelling	Ontology namespace	3	Common ontology namespace is important.	ebbits ontologies will share namespace http://www.ebbits-project.eu/ontologies	Open

467	Functional	Modelling	Only relevant parts in the ebbits ontologies	3	The development of the semantic models must be driven by the real use-cases instead of the theoretical assumptions to avoid the unnecessary complexity in the knowledge.	Ontologies are relevant for ebbits use cases.	Open

3.1.7 Security

ID	Requirement Type	Components	Summary	Priority	Rationale	Fit Criteria	Status
379	Non-Functional: Operational	Security	All stakeholders should be annotated with unique Id, type, name and relevant info.	2	It is important to recognise the users interacting with the system and their privileges and restrictions.	One or multiple directories of stakeholders	Open
397	Non-Functional: Operational	Security	Stakeholders should be stored in local catalogues or external directories (advisory company, chamber of commerce, etc) and accessed by the different subsystems inside and outside ebbits.	2	In order to apply access control lists/policies, all stakeholders must be identified.	Stakeholder directories.	Open
398	Non-Functional: Operational	Security	Manufacturing Monitor System should have write access to local server/repo/store.	2	The reasoning processes devised in ebbits require access to knowledge/information found in local stores.	Access rules/policy granted for MMS.	Open
399	Non-Functional: Operational	Security	Manufacturing System for Analysis should have read/write access to local server/repo/store.	2	Analysis/reasoning is based on local monitored data, and reports are sent back to local server/repo/store.	Access rules/policy granted for MMS.	Open
400	Non-Functional:	Security	Reports should	3	Aggregated data,	Access rules/policy	Open

	Operational		have a list of allowed readers/subscribers.		reports, alerts, etc, should be available only to stakeholders interested in them.	for generation/reading of reports.	
403	Non-Functional: Operational	Security	Maintenance Manager and operators should have access to devices' and production info (proper ACL have to be implemented)	2	Some of the added values that ebbits will provide to managers in the manufacturing domain require a continuous tracking of the production processes, metrics and modifications introduced.	Stakeholders access policies for readings, devices and production.	Open
409	Non-Functional: Operational	Security	Consumer should have access to meat reports.	3	Relevant reports about the produced meat since the piglet born will be used by ebbits in order to enhance the information provided to consumers about the meat they are buying.	Access policies to production reports.	Open
410	Non-Functional: Operational	Security	Controller should have access to (online) queries to meat packages IDs.	3	Quality and health control authorities can rely on ebbits in order to track the distribution of meat packages when they discover some anomaly.	Access policies to production reports.	Open
411	Non-Functional: Operational	Security	Farm's Management System should have access to field info repository.	3	Optimisation of the resources, like fields in an agricultural domain can be achieved by analyzing and processing information logged by their respective monitoring systems.	Access policies to historical info by Farm's MS.	Open
414	Non-Functional: Operational	Security	Slaughterhouse Management System and Retail	3	Accounting though ebbits will be simplified thanks to	Access policies to production and slaughter	Open

			Management System should have access to both production (read) and slaughter (write) repositories.		the knowledge acquired by multiple systems in the food production chain.	servers/repos/stores.	

3.2 Requirements of WP5 – Centralised and Distributed Intelligence

3.2.1 Architecture

ID	Requirement Type	Components	Summary	Priority	Rationale	Fit Criteria	Status
390	Non-Functional: Operational	Architecture	Farm's repository should store information about harvesting equipment, man power, etc.	3	The ebbits platform would provide also some functionalities for the management of resources needed for harvesting, thus they need to be included in the knowledge model.	Semantic store with knowledge model for farm resources.	Open
401	Non-Functional: Operational	Architecture	ebbits platform should have a list of alerts and subscribers.	3	The different monitored processes in ebbits should generate alerts and send them to the interested subsystems or stakeholders.	Directory of alerts/events.	Open
404	Non-Functional: Operational	Architecture	Farm's Management System should have access (through secure connection) to Feed Provider Resources Monitoring System.	3	The food traceability scenario requires an exchange of information between all the enterprises involved in the production chain.	Authenticated/secure access to Feed Provider RMS.	Open
406	Non-Functional: Operational	Architecture	Farm's Local server/repo should be accessible by RFID tag readers and National servers/repos/stores.	2	Information about animals is stored in farm local servers and used to retrieve information when reading RFID tags or when requested by National/European authorities.	Access policies to local server	Open

407	Non-Functional: Operational	Architecture	Farm's Monitoring System should have access to local server/repo/store.	2	Monitoring system keep track of several process in the farm and needs the metadata stored in local server.	Access policies to local server.	Open
408	Non-Functional: Operational	Architecture	Farm's Management Application Server should access local monitoring system servers/repos/stores for generation of reports	3	Management/accounting systems perform their tasks based on the information stored/provided by local monitoring systems.	Access policies to local monitoring system.	Open
412	Non-Functional: Operational	Architecture	Farm's Management System should have access to external information (crop price, fertilizers price, consumables price, weather, etc).	4	Consumables information can be exploited through ebbits in order to manage efficiently the farm's production processes.	Access to external/providers information.	Open
413	Non-Functional: Operational	Architecture	Sow Farm Management System should have access to production/animal repository.	3	The knowledge obtained by tracking all production processes in the farm will allow managers to optimise them through a single platform.	Access policies to local server/repo/store.	Open

3.2.2 Communication

ID	Requirement Type	Components	Summary	Priority	Rationale	Fit Criteria	Status
36	Functional	Communication	Controlling of machines/stations in manufacturing plant remotely	3	To optimize production process.	Relevant stations that operate automatically can be started / stopped via remote calls.	Incomplete
67	Functional	Communication	Automatic analysis of cross enterprises product life cycle data	3	Searching production problem from end costumer complaints need to track back data from several enterprises and logistic.	Analyzing data cross enterprises can be done online and automatically.	Open

3.2.3 Configurability

ID	Requirement Type	Components	Summary	Priority	Rationale	Fit Criteria	Status
47	Non-Functional: Operational	Configurability	Resilience and adaptable to environment condition changes	2	Environmental changes such as lighting, temperature affect the results of manufacturing process. so far machines are tuned manually by technicians. adapting to environmental condition can lead to reducing energy consumption e.g.: reduce heater temperature when it's warm outside.	Machines can adapt its parameters adapting to environmental changes.	Open
79	Non-Functional: Operational	Configurability	Location tracking should be implemented as independent app	2	Decoupling from existing system	Tracking system is implemented independently	Incomplete
134	Functional	Configurability	The context model needs to be extensible during runtime.	2	A knowledge model enables the middleware to contain a representation of itself and manipulate its state during its execution. This feature should serve as the basis for self-adaptation of the middleware (e.g. reconfiguration of resource usage, triggering the component-based services).	Middleware is able to adapt its configuration in 60% of identified cases requiring reconfiguration.	Open
139	Functional	Configurability	Support runtime reconfiguration	3	To supporting monitoring leading to adaptation, the architecture should be dynamic in the sense that components/services should be connectable	Services and devices can be connected during runtime.	Open

					at runtime.		

3.2.4 Context

ID	Requirement Type	Components	Summary	Priority	Rationale	Fit Criteria	Status
35	Non-Functional: Maintainability	Context	Hazardous Environmental Monitoring of Manufacturing Plant	1	Currently the environment of a plant is not monitored properly. However, this is quite important to guarantee the safety of an operator.	The safety of the operator is improved by 20% on the basis of environmental input information.	Open
49	Functional	Context	Access to energy-related information from production machines needs to be provided.	2	Energy-related information is measured by some of the operational machines (e.g. in the production plant), but it is not distributed into a network.	If any machine provides access to energy-related information, ebbits distributes this information to all interested parties.	Open
75	Functional	Context	System should aware of what which livestocks are in the building	3	Pigs in different phases have different requirements of climate, insulation, feed, vitamins, etc	System can adjust itself according to what's inside the building.	Incomplete
78	Functional	Context	System should provide location tracking of the stocks/livestocks	1	Users sometimes lost track where the goods /animals are.	Users can identified where the stocks / livestocks are	Implemented
81	Functional	Context	System should show Energy Cost for different granularity of production processes	1	Energy cost at different levels is needed to do benchmarking of operational processes.	Each automated process, machine is able to show energy cost	Incomplete
91	Functional	Context	Filter/fusion information for each operational process	3	Each process needs different resolution of information	Processes only get information needed	Incomplete
92	Functional	Context	Early maintenance notification when	4	Early maintenance prevent permanent damage to the robots,	Users/technicians are notified if robots need	Incomplete

			needed		ensure the reliability of robots	maintenance	
103	Functional	Context	Automatic calibration	1	Calibration is still done manually it is error prone, and takes time.	75% of existing manual calibration is done automatically.	Open
109	Functional	Context	Recognition of energy wasting behaviours	4	Help decision makers to optimize energy usage	Decision makers are alerted when energy wasting takes place	Open
155	Functional	Context	Synchronisation of Acquired Data is necessary	3	Data synchronization might be necessary, because data will be acquired automatically, manually, semi-manually with different timestamps.	A data synchronization component performs a timestamp-based synchronisation of a data set.	Open
450	Functional	Context	The system should compensate deviations of incoming data.	3	The incoming data could contain outliers e.g.: spikes which should not influence the measurement.	System provides configurable filter to exclude outliers e.g.: define upper and lower threshold	Open
454	Functional	Context	The system must monitor the state of devices and entities.	1	Continuous monitoring of entities is needed to detect anomalies (e.g.: ill Pigs, overheated welding gun)	The system should monitor and analyse sensor events	Open
455	Functional	Context	System needs to trigger business events based on changes of devices and entities states.	1	Enterprise applications have to be notified when the process starts and finishes, and further how much resources have been consumed for the process.	System generates business events defined in WP3.	Open

3.2.5 Devices

ID	Requirement Type	Components	Summary	Priority	Rationale	Fit Criteria	Status
140	Non-Functional: Operational	Devices	The middleware should monitor device's resource usage	3	The middleware should contain services that allow monitoring on what devices are doing.	Devices should provide an interface monitoring	Open

					This includes monitoring response time, device load (e.g., CPU), and message interchanges per second.	resource usage.	
141	Functional	Devices	Report errors in devices	2	Devices should be able to report errors.	Devices provide services for reporting errors.	Incomplete

3.2.6 Interface

ID	Requirement Type	Components	Summary	Priority	Rationale	Fit Criteria	Status
50	Non-Functional: Usability	Interface	Filtering to Obtain relevant Information	3	Too much information overwhelms farmers while making decisions.	Farmers are able to view the relevant information out of the whole.	Open
72	Functional	Interface	Officials have a back door access to highly important information	4	Officials want to avoid enterprises commit information / documents forgery	Officials have an access to certain information	Open
157	Functional	Interface	Different Views on the Data is necessary	3	We need services that provide different views on the data cloud by combining data from different sources.	Data can be filtered and sorted based on an arbitrary set of parameters.	Open
159	Non-Functional: Usability	Interface	End-users need to be able to management their distributed data	3	Farmers want to manage their distributed data, because today they have no full control of data.	End-users can easily manage data from distributed sources.	Open

3.2.7 Modelling

ID	Requirement Type	Components	Summary	Priority	Rationale	Fit Criteria	Status
27	Functional	Modelling	Product-related information should be represented in a machine-	3	Automatic processing requires that machines can understand and process information	Machines can process information of a product automatically.	Open

			readable format				
66	Functional	Modelling	Correlate problems found with production batches	2	When the source of problem have been isolated, producers must know which products/batches are affected.	Production batches affected by problems can be identified.	Open
131	Functional	Modelling	The system should support soft logic algorithms	4	The users want to know how trustful the data is.	The system supports at least two different soft-logic algorithms.	Open

3.2.8 Networking

ID	Requirement Type	Components	Summary	Priority	Rationale	Fit Criteria	Status
93	Functional	Networking	Bring data from fieldbus network to Ethernet network	3	Analytics is done by ERP program on a computer that work on TCP/IP.	Analytics software can analyse data from manufacturing robots	Open

3.2.9 Security

ID	Requirement Type	Components	Summary	Priority	Rationale	Fit Criteria	Status
82	Functional	Security	System should provide access restrictions to sensitive information.	1	Some sensitive information endanger company existence.	System provides access restrictions to sensitive information.	Open
135	Functional	Security	Dynamically loaded libraries must undergo a security check before their usage	3	Dynamically loaded libraries (e.g. DLL, JAR, OSGI bundle) could contain malicious code.	Dynamically loaded libraries must contain a valid signature in order to prevent security breaches in the system.	Open
458	Non-Functional: Security	Security	Libraries must only be accessible only for permitted applications.	4	Libraries could contain functionality that should not be available to all kinds of applications (e.g. calculation of	The dynamic loading of libraries must be restricted through policies.	Open

					quality rating of meat should only be allowed for slaughterhouse application but not for consumer application.)		

3.3 Requirements of WP6 – Mainstream Business Systems

3.3.1 Communication

ID	Requirement Type	Components	Summary	Priority	Rationale	Fit Criteria	Status
55	Functional	Communication	Reduce paper based communication	1	Paper based communication between enterprises takes time and efforts for inputting data to the system	50% Reduction of current paper based communication.	Open
80	Functional	Communication	ebbits should bridge communication between different applications in farms	3	Application between vendors are not able to communicate to each other, while data sometimes needs to be shared among applications.	At least applications from 3 different vendors are able to exchange information.	Open
161	Non-Functional: Operational	Communication	Electronic exchange of reports between enterprises	3	Report on the slaughtered pig is sent back ON PAPER to the farmer. This should be done electronically, in order for the farmer to combine it with the feeding information, e.g. identification of a father that produces; ill pigs.	Information is automatically put into a report format and sent to a recipient.	Open

3.3.2 Configurability

ID	Requirement Type	Components	Summary	Priority	Rationale	Fit Criteria	Status
106	Functional	Configurability	Energy benchmarking of different granularity such as machines, processes, plants	1	Management would like to know how effective the energy is used in different operational levels.	Management can do benchmarking in different operational level	Open

3.3.3 Context

ID	Requirement Type	Components	Summary	Priority	Rationale	Fit Criteria	Status
57	Functional	Context	Products rating by experts	1	Customer satisfaction is an important factor in business area.	Experts are able to give rating to products	Open
83	Functional	Context	Adjust production processes according to energy price policies	2	Reduce production cost by taking into account energy price policy from energy provider.	At least production speed and start/stop production can be adjusted according to the price of energy.	Open
107	Functional	Context	Support system for comparing different energy consumption among plants and corresponding processes	2	Management would like to learn from other plants if they use energy more efficiently.	Management can compare energy profile of plants.	Open
108	Functional	Context	Summary of energy related information at operational level for supporting management level optimizing energy use	3	Operational management needs a summary of energy related information that help them making decision to optimize the energy usage.	Management can access operational information.	Open
160	Functional	Context	The system should be self-ware	3	User statement: "Let me make my system self-aware"	User gets informed of any event she is interested in.	Open
167	Functional	Context	Save historical	3	Feeding history	Historical information	Open

			information in farms			is saved	

3.3.4 Devices

ID	Requirement Type	Components	Summary	Priority	Rationale	Fit Criteria	Status
459	Non-Functional: Operational	Devices	Mobile access to farm data in the ERP system	3	Mobility and access to mobile devices are required to manage aspects of the farm remotely if you are outside your office.	Mobile access (at least a read one) to the data of the ERP system is granted and solution should be browser-based.	Open

3.3.5 Modelling

ID	Requirement Type	Components	Summary	Priority	Rationale	Fit Criteria	Status
153	Functional	Modelling	Store meta-information with package labels	3	Temperature, location, humidity needs to be acquired and assigned with the packages. Dimension and weight is also required for the package, box, palette, etc.	A minimal set of 5 attribute-value pairs can be associated with any package/label.	Open

3.3.6 Security

ID	Requirement Type	Components	Summary	Priority	Rationale	Fit Criteria	Status
162	Functional	Security	Access-control of data sets	1	Access to data needs to be controlled, because some authorities require having access to this data, other stakeholders might have restricted access, other	Access rights can be defined for several stakeholder roles.	Open

					information could be made publicly available.		

3.4 Requirements of WP7 – Event Management and Service Orchestration

3.4.1 Architecture

ID	Requirement Type	Components	Summary	Priority	Rationale	Fit Criteria	Status
214	Functional	Architecture	Graceful degradation	2	The system should be functional even if some parts are unreachable or non functioning	System should be working even if 50 percent of sub systems are unreachable	Open
215	Functional	Architecture	Automatic Recovery from communication failures	2	We need to able to still guarantee delivery of data/events also in case of minor/temporary communication disruption.	The system should be able to recover from a 10 minute communication failure	Open
217	Functional	Architecture	It must be possible to order events in the actual event sequence	2	The delivery of events received from different sources might not follow in the original sequence at transmission. There might be communication delays etc that make them arrive in the wrong order. Nevertheless rules should be able to express temporal/sequence dependencies on events which reflect	It will be possible to express rules that contain temporal/sequence dependencies	Open

					the actual temporal event sequence at the sources.		
218	Functional	Architecture	An event history should be maintained	2	Rule definitions can refer to past events, and behaviour can be defined based on that event history.	A rule that refers to previous events can be expressed.	Open
220	Functional	Architecture	Event model based on common vocabulary	3	The ebbits platform must be able to handle a potentially large number of events on different levels of abstraction in the system architecture and with different semantics. The processing of events should also be related to the processing and management of data in ebbits, and this should be based on the use of common vocabularies.	The system can distinguish between several (minimum 2) event types based on a common ebbits vocabulary, represented in the event model.	Open
221	Functional	Architecture	Events mapped to (business) rules	3	Events and services are basic mechanisms for the implementation of the (business) rules logic in the ebbits architecture.	Events of a at least two different event types can be detected in a (business) rule, as expressed in the ebbits business rules framework (based on the ISA-95 standard).	Open
461	Functional	Architecture	Dependencies on past events possible	3	An action executed by the system may be dependent on more than one event, and some of them could have occurred in the past.	Actions executed by the system can be processed dependent on current events as well as events having occurred in the past.	Open

462	Non-Functional: Performance	Architecture	Scalable event processing	1	The platform must be able to handle a large number/high frequency of parallel event streams.	The platform can be configured to handle a span from low frequency processing (10 e/s) to high frequency processing (50000 e/s).	Open
463	Functional	Architecture	Semantic event processing	3	It must be possible to interpret events in the context of the different layers in the architecture (from PWAL to a business rules layer).	The system provides at minimum two layers of event processing where events can be captured and possibly filtered/fused	Open

3.4.2 Communication

ID	Requirement Type	Components	Summary	Priority	Rationale	Fit Criteria	Status
28	Functional	Communication	Heterogeneous enterprise systems need to be able to exchange information.	1	Enterprises use various information systems that need to exchange information. Information needs to be propagated throughout the chain.	At least three different enterprise systems can exchange information.	Open
68	Functional	Communication	Notification throughout the chain	2	A reduction of time for recalling a product from end consumers is needed.	At least 2 levels of the link in the chain can be notified automatically. e.g.: a farmer found a mutating pig problem, it can notify slaughterhouse and sausage factories who	Open

						bought meat from the slaughterhouse automatically.	
158	Functional	Communication	Alarms are send when specific situations occur	4	Issue alarms when an animal or production machine behaves differently or abnormal. However, context-aware behaviour is wished for experts NOT for the farmers.	Simple rules can be defined that trigger alarms.	Open

3.4.3 Configurability

ID	Requirement Type	Components	Summary	Priority	Rationale	Fit Criteria	Status
219	Functional	Configurability	Event history size and/or time span should be configurable	2	Different applications have different needs for the event history.	The event history can be configured wrt to timespan and size.	Open

3.4.4 Context

ID	Requirement Type	Components	Summary	Priority	Rationale	Fit Criteria	Status
39	Functional	Context	Retrieve manufacturing data history of any relevant event during production	1	If production defects are recognized, it is helpful to look at the production process history in order to find out what caused the defects.	Any manufacturing relevant (pressure, energy consumption, temperature, humidity, time etc) data is retrievable.	Open
216	Functional	Context	Explicit model of context	3	It must be possible to trace events and data items across processes and workflows, context	The system supports an explicit model of context, which can be applied in at least two application	Open

					management is one of the mechanisms to support this.	domains.	

3.4.5 Devices

ID	Requirement Type	Components	Summary	Priority	Rationale	Fit Criteria	Status
63	Functional	Devices	Diagnostic component to detect and correct malfunctions	3	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.	Malfunctions or strange behaviour of machinery are recognized early enough.	Open

3.4.6 Interface

ID	Requirement Type	Components	Summary	Priority	Rationale	Fit Criteria	Status
133	Functional	Interface	Different views on the device ontology	3	It should be possible to present a developer user with different perspectives on the device ontology, depending on that user functional need (e.g., a services perspective, device category perspective. etc.)	At least two different views are available in the ontology browser.	Open

3.4.7 Security

ID	Requirement Type	Components	Summary	Priority	Rationale	Fit Criteria	Status
156	Non-Functional: Security	Security	Meta-information associated with entities needs to be trustworthy	3	ebbits needs to guarantee that the information associated with a	Acquired information cannot be manipulated by unauthorized people.	Open

					specific product is the right and correct one that has not been manipulated.		

3.5 Requirements of WP8 – Physical World Sensors and Networks

3.5.1 Architecture

ID	Requirement Type	Components	Summary	Priority	Rationale	Fit Criteria	Status
23	Functional	Architecture	The ebbits platform should facilitate the integration of new physical devices into existing enterprise systems	1	Enterprises that already have a running ebbits system may need to add new devices.	ebbits provides a plug 'n' play framework for the integration of new devices into existing running systems.	Open
145	Constraint: Scope of the product	Architecture	Support of low-end devices	2	Middleware must support low-end devices like RFID tags. Therefore, it must be compatible with at least 32-bit devices with < 512 KB RAM/FLASH or less. For smaller devices, proxies can be used.	Middleware is able to be installed and run on low-end 32-bit devices with 512 KB RAM/FLASH in 90% of all cases. Proxies can be created to support more limited devices in 40% of all cases.	Open
470	Functional	Architecture	PWAL should support reconfigurable dynamic polling policies	3	Applications could have different polling needs, which eventually could change in runtime, so the PWAL must offer an easy reconfiguration of the polling policies per parameter and per application	The PWAL supports the dynamic polling policies of a PLC involved in a manufacturing plant and provide instruments for polling configuration.	Open

471	Functional	Architecture	ebbits should implement a distributed time dissemination and synchronization service	3	Several application in ebbits relay directly or indirectly on accurate timestamping of data and events, thus given the distributed nature of ebbits, a time dissemination and synchronization service is required within the platform.	ebbits provides a time dissemination and time synchronization service	Open
472	Functional	Architecture	PWAL should support accurate timestamping of data acquainted	3	The PWAL should be able to properly handle time information of the data and events it access/generate. This handling must include the synchronization to the ebbits time dissemination service and compensation of hardware and communication delays if possible.	The PWAL, while managing data or events, handles time-related information through the interaction with the time synchronization services.	Open
473	Functional	Architecture	PWAL should expose suitable methods in order to enrich raw data	3	Data acquainted through the PWAL needs to be enriched with meta-data (like source, geotag, timestamp, units, etc), which will be then used by upper layers and applications. Being the PWAL the lowest link between ebbits platform and the physical world, part of this meta-information could be	Meta-information could be attached to data or events at PWAL level	Open

					already attached at this level, easing the processing of it by the multi-sensor fusion and context awareness services.		
477	Functional	Architecture	PWAL should implement a heterogeneous multi-data aggregation in single events.	3	Event-driven data acquisition can easily generate scalability issues if single events are generated per sample. Thus aggregation of several samples in a single or few events has to be devised.	It is possible to aggregate events coming from the same source at PWAL level	Open
478	Functional	Architecture	PWAL should expose basic feature extraction and sensor fusion functionalities (e.g., moving average, decimation, filtering, etc) in order to minimize scalability issues	3	Some variables gathered through the PWAL could require high sampling frequencies and maybe just some feature of the acquired signal is of interest, so in order to save some bandwidth and avoid scalability issues, the PWAL could offer some basic feature extraction and sensor fusion capabilities.	The PWAL offers methods to extract features and to exploit the sensor fusion	Open
480	Functional	Architecture	Multiradio devices should avoid re-generation of HIDs when migrating to a different LinkSmart Network Manager	3	Devices with multiradio capabilities should be able to migrate to different networks without affecting the functionalities at the LinkSmart layer, including identificability, thus when registering to a	Devices with multiradio capabilities don't affect the LinkSmart layer while switching radio interface, thanks to a proper HIDs assignments' mechanism.	Open

					new Network Manager, the device should try to register itself with the previous HID, triggering an unregistration request to the previous Network Manager		
481	Functional	Architecture	Multiradio devices should be use local data caching and delay tolerance networking	3	Devices with multiple radio interfaces and in general devices with delay tolerance networking capabilities may experience periods with no networking, where some events may have happen, thus a local caching of its data may prevent any loss of data.	ebbits allow multiradio devices to use local data caching and delay tolerance networking.	Open
482	Functional	Architecture	Multiradio devices with local data caching should implement suitable application specific data-expiration policies in order to prevent cache overflows	3	Devices with local data caching for delay tolerance networking, may exhibit cache overflows if they generate big amounts of information and/or experience large offline periods, thus data-expiration policies need to be applied in order to prevent this.	The platform prevents the data loss of devices with multiple radio interfaces (e.g. in case of loss of network intervals)	Open
483	Functional	Architecture	Multiradio devices should be able to gather information about their network	3	Multiradio devices must be able to collect and expose some information about their	Multiradio devices can expose information about network interfaces	Open

			interfaces needed for the selection policies.		interfaces, like throughput, energy consumption, cost of traffic, quality of service, between others. Such information will be useful for defining interface selection policies.		
484	Functional	Architecture	Multiradio devices should select the most proper network interface according to the application requirements	3	Depending on the properties of the information (e.g., importance, quality of service, timeout, etc), multiradio devices should select the network interface most suitable to the requirements of the application accessing it, which could be a energy or cost saving policy for instance, or an urgent event that should be transmitted at all costs	Multiradio devices are able to select most proper network interface consistently with the application implemented	Open

3.5.2 Communication

ID	Requirement Type	Components	Summary	Priority	Rationale	Fit Criteria	Status
485	Functional	Communication	6LoWPAN networks should include frequency agility features in order to enhance	3	The ability to jump to a different channel automatically according to the channel occupancy	The ebbits platform includes frequency agility features in order to increase overall system	Open

			the overall system reliability		or interference seems a promising solution in order to cope with the high electromagnetic pollution present in manufacturing scenarios, thus a frequency agility service should be included in 6LoWPAN networks	reliability	

3.5.3 Configurability

ID	Requirement Type	Components	Summary	Priority	Rationale	Fit Criteria	Status
51	Non-Functional: Operational	Configurability	The network infrastructure needs to have self-configuration capabilities	3	Due to the huge amount of heterogeneous devices that can be connected to one network, this network needs to support the deployment through some sort of self-configuration.	A network of 20 devices can be deployed within one hour.	Open
53	Non-Functional: Operational	Configurability	New products should be networked with mainstream enterprise systems easily and cost-efficiently.	3	New products should be integrated into existing systems easily and cost-effectively, in order to support higher value-added, interoperable solutions.	A new product can be connected to an existing enterprise system within one day by one person.	Open
129	Functional	Configurability	Energy consumption should be optimized automatically	3	Farmers want to optimize the energy consumption in their production and they adapt this manually, and they would like	Rules can be defined to automatically optimize the energy Consumption	Open

					to have this done automatically (e.g. putting some production steps towards a cheaper time of the day).		
137	Non-Functional: Performance	Configurability	Systems built using LINKSMART should be scalable in terms of devices communicating	3	In large installations (e.g. in large factories) there will be many (embedded) devices in total. The middleware should support the development of such big systems.	The middleware supports applications in which more than 500 devices exist.	Open

3.5.4 Context

ID	Requirement Type	Components	Summary	Priority	Rationale	Fit Criteria	Status
41	Functional	Context	Integration of mobile sensing devices on running infrastructure manufacturing plant without interrupting running processes	2	In brown field it is too risky or too expensive to stop production in order to install missing sensors (e.g. smart meter).	It is possible to enhance a station/machine/robot with mobile sensing devices to gather data.	Open
125	Functional	Context	Associate meta-information to items	3	In parallel to the actual lifecycle (grow up of the animal, feeding, butchering, transportation, selling, consuming) there exists additional information such as the amount of food, medication an	Any item with an ebbits identifier can be associated with a set of meta-information.	Open

					animal has had, the energy for the production and transportation, that needs to be acquired and associated with the (bits and pieces of) animal.		
149	Functional	Context	Retrieve the behaviour on an individual animal level	3	Monitoring the drinking behaviour allows to recognize diseases 20 hours before. However, today it is not able to retrieve the drinking behaviour on an individual pig level, rather in a group. But with the emerging RFID identification it would be possible	The drinking behaviour can be retrieved on a individual pig level	Open

3.5.5 Devices

ID	Requirement Type	Components	Summary	Priority	Rationale	Fit Criteria	Status
34	Functional	Devices	Each sow carry an electronic unique ID	3	If battery cage is not used anymore and sows are let loose, farmers need to identify and track sows uniquely because each sow produces piglets with different quality and productivity.	Farmers can identify sows uniquely and track them if they are on the loose.	Open
46	Non-Functional: Operational	Devices	Affordable tagging/tracking system for pigs	2	If the tagging price is too high, the farmers are reluctant to use this tags	The price of a tag is less than 5% of the total profit a farmer can get from a pig.	Open

150	Non-Functional: Operational	Devices	Applied stand-alone devices should have a long battery life span	3	Data analysis is not done on the chip, because this consumes energy and the battery life-span comes to 3 months which is far too low.	Applied stand-alone devices have a minimal battery life span of one year.	Open
474	Functional	Devices	PWAL should be able to match PLC symbols with ebbits ontologies	3	The definition of symbols on an OPC server (i.e., variables of interest inside the PLC) could be made in accordance with the PLC programmer according to an agreed convention that could be exploited for an automatic matching with a device catalogue or ontology	The PWAL includes mechanisms to associate custom PLC symbols to ontologies selected within the ebbits environment.	Open
475	Functional	Devices	PWAL should adopt a lock and semaphore-based policy to the access of PLC memory	3	Since different applications could eventually be interested in a common variable, the PWAL must assure its access is controlled in order to avoid collisions in concurrent requests, as well as possible locks or restrictions to specific applications	When the PWAL tries to read or write some critical variable on-board the PLCs, all variables exposed are safe	Open
476	Functional	Devices	PWAL should implement an error control strategy to assert correct data type and values	3	Errors in writing variables to the PLC must be avoided at all cost, since they can lead to a halt in the running	The PWAL always writes on the PLC memory the appropriate data types, consistently with variables and	Open

			written to the PLC		program. The PWAL has to adopt a suitable error control strategy in order to assert data has been introduced correctly (this eventually would require a control logic in the PLC program as well).	respecting the appropriate value ranges.	
479	Functional	Devices	Multiradio devices should be able to detect which LinkSmart Network Manager to connect/migrate to, according to the current network interface active	3	Devices with multiradio capabilities should be able to switch interface, and therefore network, without compromising the connectivity to the LinkSmart layer, this means that when migrating to a new interface, the device should register itself to the closest Network Manager available in that network	Devices accessing ebbits by using non-corporate or external networks (e.g. 3GPP) can detect which border network manager they must connect to.	Open

3.5.6 Interfaces

ID	Requirement Type	Components	Summary	Priority	Rationale	Fit Criteria	Status
62	Functional	Interface	Seamless data collection	1	Data collection is the required input for simple and complex analysis in both manufacturing and traceability scenario.	Both mobile and static sensors are affixed to any medium (animal, robot etc) in order to sense the environment.	Open
77	Functional	Interface	ebbits should support legacy	1	Many legacy systems still use old	At least 3 types of common old	Open

			network interfaces		network interfaces	interfaces of each domain (manufacturing and farm) are supported	
84	Functional	Interface	Interfacing with Programmable Logic Controller of production robots	3	Production automation is controlled through PLC	Software and hardware interfaces to PLC is defined	Open
130	Functional	Interface	ID Management: Item identification system should provide open interfaces to other systems	3	Identification of pigs is done with RFID tags at their ears and with antennas in corridors that recognize pigs passing by. The system should not be connected to a specific system, but rather provide open interfaces that can be exploited by any system.	Any system can easily access the item identification system.	Open

3.5.7 Middleware Layer

ID	Requirement Type	Components	Summary	Priority	Rationale	Fit Criteria	Status
469	Functional	Middleware Layer	Local LinkSmart instances should properly handle local variables when remote environments are restarted	3	LinkSmart should be more robust to different network issues, like temporary offline periods or restarting of remote instances	The critical event of LinkSmart environment reboot while developing is handled by a specific exception	Open

3.5.8 Modelling

ID	Requirement Type	Components	Summary	Priority	Rationale	Fit Criteria	Status
127	Functional	Modelling	Batches need to be identified on a	2	The average feed production batch	Every batch has one unique identifier.	Open

			farm level		size is 20 tons; however farmers want their own specific production which is way smaller. A unique identification of the batch to the farm is necessary.		

3.5.9 Networking

ID	Requirement Type	Components	Summary	Priority	Rationale	Fit Criteria	Status
22	Non-Functional: Operational	Networking	The ebbits should be able to handle massive number of devices	2	The future use cases of ebbits need to handle massive number of devices and applications within and cross enterprises, i.e. ci. 300-1000 in a manufacturing plant and 500 in a farm.	ebbits is able to handle 500 devices simultaneously.	Open
37	Non-Functional: Performance	Networking	Higher bandwidth and range of wireless connection	3	Currently, in manufacturing plants BT is used for wireless communication. It supports ranges <50m. Especially, the bandwidth is too low: max 24 MBit/s.	Communication range and in particular bandwidth are considerably higher.	Open
86	Functional	Networking	Reliable wireless solution for new sensors	3	Cable costs are high and due to harsh condition, cable might break	Sensors are using wireless connection.	Open
143	Non-Functional: Legal	Networking	Comply with industrial standards	2	The middleware should embrace existing industrial device integration and communication	Claimed support for any specific standard in ebbits can be verified using the conformance rules /	Open

					standards, e.g. EIB/KNX	procedures available from the issuing standards body.	

3.5.10 SDK

ID	Requirement Type	Components	Summary	Priority	Rationale	Fit Criteria	Status
468	Functional	SDK	ebbits platform should support automatic builds	3	To make the ebbits platform easier to use for future developers it has to be easy to build and start. Else developers will not take it serious and will not try it.	All demos and prototypes have to provide a uniform method for building and starting. The build process has to be decided in advance else it will take extra effort to realize it.	Open

4. Impacts on the individual work packages

This section summarizes the impact of new and updated requirements on the future work conducted in the technical WPs.

4.1 Impact on WP4

Within in the first iteration of the ebbits project, five LLs have been reported, validated and analysed. This yielded 40 new requirements of which 36 have resulted from deliverable Coverage and scope definition of a semantic knowledge model (D4.3). It furthermore yielded update of one requirement.

4.1.1 Context

We learned that monitored and sensed data [ebbits-380], and alerts [ebbits-382] should be modelled with a context. Both requirements are findings from deliverable D4.3 and have been discussed there.

4.1.2 Modelling

We made the requirement for a common ontology namespace explicit [ebbits-465], agreed to use OWL-Lite as Knowledge Representation [ebbits-464], and agreed to model only items that are relevant to ebbits in order to limit the ontologies [ebbits-467]. As a result from D4.3 we created several requirements how to model defined items and what properties, e.g. status, quality, energy consumption, etc. they should have. It should be noted that meanwhile we achieved progress on the ebbits vocabulary in WP3, especially in the submitted Vertical and horizontal business vocabularies (D3.2) so that it is currently under discussion to remove the detailed requirements again in order to reduce redundancy. However, currently our set of requirements contains definitions how to model:

- Animal [ebbits-387],
- Device [ebbits-383],
- Feedstuff [ebbits-386],
- Generic Information [ebbits-396],
- Halves of slaughtered pigs [ebbits-392],
- Invoices [ebbits-394],
- Logistic [ebbits-385],
- Meat package [ebbits-388],
- Monitored data [ebbits-381].
- Order [ebbits-393],
- Payment reports [ebbits-395],
- Production [ebbits-384],
- Soil [ebbits-389], and
- Sow farm production [ebbits-391],

4.1.3 Architecture

Several requirements for the architecture resulted from the analysed use cases in D4.3. They are dealing with several parts of Semantic Stores, though more general parts of the ebbits platform as well.

4.1.4 Security

Several requirements have been added in the component Security. These requirements also resulted from D4.3. From the user stories it became clear that not every farmer, consumer, retailer, etc. is allowed to access all information. That has a direct influence on accessing Semantic Stores. Members of the consortium bring expertise and publications already on accessing parts of ontologies with respect to an ordered criterion, such as level of detail, level of expertise, user role, etc. This state of the art might be a candidate to solve the problems induced by the requirements in the security component.

4.1.5 Interface

The interface component contains 4 added requirements. Information exchange about sent feedstuff [ebbits-405], as well as orders, invoices [ebbits-415] and payments [ebbits-416] requires defined interfaces as it has been discussed in deliverable D4.3. In order to query ontologies conveniently SPARQL is used as query language, or an abstraction is used if this turns out to be more usable. This has been collected in [ebbits-466].

4.2 Impact on WP5

Within in the first iteration of ebbits project 11 LLs (LLs) have been reported. The analysis of the LLs has resulted into 14 new requirements and one updated requirement.

4.2.1 Architecture

We learned that a particular component is needed in order to monitor contextual changes of devices (e.g. usage, as stated in [ebbits-140]) and processes ([ebbits-454]), and based on the information of contextual changes trigger business events (as stated in [ebbits-455]). Therefore, WP5 has developed a Context manager that deals with the context monitoring. However, we also learned if the amount of events increases, a centralized component does not scale well. Hence, a distribution of the context management has to be investigated including a synchronization strategy of context information among distributed context managers ([ebbits-456]).

Furthermore, we learned that sensors couldn't always deliver accurate data due to environmental noises. As, [ebbits-450] states that *the system should compensate deviations of incoming data*, WP5 has to look into soft-logic algorithms to calculate the confidence of a data ([ebbits-131]) so that the system is able to know when it has to compensate the unsure data.[GAl1] This implicates that WP5 investigates more use cases where soft logic algorithms have been applied successfully and develops a respective sensor fusion plug-in.

Another lesson learned that we have obtained is, that the ebbits projects involves many types of users. For instance, end-users, such as operators, must be able to configure the behaviour of their particular system, such as operators. For this target group of users WP5 has to define simple rules to configure the system ([ebbits-139]), as they are not familiar dealing with complex programming languages. This implicates that WP5 will develop a rule engine, which allows users define new context information that needs to be monitored by the context manager during run-time.

The result of the lesson learned indicates that in WP5 a sensor fusion component needs to be developed as an independent module ([ebbits-452]) from the context manager to allow reusing sensor fusion algorithms. Therefore the communication between context manager and sensor fusion manager needs to be defined and investigated. An approach that LinkSmart proposes is to use SOAP based Web Services as a stateless communication medium among managers. However, this approach is not suitable for all cases where communication delay is an issue. Therefore, other communication approaches, e.g. REST Web Services, caching have to be investigated.

Further, we learned that new devices can be added during operational or maintenance processes. Adding new devices might require new sensor fusion modules to be integrated during run-time [ebbits-451]. This implicates that WP5 must utilize a plug-in system that allows these algorithms (e.g. mathematical calculations) to be dynamically added and loaded during run-time. Moreover, dealing with different versions of plug-ins is another issue WP5 needs to take into account.

4.2.2 Interface

For the Sensor Fusion Manager the definition of a service interface needs to be agreed on.

Dynamic loading of libraries usually must be done through established software design patterns. For this WP5 will investigate how different design patterns suit this need, such as the factory pattern, plug-in pattern, or dependency injection pattern.

Since shutting down a manufacturing plant costs a large amount of losses and secondly the high mobility that the traceability scenario involves, the system should support being extended at run-time. This also includes the deployment of new devices. In order to be integrated into the running system the context model must allow for being extensible during run-time ([ebbits-134]), and thus the ontology manager has to provide an interface facilitating to insert new contextual concepts and allow assigning this concept with a particular physical device ([ebbits-453]).

4.2.3 Security

Since libraries need to be loaded during run-time, they need to contain a valid signature in order to prevent security breaches in the system. It implicates that WP5 should investigate techniques to sign the libraries to be loaded at run-time and checking their validity ([ebbits-135]).

On the other hand, sensor fusion algorithms could be used to calculate information that is restricted to certain applications since it might contain sensitive information, and thus it is necessary to restrict the access to specific plug-ins ([ebbits-458]). This implicates that policy enforcement techniques for accessing software plug-ins need to be investigated and implemented.

4.2.4 Storage

To infer context requires historical information. Therefore historical events may require to be stored by the LinkSmart Event Manager. Further, a query interface will be provided in order to query events based on specific criteria. This implicates that WP5 and WP7 should agree on a query language that is used for the retrieval of historical information.

4.3 Impact on WP6

Within in the first iteration of the ebbits project, two LLs have been reported, validated and analysed. This yielded one updated requirements, and neither new and nor any deleted requirements.

4.3.1 Devices

We learned that mobile access to data in the management systems (for example ERP systems) would be desirable especially in the Food Traceability scenario, as farmers do often work outside their offices. Hence, the ebbits project should support mobile devices (dedicated for farmers) to at least enable reading access to management systems from everywhere, i.e. any possible location allowing remote connection. As, there exists a variety of mobile operating systems (Google Android, Symbian OS, or Apple's iOS) and the farmer should not be restricted to only use special ones, a browser-based access on the device would allow for the use of different mobile platforms ([ebbits-459]).

4.3.2 Interface

We have learned that the context manager that monitors and analyses sensor events will generate business events based on changes of devices and entities states. Such events may include the start and the end of processes and the consumed resources and later will be fed into the enterprise applications ([ebbits-455]). Therefore, a corresponding interface between the context manager and the enterprise systems is necessary.

4.3.3 Context

We learned that monitored and sensed data ([ebbits-380]) and alerts ([ebbits-382]) should be modelled with a context, i.e., the data should be annotated with concepts from ontology. Furthermore, such enrichment of raw data with meta-data can already happen at a level just above the physical world ([ebbits-473]). That means that on the highest level, i.e. the enterprise systems level, a provision for handling and storing of such already annotated meta-data is necessary.

4.3.4 Modelling

The necessary meta-data for annotations has now been specified in much more detail for various items and concepts such as:

- Animal ([ebbits-387]),
- Device ([ebbits-383]),
- Feedstuff ([ebbits-386]),
- Generic Information ([ebbits-396]),
- Halves of slaughtered pigs ([ebbits-392]),
- Invoices ([ebbits-394]),
- Logistic ([ebbits-385]),
- Meat package ([ebbits-388]),
- Order ([ebbits-393]),
- Payment reports ([ebbits-395]),
- Production ([ebbits-384]).
- Soil ([ebbits-389]), and
- Sow farm production ([ebbits-391]),

4.4 Impact on WP7

Within in the first iteration of ebbits project seven LLs have been reported for WP7. The analysis of the LLs has resulted into six new requirements and one updated requirement.

4.4.1 Architecture

Scalability of event architecture is important, filtering and aggregation is needed ([ebbits-462]). In order for the ebbits system to interpret the aggregated/filtered events there needs to be semantic descriptions available for the aggregated/filtered events, not just the individual events ([ebbits-220], [ebbits-463]).

4.4.2 Modelling

It should be possible to interpret events based on semantic descriptions in order to relate different event types and processes ([ebbits-220], [ebbits-463]). Hence, there is a need for an event ontology being based on a vocabulary related to the overall ebbits data and process models ([ebbits-220], [ebbits-463]).

4.4.3 Storage

In order for the rules to express historic event dependencies we need to store an event history ([ebbits-218], [ebiits-461]). The event history will also allow for event ordering ([ebbits-217], [ebbits-461]).

4.5 Impact on WP8

Within in the first iteration of ebbits project, concerning WP8, 17 LLs have been reported, validated and analysed. The analysis has resulted in the following changes to the initial set of requirements (see D2.9.1). The analysis of the LLs in WP8 has led to the definition of 18 new requirements, 3 existing requirements have been updated.

4.5.1 Development

We learned that, when the remote LinkSmart environment is restarted while developing, it is necessary to throw a specific exception. This shall allow prevent or handle the loss of variables in the local environment. Other unexpected events generate further explicit exceptions, in order to enhance code safety ([ebbits-469]).

4.5.2 Architecture

Applications access physical-world parameters (like PLC's data) with a dynamically changing polling interval. In order to achieve this, applications need to be aware of some characteristics in the physical devices for instance, the periodicity of detecting and exposing information data by a physical sensor or the duration of a measurement. As a consequence, it is necessary to include in the architectural model the logic that enables an easy reconfiguration of PWAL's polling policies during runtime (as explained in [ebbits-470]). The polling interval is only one of the possible device configurations. Another one is the communication handling in devices with multiradio capabilities. Some mechanisms (like cryptographic tickets) have to be implemented in order to avoid the re-generation of new HIDs in multiradio devices when connecting to new network managers ([ebbits-479]). Devices with multiradio capabilities should be able to migrate to different networks without affecting the functionalities at the LinkSmart middleware layer. As consequence, while registering to a new Network Manager, the device try to register itself with the previous HID, triggering a de-registration request to the previous Network Manager ([ebbits-480]).

A huge amount of heterogeneous devices can be connected to the same network. Requirement [ebbits-51] explains that some self-configuration features are required, in order to allow automatic connections and self-configuration.

Concerning PLC devices the requirement [ebbits-474] states issues related with the naming of PLC symbols, which is commonly fixed by PLC developers. However, it can be associated to the ebbits device catalogue or other ontologies. Moreover, since multiple parameters may change at the same time on-board a PLC (e.g. due to some synchronous polling of wired sensors), a potentially large number of events can be generated at the same time, and taking into account very large-scale deployments like the ones devised inside the ebbits manufacturing scenario, these events can originate scalability problems. This raises the need to support both aggregation and disaggregation of events to enhance scalability without compromising the application performance ([ebbits-477]).

Finally, some variables gathered through the PWAL require high sampling frequencies, but applications are just interested in some features/data instead of the whole signal. Therefore, in order to save bandwidth and to avoid scalability issues, the PWAL offers, by employing a local data fusion manager, some early feature extraction and sensor fusion capabilities ([ebbits-478]).

Data acquainted through the PWAL needs to be enriched with meta-data (like source, geo tag, timestamp, units, etc.), which will be then used by upper layers and applications. Being the PWAL the lowest link between ebbits platform and the physical world, part of this meta-information can be already attached at this level. For instance, by applying some multi-sensor fusion and context awareness rules at this stage ([ebbits-473]). Another added value that ebbits shall introduce is the monitoring and log of several metrics in devices, tools, or machinery and resources in general. By using suitable symbols at PWAL level, such information can be identified and correlated with the corresponding industrial processes. In order to achieve this, devices have to be annotated with id, type, name, location and current/historical data ([ebbits-383]).

One of the most critical functionalities both the PWAL and the overall ebbits platform will support is the accurate data timestamping. Several ebbits applications rely directly or indirectly on such timestamping of data and events, thus given the distributed nature of ebbits, this requirement imposes the need of a time dissemination and synchronization service within the platform. Furthermore, the PWAL should synchronize to the ebbits time dissemination service, and when timestamping, compensate any hardware and communication delays if possible, in this way, the ebbits platform provides a common time scale ([ebbits-471] and [ebbits-472]).

4.5.3 Interface

Multiradio devices must be able to collect and expose some data through their interfaces. For instance throughput, energy consumption, cost of traffic, or quality of service between others. Such information will be useful for defining interface selection policies ([ebbits-483]). This information may be shared inside the meta-data, propagated through additional events or exposed as services. Furthermore, depending on the properties of the information (e.g., level of importance, quality of service, time to live, etc.), multiradio devices select the network interface most suitable to the requirements of the application accessing it. For instance, this could be energy or cost saving policy, or an urgent event to be transmitted immediately at any cost ([ebbits-484]).

New devices should be integrated into existing systems easily and cost-effectively, in order to support higher value-added, interoperable solutions. Flexible integration and deployment of such devices may require the use of different LinkSmart network managers depending on the available network interface and on service being implemented ([ebbits-53]).

The requirement [ebbits-485] deals with the following problem: the presence of interference in a specific environment might significantly degrade the performance of a 6LoWPAN (which comes from the real-world problem of high electromagnetic pollution present in manufacturing scenarios). Thus 6LoWPAN sensor networks integrated within the ebbits platform must include a set of frequency agility features in order to increase the overall system reliability.

4.5.4 Security

The early use of PLCs in the first year of the project, has given the possibility to discover another critical situation to be managed: when the PWAL tries to read or write some critical variable from the PLCs, all variables exposed shall be safe. Therefore a semaphore-based policy will be implemented, in order to ensure that all critical variables, while being read or written, do not return weird values due to asynchronous unmanaged actions ([ebbits-475]). Furthermore, the PLC's memory writing processes may trigger sharp program interruption, if the data receives is incorrect or inadequate. The PWAL has therefore to adopt a suitable error control strategy in order to assert data that has been introduced correctly; in the PLC case, eventually this might require control logic in the PLC program as well ([ebbits-476]).

4.5.5 Storage

Devices with multiple radio interfaces and, generally speaking, devices with delay tolerance networking capabilities may experience periods with no networking, in which some events may have happen, thus a local caching of this data may prevent any loss of important data. This issue has been translated into the requirement [ebbits-481], which imposes that multiradio devices shall support local data caching and delay tolerant networking. Furthermore, devices with local data caching for delay tolerance networking, may exhibit cache overflows if they generate big amounts of information and/or experience large offline periods, thus data-expiration policies need to be applied in order to prevent this. The platform shall prevent the data loss of devices with multiple radio interfaces (e.g. in case of loss of network intervals) with adequate buffering instruments and policies, maintaining the data storage as low as possible ([ebbits-482]).

5. Conclusion

Following up on the work reported in D2.4 and evoked by the LLs reported in D2.7.1 and respective change requests reported in D2.8.1 more new requirements have been defined and further some existing one's have been updated. As just the first year of the ebbits project is completed and the focus was rather on eliciting requirements and clustering those, most of the requirements do remain in an open status. Hence, viewing the development of the ebbits project from a requirements engineering perspective the first iteration of the ebbits project was successful, but to able to build on this for the next iteration the main goal should be to fulfill continuously the list of requirements yielding to a total of more implemented requirements.

6. Open Requirements

This chapter outlines the complete list of open requirements after the first iteration of the ebbits project.

ID	Requirement Type	Component/s	Summary	Priority	Rationale	Fit Criteria
21	Functional	Interface	Seamless Access to Energy Related Information	3	Energy-related information is only available right at the Human-Machine Interface at the respective station.	At least three different types of devices within the enterprise (manufacturing plant, management, administration) that feature a user interface can display energy-related information.
22	Non-Functional: Operational	Networking	The ebbits should be able to handle massive number of devices	2	The future use cases of ebbits need to handle massive number of devices and applications within and cross enterprises, i.e. ci. 300-1000 in a manufacturing plant and 500 in a farm.	ebbits is able to handle 500 devices simultaneously.
23	Functional	Architecture	The ebbits platform should facilitate the integration of new physical devices into existing enterprise systems	1	Enterprises that already have a running ebbits system may need to add new devices.	ebbits provides a plug 'n' play framework for the integration of new devices into existing running systems.
24	Functional	Communication	A product's lifecycle history can be traced within less than 24 hours.	1	EU Regulation in the future will obligate enterprises to be able to produce a request of information within 24 hours.	A product lifecycle cross Enterprises can be traced within less than 24 hours.
25	Functional	Security	Authentication to product-related tracing information should be provided on a product level	3	Enterprises want to share valuable product-related information, if there is a business value for them, and they	Only authorized stakeholders can see the product's information.

					want to be able to grant access to this information on a product level.	
27	Functional	Modelling	Product-related information should be represented in a machine-readable format	3	Automatic processing requires that machines can understand and process information	Machines can process information of a product automatically.
28	Functional	Communication	Heterogeneous enterprise systems need to be able to exchange information.	1	Enterprises use various information systems that need to exchange information. Information needs to be propagated throughout the chain.	At least three different enterprise systems can exchange information.
29	Functional	Context	Stakeholders should be able to access product-related information instantly.	2	Farms want to able to sell their “brand” to the costumer, which creates a good business case to motivate farmers to participate in sharing information. A certificate of origin would be good for the customers. There exists also a safety issue: when the end-user is allergic to something, we can provide information about the contents of the beef.	For each meat product, stakeholders can access the name of the producing farm, the millage, Feed stuff, medicine and the CO2 footprint. stakeholders e.g.: meat quality control of slaughterhouse and retail store, end consumers
34	Functional	Devices	Each sow carry an electronic unique ID	3	If battery cage is not used anymore and sows are let loose, farmers need to identify and track sows uniquely because each sow produces piglets with different quality and productivity.	Farmers can identify sows uniquely and track them if they are on the loose.

35	Non-Functional: Maintainability	Context	Hazardous Environmental Monitoring of Manufacturing Plant	1	Currently the environment of a plant is not monitored properly. However, this is quite important to guarantee the safety of an operator.	The safety of the operator is improved by 20% on the basis of environmental input information.
37	Non-Functional: Performance	Networking	Higher bandwidth and range of wireless connection	3	Currently, in manufactruring plants BT is used for wireless communication. It supports ranges <50m. Especially, the bandwidth is too low: max 24 MBit/s.	Communication range and in particular bandwidth are considerably higher.
39	Functional	Context	Retrieve manufacturing data history of any relevant event during production	1	If production defects are recognized, it is helpful to look at the production process history in order to find out what caused the defects.	Any manufacturing relevant (pressure, energy consumption, temperature, humidity, time etc) data is retrievable.
40	Non-Functional: Maintainability	Context	Life-cycle of a robot and its components is traceable	2	At the moment the lifecycle of a robot's component is not predictable. However, being able to predict its life-cycle could support to avoid deviations during production.	Based on analyzing data of real field tests the lifecycle can be predicted properly.
41	Functional	Context	integration of mobile sensing devices on running infrastructure manufacturing plant without interrupting running processes	2	In brown field it is too risky or too expensive to stop production in order to install missing sensors (e.g. smart meter).	It is possible to enhance a station/machine/robot with mobile sensing devices to gather data.
42	Functional	Modelling	semantic relationships between data	1	Currently, any data is stored in a simple database. Hence, data is available, but cannot	Data can be queried and inferred in order.

					be interrelated intelligently.	
44	Functional	unassigned	Farmers are able to retrieve optimized models from research	4	Farmers are willing to share data if they could get something in return such as models to optimize feeding process.	Farmers can get optimized models electronically.
45	Functional	unassigned	System can feed the farms data to research	4	Most of the farming models are developed by research organizations, universities etc.	Researchers are able to get their hands on life data on farms.
46	Non-Functional: Operational	Devices	Affordable tagging/tracking system for pigs	2	If the tagging price is too high, the farmers are reluctant to use this tags	The price of a tag is less than 5% of the total profit a farmer can get from a pig.
47	Non-Functional: Operational	Configurability	Resilience and adaptable to environment condition changes	2	Environmental changes such as lighting, temperature affect the results of manufacturing process. so far machines are tuned manually by technicians. adapting to environmental condition can lead to reducing energy consumption e.g.: reduce heater temperature when it's warm outside.	Machines can adapt its parameters adapting to environmental changes.
49	Functional	Context	Access to energy-related information from production machines needs to be provided.	2	Energy-related information is measured by some of the operational machines (e.g. in the production plant), but it is not distributed into a network.	If any machine provides access to energy-related information, ebbits distributes this information to all interested parties.

50	Non-Functional: Usability	Interface	Filtering to Obtain relevant Information	3	Too much information overwhelms farmers while making decisions.	Farmers are able to view the relevant information out of the whole.
51	Non-Functional: Operational	Configurability	The network infrastructure needs to have self-configuration capabilities	3	Due to the huge amount of heterogeneous devices that can be connected to one network, this network needs to support the deployment through some sort of self-configuration.	A network of 20 devices can be deployed within one hour.
52	Functional	Communication	Interoperability needs to be created between various subsystems in the manufacturing area	2	The subsystems in manufacturing environments are currently not interconnected and not able to exchange information.	Three independent subsystems from the manufacturing area can exchange information (show a use case)
53	Non-Functional: Operational	Configurability	New products should be networked with mainstream enterprise systems easily and cost-efficiently.	3	New products should be integrated into existing systems easily and cost-effectively, in order to support higher value-added, interoperable solutions.	A new product can be connected to an existing enterprise system within one day by one person.
54	Functional	Context	Slaughter house needs to know how many pigs they will get from farmers	3	Slaughter house needs to know the flow of the pigs coming to them so that they can plan and balance supply from farmers and demands from retail store.	Slaughter house can forecast how many pigs are going to be supplied by farmers.
55	Functional	Communication	Reduce paper based communication	1	Paper based communication between enterprises takes time and efforts for inputting data to the system	50% Reduction of current paper based communication.
56	Functional	Modelling	Farmers need to save	2	Breeders combinations	Farmers can at least

			and able to reflect breedings history information		produce different quality of piglets	trace breeders combinations that produce unhealthy piglets.
57	Functional	Context	Products rating by experts	1	Customer satisfaction is an important factor in business area.	Experts are able to give rating to products
59	Functional	Communication	Back tracing production problem from complaints	1	The source of the problem during production need to be localized and used for repairing recalled products.	50% of time reduction to localize the source of the problem. (no determinant time is possible because it depends on how complex the production processes are)
60	Non-Functional: Operational	Devices	Improve air compression energy usage	3	Air compression is one main energy guzzler. Only 40% of air can be transferred effectively.	More than 40% of air can be transferred effectively
61	Non-Functional: Usability	Interface	Display plant activities in real-time	3	To observe the complexity of a production inside the plant.	A user-friendly interface is provided to the relevant stakeholders to view activities inside the plant.
62	Functional	Interface	Seamless data collection	1	Data collection is the required input for simple and complex analysis in both manufacturing and traceability scenario.	Either mobile or static sensors are affixed to any medium (animal, robot etc) in order to sense the environment.
63	Functional	Devices	Diagnostic component to detect and correct malfunctions	3	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.	Malfunctions or strange behaviour of machinery are recognized early enough.
64	Functional	Context	Historical data should be recorded persistently.	1	Quality is very important inside an assembly line as it is the essential parameter used for force tests or lack tests.	Quality related information is logged inside a proper carrier medium.

					Furthermore, if failures are detected lately when a car is already in the market, but shows some lack, the production history can be traced to find the devil in the detail.	
65	Functional	Communication	Producers can push notification of recalled products to costumers	2	Producers want to avoid getting sued because they weren't fast enough notifying consumers about recalled products. the common methods is through TV, Radio, Website, for cars can be through phones	Customers who bought the products are notified within 24 hours since products being recalled
66	Functional	Modelling	Correlate problems found with production batches	2	When the source of problem has been isolated, producers must know which products/batches are affected.	Production batches affected by problems can be identified.
67	Functional	Communication	Automatic analysis of cross enterprises product life cycle data	3	Searching production problem from end costumer complaints need to track back data from several enterprises and logistic.	Analyzing data cross enterprises can be done online and automatically.
68	Functional	Communication	Notification throughout the chain	2	A reduction of time for recalling a product from end consumers is needed.	At least 2 levels of the link in the chain can be notified automatically. e.g.: a farmer found a mutating pig problem, it can notify slaughterhouse and sausage factories who bought meat from the slaughterhouse automatically.

69	Functional	Context	Supplier can predict when to make the next delivery to a consumer	2	Suppliers can make an early offer when the inventory of their costumers almost depletes	Supplier can make an estimation when the inventory of their costumer are almost empty.
70	Functional	Context	Predict if his suppliers will not be able to fulfil their demands (quantity)	3	Consumers are able to find other suppliers to fulfil their demands	Consumers are able to make estimation if the a supplier will not be able to supply his demands.
71	Functional	Modelling	Definition of smallest unit can be traced or uniquely identified	2	Small parts of products / cheap parts of products cannot be identified anymore because of physical limitation of tags. Price of tags is too expensive to tag cheap parts.	Clear definition in both domains of what can be tagged.
72	Functional	Interface	Officials have a back door access to highly important information	4	Officials want to avoid enterprises commit information / documents forgery	Officials have an access to certain information
73	Functional	Communication	Transferring IDs cross enterprises	1	When IDs have no standard (such as EAN), IDs are generated from each ERP. thus each enterprise has their own IDs.	IDs from cross enterprises can be linked automatically.
76	Functional	Context	Feeding systems should aware of the animals weight	3	Farmers must keep poultry products on certain weights	Feeding system can control the portion of feed based on the weight
77	Functional	Interface	ebbits should support legacy network interfaces	1	Many legacy systems still use old network interfaces	At least 3 types of common old interfaces of each domain (manufacturing and farm) are supported
80	Functional	Communication	ebbits should bridge communication between different applications in farms	3	Application between vendors are not able to communicate to each other, while data	At least applications from 3 different vendors are able to exchange information.

					sometimes needs to be shared among applications.	
82	Functional	Security	System should provide access restrictions to sensitive information.	1	Some sensitive information endanger company existence.	System provides access restrictions to sensitive information.
83	Functional	Context	Adjust production processes according to energy price policies	2	Reduce production cost by taking into account energy price policy from energy provider.	At least production speed and start/stop production can be adjusted according to the price of energy.
84	Functional	Interface	Interfacing with Programmable Logic Controller of production robots	3	Production automation is controlled through PLC	Software and hardware interfaces to PLC is defined
85	Functional	Devices	3 Measurement Points for every station in body welding	3	Energy cost that can be calculated includes: lightings, processes, energy for welding	3 measurement points are covered
86	Functional	Networking	Reliable wireless solution for new sensors	3	Cable costs are high and due to harsh condition, cable might break	Sensors are using wireless connection.
87	Non-Functional: Operational	Devices	Reliability of the system should be more than 30 years	2	A production plant of trucks can runs for 30 years	99% of the time system is able to run 30 years.
88	Non-Functional: Operational	Devices	Hardware components are able to handle harsh condition	3	Harsh condition damage electronic devices	Devices fulfil manufacturing insulation standard for cables and sensors.
89	Non-Functional: Maintainability	Architecture	Scalable solution (scale up and scale down)	3	Adjustment to desired number of production, require to add or reduce machines	Configuration of scaling up / down a plant can be achieved in max 8 hours.
90	Non-Functional: Operational	Interface	Central point to start the whole plant	3	Machines have to be started in the right order.	The whole machines in a plant can be started from a central point.
93	Functional	Networking	Bring data from fieldbus network to Ethernet	3	Analytics is done by ERP program on a	Analytics software can analyse data from

			network		computer that works on TCP/IP.	manufacturing robots
103	Functional	Context	Automatic calibration	1	Calibration is still done manually it is error prone, and takes time.	75% of existing manual calibration is done automatically.
104	Functional	Devices	Automatic start up synchronization among machines	2	Starting up machines in a plant is complicated such as the order of machines, min temperature etc.	A plant can be "re-started" automatically in less than an hour.
105	Non-Functional: Operational	Devices	Reduce water consumption in PWT	4	Water consumption for cooling and lubricating purposes in PWT is really high (300-500 lt./minute)	10% of water consumption can be reduced
106	Functional	Configurability	Energy benchmarking of different granularity such as machines, processes, plants	1	Management would like to know how effective the energy is used in different operational levels.	Management can do benchmarking in different operational level
107	Functional	Context	Support system for comparing different energy consumption among plants and corresponding processes	2	Management would like to learn from other plants if they use energy more efficiently.	Management can compare energy profile of plants.
108	Functional	Context	Summary of energy related information at operational level for supporting management level optimizing energy use	3	Operational management needs a summary of energy related information that help them making decision to optimize the energy usage.	Management can access operational information.
109	Functional	Context	Recognition of energy wasting behaviours	4	Help decision makers to optimize energy usage	Decision makers are alerted when energy wasting takes place
121	Functional	Context	Items need to be traced within an enterprise	2	Goods and items need to be traced within one farm or enterprise.	Any item that has an ebbits identifier can be reliably located within a determined area.

122	Functional	Context	Items need to be traced between enterprise	2	Goods, parts and items need to be traced when they leave one enterprise and appear in another.	Any item with an ebbits identifier can be recognized when it leaves and enters an enterprise.
123	Functional	Communication	Information needs to be described in a standardised way	3	Enterprises working in the same sector adapt different ways to describe the input, the production processes, and the output; thus it will not be possible to communicate information either to providers or to consumers.	A standard for exchanging information between system components is provided.
124	Functional	Communication	The ebbits platform should amplify branding for enterprises.	3	Support for branding in the ebbits platform will help to overcome several challenges and needs to be analyzed in details in the ebbits business models. Thus, using and being part of the ebbits platform should be a quality attribute. This is a demand from consumer in future.	90% of such businesses exploiting the ebbits platform perceive this as a quality attribute.
125	Functional	Context	Associate meta-information to items	3	In parallel to the actual lifecycle (grow up of the animal, feeding, butchering, transportation, selling, consuming) there exists additional information such as the amount of food, medication an animal has had, the	Any item with an ebbits identifier can be associated with a set of meta-information.

					energy for the production and transportation, that needs to be acquired and associated with the (bits and pieces of) animal.	
127	Functional	Modelling	Batches need to be identified on a farm level	2	The average feed production batch size is 20 tons; however farmers want their own specific production which is way smaller. A unique identification of the batch to the farm is necessary.	Every batch has one unique identifier.
128	Functional	Modelling	A standard way of numbering batches needs to be provided	3	No world standard way of numbering the batches exists. The numbering depends on the local ERP system: Currently, one solution is to create a huge database to link the numbers together	A standard way of numbering/identifying batches exists.
129	Functional	Configurability	Energy consumption should be optimized automatically	3	Farmers want to optimize the energy consumption in their production and they adapt this manually, and they would like to have this done automatically (e.g. putting some production steps towards a cheaper time of the day).	Rules can be defined to automatically optimize the energy Consumption
130	Functional	Interface	ID Management : Item identification system should provide open	3	Identification of pigs is done with RFID tags at their ears and with	Any system can easily access the item identification system.

			interfaces to other systems		antennas in corridors that recognize pigs passing by. The system should not be connected to a specific system, but rather provide open interfaces that can be exploited by any system.	
131	Functional	Modelling	The system should support soft logic algorithms	4	The users want to know how trustful the data is.	The system supports at least two different soft-logic algorithms.
132	Functional	Devices	Device and service exception handling	3	The development and run-time environment should support exception handling constructs that the developer can employ to manage service and device availability and malfunctioning, isolated from the main application logic.	Exception handling constructs that the developer can use to specify exception responses with a success rate of 9/10.
133	Functional	Interface	Different views on the device ontology	3	It should be possible to present a developer user with different perspectives on the device ontology, depending on that users functional needs (e.g., a services perspective, device category perspective. etc.)	At least two different views are available in the ontology browser.
134	Functional	Configurability	The context model needs to be extensible during runtime.	2	A knowledge model enables the middleware to contain a representation of itself and manipulate its state during its execution. This feature should	Middleware is able to adapt its configuration in 60% of identified cases requiring reconfiguration.

					serve as the basis for self-adaptation of the middleware (e.g. reconfiguration of resource usage, triggering the component-based services).	
135	Functional	Security	Dynamically loaded libraries must undergo a security check before their usage	3	Dynamically loaded libraries (e.g. DLL, JAR, OSGI bundle) could contain malicious code.	Dynamically loaded libraries must contain a valid signature in order to prevent security breaches in the system.
136	Functional	Modelling	Handling of different device versions in device ontology	3	The device ontology should be able to handle different versions of a device.	The device ontology can maintain at minimum 2 versions of any single device
137	Non-Functional: Performance	Configurability	Systems built using LINKSMART should be scalable in terms of devices communicating	3	In large installations (e.g. in large factories) there will be many (embedded) devices in total. The middleware should support the development of such big systems.	The middleware supports applications in which more than 500 devices exist.
138	Constraint: Scope of the product	Architecture	The system should support distributed intelligence on embedded system.	1	We have a need for “intelligence” (Semantics, reflection etc.). We have a need for supporting embedded systems. This should not conflict	Minimum hardware requirements (which must be supported by all target hardware) are defined and all hardware that meets the specifications is guaranteed to work with LinkSmart.
139	Functional	Configurability	Support runtime reconfiguration	3	To supporting monitoring leading to adaptation, the architecture should be dynamic in the sense that	Services and devices can be connected during runtime.

					components/services should be connectable at runtime.	
140	Non-Functional: Operational	Devices	The middleware should monitor device's resource usage	3	The middleware should contain services that allow monitoring on what devices are doing. This includes monitoring response time, device load (e.g., CPU), and message interchanges per second.	Devices should provide an interface monitoring resource usage.
142	Functional	Modelling	Download and harmonisation of third party device ontologies	1	Device ontological models describing devices, which will be provided by manufacturers or third parties, should be automatically downloaded (updated) and harmonised to ensure the same ontological view. Formal definition of ontologies should be realised using the world wide accepted formats, recommended by W3C, such as RDF, OWL, OWL-S.	Ontologies from different manufacturers can be used if they are in RDF, OWL or OWL-S.
143	Non-Functional: Legal	Networking	Comply with industrial standards	2	The middleware should embrace existing industrial device integration and communication standards, e.g. EIB/KNX	Claimed support for any specific standard in ebbits can be verified using the conformance rules / procedures available from the issuing standards body.
144	Non-Functional: Usability	Configurability	Configurable and easy to install middleware	1	The middleware should be configurable and easy to install/deploy.	The average installation time is less than 1 hour.

145	Constraint:Scope of the product	Architecture	Support of low-end devices	2	Middleware must support low-end devices like RFID tags. Therefore, it must be compatible with at least 32-bit devices with < 512 KB RAM/FLASH or less. For smaller devices, proxies can be used.	Middleware is able to be installed and run on low-end 32-bit devices with 512 KB RAM/FLASH in 90% of all cases. Proxies can be created to support more limited devices in 40% of all cases.
146	Functional	Interface	Support for interfacing with external workflow systems	2	Applications must include workflow management possibilities	Supports at least one workflow system, for instance OpenWorkFlow.
147	Functional	Modelling	Waste of energy act definitions	3	Some users are wasting energy without realizing/being conscious that there are better alternatives.	Energy wasting behaviours are modelled
148	Non-Functional: Operational	Devices	Mobile management of farms	3	Mobility and mobile devices are required to manage aspects of the farm locally and remotely if you are on holiday or in the cinema.	Mobile access to important function of the management system is granted.
149	Functional	Context	Retrieve the behaviour on an individual animal level	3	Monitoring the drinking behaviour allows to recognize diseases 20 hours before. However, today it is not able to retrieve the drinking behaviour on an individual pig level, rather in a group. But with the emerging RFID identification it would be possible	The drinking behaviour can be retrieved on a individual pig level
150	Non-Functional: Operational	Devices	Applied stand-alone devices should have a	3	Data analysis is not done on the chip,	Applied stand-alone devices have a minimal

			long battery life span		because this consumes energy and the battery life-span comes to 3 months which is far too low.	battery life span of one year.
151	Functional	Modelling	Common structure of information is needed	3	Relevant data to describe specific situations of a pig needs to be determined and put into a common structure such as an ontology, a common definition of data.	Common data structure to describe situations of arbitrary entities is available
152	Functional	Modelling	The system should allow the correlation of information emerging from several sources	3	In order to easily analyse information, the system should allow for the correlation of information from different sources on a farm or enterprise	Acquired information is timestamped.
153	Functional	Modelling	Store meta-information with package labels	3	Temperature, location, humidity needs to be acquired and assigned with the packages. Dimension and weight is also required for the package, box, palette, etc.	A minimal set of 5 attribute-value pairs can be associated with any package/label.
155	Functional	Context	Synchronisation of Acquired Data is necessary	3	Data synchronization might be necessary, because data will be acquired automatically, manually, semi-manually with different timestamps.	A data synchronization component performs a timestamp-based synchronisation of a data set.
156	Non-Functional: Security	Security	Meta-information associated with entities needs to be trustworthy	3	ebbits needs to guarantee that the information associated with a specific product is	Acquired information cannot be manipulated by unauthorized people.

					the right and correct one that has not been manipulated.	
157	Functional	Interface	Different Views on the Data is necessary	3	We need services that provide different views on the data cloud by combining data from different sources.	Data can be filtered and sorted based on an arbitrary set of parameters.
158	Functional	Communication	Alarms are send when specific situations occur	4	Issue alarms when an animal or production machine behaves differently or abnormal. However, context-aware behaviour is wished for experts NOT for the farmers.	Simple rules can be defined that trigger alarms.
159	Non-Functional: Usability	Interface	End-users need to be able to management their distributed data	3	Farmers want to manage their distributed data, because today they have no full control of data.	End-users can easily manage data from distributed sources.
160	Functional	Context	The system should be self-ware	3	User statement: "Let me make my system self-aware"	User gets informed of any event she is interested in.
161	Non-Functional: Operational	Communication	Electronic exchange of reports between enterprises	3	Report on the slaughtered pig is sent back ON PAPER to the farmer. This should be done electronically, in order for the farmer to combine it with the feeding information, e.g. identification of a father that produces ill pigs.	Information is automatically put into a report format and sent to a recipient.
162	Functional	Security	Access-control of data sets	1	Access to data needs to be controlled, because some authorities require having access to this data, other stakeholders	Access rights can be defined for several stakeholder roles.

					might have restricted access, other information could be made publicly available.	
163	Functional	Communication	The traceability chain should be computed on demand	3	The traceability chain should be computed on demand and not stored and recalled, due to storage. Thus, the combination of data is the challenge and a handover of identities might be required. A combination of searches should be realized by semantics.	The traceability chain is computed and not stored.
164	Constraint: Ethical	unassigned	Consider ethical issues	3	It is an ethical issue if you track the driver driving a van or that a product is at the end-user's fridge	-
165	Constraint: External factor	unassigned	Consider cost benefit	3	There is no NICE-To-HAVE in the agriculture domain, there always needs to be cost benefit. A slaughter pig brings 5-15 Euros.	Cost benefit is considered.
166	Functional	Interface	Integration of legacy systems into ebbits platform	1	ebbits platform is deployed for solving interoperability of the existing systems (software and devices)	3 different existing systems used in each domain is supported e.g.: Farm : Climate, Farm Management, Feeding Slaughterhouse : ERP Body welding : ERP + Robot Management
167	Functional	Context	save historical information in farms	3	feeding history	historical information is saved
214	Functional	Architecture	Graceful degradation	2	The system should be functional even if some	System should be working even if 50

					parts are unreachable or non functioning	percent of sub systems are unreachable
215	Functional	Architecture	Automatic Recovery from communication failures	2	We need to able to still guarantee delivery of data/events also in case of minor/temporary communication disruption.	The system should be able to recover from a 10 minute communication failure
216	Functional	Context	Explicit model of context	3	It must be possible to trace events and data items across processes and workflows; context management is one of the mechanisms to support this.	The system supports an explicit model of context, which can be applied in at least two application domains.
217	Functional	Architecture	It must be possible to order events in the actual event sequence	2	The delivery of events received from different sources might not follow in the original sequence at transmission. There might be communication delays etc that make them arrive in the wrong order. Nevertheless rules should be able to express temporal/sequence dependencies on events that reflect the actual temporal event sequence at the sources.	It will be possible to express rules that contain temporal/sequence dependencies
218	Functional	Architecture	An event history should be maintained	2	Rule definitions can refer to past events, and behaviour can be defined based on that event history.	A rule that refers to previous events can be expressed.
219	Functional	Configurability	Event history size and/or time span should be	2	Different applications have different needs for	The event history can be configured wrt to

			configurable		the event history.	timespan and size.
220	Functional	Architecture	Event model based on common vocabulary	3	The ebbits platform must be able to handle a potentially large number of events on different levels of abstraction in the system architecture and with different semantics. The processing of events should also be related to the processing and management of data in ebbits, and this should be based on the use of common vocabularies.	The system can distinguish between several (minimum 2) event types based on a common ebbits vocabulary, represented in the event model.
221	Functional	Architecture	Events mapped to (business) rules	3	Events and services are basic mechanisms for the implementation of the (business) rules logic in the ebbits architecture.	Events of a at least two different event types can be detected in a (business) rule, as expressed in the ebbits business rules framework (based on the ISA-95 standard).
379	Non-Functional: Operational	Security	All stakeholders should be annotated with unique Id, type, name and relevant info.	2	It is important to recognise the users interacting with the system and their privileges and restrictions.	One or multiple directories of stakeholders
380	Non-Functional: Operational	Context	Monitored/sensed data should be contextualized (timestamp, geotag, type, etc).	3	It is important to know when and where data was sensed / monitored.	Semantic store with knowledge model for sensor readings.
381	Non-Functional: Operational	Modelling	Monitored/sensed data should be annotated (semantically) in local server/repo/store.	3	Information relationships should be available as soon as data enters the ebbits system.	Data acquisition, annotation and storing policy.
382	Non-Functional:	Context	Alerts should be	3	Generated messages	Alerting and messaging

	Operational		contextualized (timestamp, geotag, type, message, warning level, etc).		and alerts need to be traceable and provide rich information about the event detected.	policy.
383	Non-Functional: Operational	Modelling	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).	3	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.	Semantic store with knowledge model for devices. There is a unique mapping between physical devices and their semantic reflections.
384	Non-Functional: Operational	Modelling	Production should be annotated or modelled in order to calculate OEE and get number of orders/elements/products requested/delivered/in-progress/faulty.	2	Real-time traceability of produced goods/services is achieved by properly annotating their status and metrics during the manufacturing process.	Semantic store with knowledge model for production.
385	Non-Functional: Operational	Modelling	Logistic should be annotated or modelled in order to get information about element and consumables (present, in transit from supplier, ordered, etc).	3	The ebbits platform could provide also a system for efficient management of consumables and logistic aspects needed in (not only) manufacturing domains.	Semantic store with knowledge model for logistic.
386	Non-Functional: Operational	Modelling	Feedstuff should be annotated with origin, genetics, treatment, storage conditions and transport/delivery info (batch number, silo id, amount, timestamp).	2	A detailed annotation of feedstuff is required to the reasoning processes devised.	Semantic store with knowledge model for feedstuff.
387	Non-Functional:	Modelling	Animals should be	2	Proper identification of	Semantic store with

	Operational		annotated with RFID tag, weight, genetics, birth date, and current/historical (timestamped) data (growth/weight, location/movements, consumed feed, water, weaning, insemination, heat during pregnancy, born piglets, anomalies, vaccines).		animals and logging the most relevant information about their lives is vital for the traceability and quality control proposed in ebbits.	knowledge model for animals. Logging policy.
388	Non-Functional: Operational	Modelling	Meat packages should be annotated with ID of animal (for trace).	3	Meat traceability is one of the main added values of the ebbits platform in the agricultural domain.	Semantic store with knowledge model for animals.
389	Non-Functional: Operational	Modelling	Farm's soil/fields should be annotated with location, laboratory analysis info (date, sample field source, lab id/name, results, etc), current/historical data (types of crops, grain maturity, soil nutrients, quality of products grown, etc).	3	Different reasoning applications devised in ebbits for tracking the soil efficiency require a detailed annotation and logging of farms' soil.	Semantic store with knowledge model for farm soil/fields.
390	Non-Functional: Operational	Architecture	Farm's repository should store information about harvesting equipment, man power, etc.	3	The ebbits platform would provide also some functionalities for the management of resources needed for harvesting, thus they need to be included in the knowledge model.	Semantic store with knowledge model for farm resources.
391	Non-Functional: Operational	Modelling	Sow farm production should be annotated or modelled in order to allow	2	By proper reasoning and processing, the ebbits platform can exploit the	Reasoning algorithms for production tracking.

			tracking the number of piglets, pigs at fertile age, pigs ready to slaughter, maximum capacity, etc.		knowledge in the network and extract aggregated information required in real time.	
392	Non-Functional: Operational	Modelling	Halves of slaughtered pigs should be annotated with id, id of slaughtered pig, weight, fat thickness, date of slaughter, price, etc.	2	Traceability of meat requires proper tracking of pigs since birth to stores, thus the information after its slaughter is very relevant.	Semantic store with knowledge model for halves of slaughtered pigs.
393	Non-Functional: Operational	Modelling	Order should be annotated with type/ID of good/service, amount, price, dates(issue, expiry, delivery, etc).	4	ebbits platform can be exploited also for generic enterprise processes, like account management.	Semantic store with knowledge model for orders.
394	Non-Functional: Operational	Modelling	Invoices should be annotated with supplier's info (name, id, bank account, contacts, etc), goods or services info (type/id, amount, price, dates, etc).	4	ebbits platform can be exploited also for generic enterprise processes, like account management.	Semantic store with knowledge model for invoices.
395	Non-Functional: Operational	Modelling	Payment reports should be annotated with responsible id/name, bank, order number, status, etc.	4	ebbits platform can be exploited also for generic enterprise processes, like account management.	Semantic store with knowledge model for payments.
396	Non-Functional: Operational	Modelling	Generic Information should be annotated with requester, sender, content (price, capacity, dates), etc.	4	Information exchanged between stakeholders could be exploited for some reasoning, thus it is convenient to model it semantically.	Knowledge model for information exchange.
397	Non-Functional: Operational	Security	Stakeholders should be stored in local catalogues or external directories (advisory company, chamber of commerce,	2	In order to apply access control lists/policies, all stakeholders must be identified.	Stakeholder directories.

			etc) and accessed by the different subsystems inside and outside ebbits.			
398	Non-Functional: Operational	Security	Manufacturing Monitor System should have write access to local server/repo/store.	2	The reasoning processes devised in ebbits require access to knowledge/information found in local stores.	Access rules/policy granted for MMS.
399	Non-Functional: Operational	Security	Manufacturing System for Analysis should have read/write access to local server/repo/store.	2	Analysis/reasoning is based on local monitored data, and reports are sent back to local server/repo/store.	Access rules/policy granted for MMS.
400	Non-Functional: Operational	Security	Reports should have a list of allowed readers/subscribers.	3	Aggregated data, reports, alerts, etc, should be available only to stakeholders interested in them.	Access rules/policy for generation/reading of reports.
401	Non-Functional: Operational	Architecture	ebbits platform should have a list of alerts and subscribers.	3	The different monitored processes in ebbits should generate alerts and send them to the interested subsystems or stakeholders.	Directory of alerts/events.
402	Non-Functional: Operational	Modelling	Manufacturing Monitor System must have read access to internal and external environment data.	3	The reasoning processes devised by ebbits for the manufacturing domain require environmental monitoring.	Internal/external sensors collected and annotated in respective repositories.
403	Non-Functional: Operational	Security	Maintenance Manager and operators should have access to devices' and production info (proper ACL have to be implemented)	2	Some of the added values that ebbits will provide to managers in the manufacturing domain require a continuous tracking of the production processes, metrics and modifications	Stakeholders access policies for readings, devices and production.

					introduced.	
404	Non-Functional: Operational	Architecture	Farm's Management System should have access (through secure connection) to Feed Provider Resources Monitoring System.	3	The food traceability scenario requires an exchange of information between all the enterprises involved in the production chain.	Authenticated/secure access to Feed Provider RMS.
405	Non-Functional: Operational	Interface	Feed Provider should transfer delivery information about sent feedstuff.	2	The traceability relays on the successful exchange of information about the monitored processes linked to the tracked product.	Interface/procedure for request/receive/exchange of relevant information about deliveries.
406	Non-Functional: Operational	Architecture	Farm's Local server/repo should be accessible by RFID tag readers and National servers/repos/stores.	2	Information about animals is stored in farm local servers and used to retrieve information when reading RFID tags or when requested by National/European authorities.	Access policies to local server
407	Non-Functional: Operational	Architecture	Farm's Monitoring System should have access to local server/repo/store.	2	Monitoring system keep track of several process in the farm and needs the metadata stored in local server.	Access policies to local server.
408	Non-Functional: Operational	Architecture	Farm's Management Application Server should access local monitoring system servers/repos/stores for generation of reports	3	Management/accounting systems perform their tasks based on the information stored/provided by local monitoring systems.	Access policies to local monitoring system.
409	Non-Functional: Operational	Security	Consumer should have access to meat reports.	3	Relevant reports about the produced meat since the piglet born will be used by ebbits in order to enhance the information provided to consumers about the	Access policies to production reports.

					meat they are buying.	
410	Non-Functional: Operational	Security	Controller should have access to (online) queries to meat packages IDs.	3	Quality and health control authorities can rely on ebbits in order to track the distribution of meat packages when they discover some anomaly.	Access policies to production reports.
411	Non-Functional: Operational	Security	Farm's Management System should have access to field info repository.	3	Optimisation of the resources, like fields in an agricultural domain can be achieved by analyzing and processing information logged by their respective monitoring systems.	Access policies to historical info by Farm's MS.
412	Non-Functional: Operational	Architecture	Farm's Management System should have access to external information (crop price, fertilizers price, consumables price, weather, etc).	4	Consumables information can be exploited through ebbits in order to manage efficiently the farm's production processes.	Access to external/providers information.
413	Non-Functional: Operational	Architecture	Sow Farm Management System should have access to production/animal repository.	3	The knowledge obtained by tracking all production processes in the farm will allow managers to optimise them through a single platform.	Access policies to local server/repo/store.
414	Non-Functional: Operational	Security	Slaughterhouse Management System and Retail Management System should have access to both production (read) and slaughter (write) repositories.	3	Accounting though ebbits will be simplified thanks to the knowledge acquired by multiple systems in the food production chain.	Access policies to production and slaughter servers/repos/stores.
415	Non-Functional:	Interface	Accounting Management	4	The ebbits paradigm	Access policies to

	Operational		System should store orders, and have access to Supplier Management System (to send/receive orders/acks/invoices).		can be exploited also for improving the efficiency of accounting task.	Supplier MS.
416	Non-Functional: Operational	Interface	Accounting Management System should have access to bank's management system for sending/receiving payment orders/confirms.	4	The ebbits paradigm can be exploited also for improving the efficiency of accounting task.	Access policies to Banks' MS.
417	Non-Functional: Operational	Security	Manager should have access to directory of stakeholders to interact with them (send/receive info).	3	The ebbits paradigm can be exploited also for improving the efficiency of accounting task.	Access policies to Stakeholders directory.
450	Functional	Context	The system should compensate deviations of incoming data.	3	The incoming data could contain outliers e.g.: spikes which should not influence the measurement.	System provides configurable filter to exclude outliers e.g.: define upper and lower threshold
451	Functional	Architecture	Sensor fusion algorithm must be added during run-time in a modular and extensible way.	2	Sensor fusion algorithms vary greatly and can't be generalized only in one module.	New sensor fusion algorithms can be added in a pluggable way
452	Non-Functional: Maintainability	Architecture	Sensor fusion algorithms must be realized as a decoupled component.	3	Sensor fusion algorithms can be re-used by several other components.	Unrelated components are able to use sensor fusion modules
453	Functional	Interface	The system must be able to assign fused data as a context attribute of an entity	1	Entities cannot provide their own context values, therefore sensors are needed to provide their context values. e.g.: a thermometer is needed to provide the temperature of a room.	The system allows relationship among context of entities and sensors to be modelled.
454	Functional	Context	The system must monitor	1	Continuous monitoring	The system should

			the state of devices and entities.		of entities is needed to detect anomalies (e.g.: ill Pigs, overheated welding gun)	monitor and analyze sensor events
455	Functional	Context	System needs to trigger business events based on changes of devices and entities states.	1	Enterprise applications have to be notified when the process starts and finishes, and further how much resources have been consumed for the process.	System generates business events defined in WP3.
456	Functional	Architecture	The system should be able to process a large number of sensor events	3	A Manufacturing site has at least 500 sensors	System is able to process at least 500 events / second.
458	Non-Functional: Security	Security	Libraries must only be accessible only for permitted applications.	4	Libraries could contain functionality that should not be available to all kinds of applications (e.g. calculation of quality rating of meat should only be allowed for slaughterhouse application but not for consumer application.)	The dynamic loading of libraries must be restricted through policies.
459	Non-Functional: Operational	Devices	Mobile access to farm data in the ERP system	3	Mobility and access to mobile devices are required to manage aspects of the farm remotely if you are outside your office.	Mobile access (at least a read one) to the data of the ERP system is granted and solution should be browser-based.
461	Functional	Architecture	Dependencies on past events possible	3	An action executed by the system may be dependent on more than one event, and some of them could have occurred in the past.	Actions executed by the system can be processed dependent on current events as well as events having occurred in the past.
462	Non-Functional: Performance	Architecture	Scalable event processing	1	The platform must be able to handle a large number/high frequency	The platform can be configured to handle a span from low frequency

					of parallel event streams.	processing (10 e/s) to high frequency processing (50000 e/s).
463	Functional	Architecture	Semantic event processing	3	It must be possible to interpret events in the context of the different layers in the architecture (from PWAL to a business rules layer).	The system provides at minimum two layers of event processing where events can be captured and possibly filtered/fused
464	Functional	Modelling	OWL-lite will be used to model ontologies	3	Appropriate knowledge representation formalism is needed to avoid the high complexity in the knowledge manipulation process.	The knowledge representation formalisms must be as small and easy as possible, but as expressive as necessary for our scenarios.
465	Functional	Modelling	Ontology namespace	3	Common ontology namespace is important.	ebbits ontologies will share namespace http://www.ebbits-project.eu/ontologies
466	Non-Functional: Usability	Interface	Query the ontologies conveniently	3	Easy query possibility	SPARQL is used as query language, or an abstraction is used if this turns out to be more usable.
467	Functional	Modelling	Only relevant parts in the ebbits ontologies	3	The development of the semantic models must be driven by the real use-cases instead of the theoretical assumptions to avoid the unnecessary complexity in the knowledge.	Ontologies are relevant for ebbits use cases.
468	Functional	SDK	ebbits platform should support automatic builds	3	To make the ebbits platform easier to use for future developers it has to be easy to build and start. Else developers will not take	All demos and prototypes have to provide a uniform method for building and starting. The build process has to be decided in advance else

					it serious and will not try it.	it will take extra effort to realize it.
469	Functional	Middleware Layer	Local LinkSmart instances should properly handle local variables when remote environments are restarted	3	LinkSmart should be more robust to different network issues, like temporary offline periods or restarting of remote instances	The critical event of LinkSmart environment reboot while developing is handled by a specific exception
470	Functional	Architecture	PWAL should support reconfigurable dynamic polling policies	3	Applications could have different polling needs, which eventually could change in runtime, so the PWAL must offer an easy reconfiguration of the polling policies per parameter and per application	The PWAL supports the dynamic polling policies of a PLC involved in a manufacturing plant and provide instruments for polling configuration.
471	Functional	Architecture	ebbits should implement a distributed time dissemination and synchronization service	3	Several application in ebbits relay directly or indirectly on accurate timestamping of data and events, thus given the distributed nature of ebbits, a time dissemination and synchronization service is required within the platform.	ebbits provides a time dissemination and time synchronization service
472	Functional	Architecture	PWAL should support accurate timestamping of data acquainted	3	The PWAL should be able to properly handle time information of the data and events it access/generate. This handling must include the synchronization to the ebbits time dissemination service and compensation of hardware and	The PWAL, while managing data or events, handles time-related information through the interaction with the time synchronization services.

					communication delays if possible.	
473	Functional	Architecture	PWAL should expose suitable methods in order to enrich raw data	3	Data acquainted through the PWAL needs to be enriched with meta-data (like source, geotag, timestamp, units, etc), which will be then used by upper layers and applications. Being the PWAL the lowest link between ebbits platform and the physical world, part of this meta-information could be already attached at this level, easing the processing of it by the multi-sensor fusion and context awareness services.	Meta-information could be attached to data or events at PWAL level
474	Functional	Devices	PWAL should be able to match PLC symbols with ebbits ontologies	3	The definition of symbols on an OPC server (i.e., variables of interest inside the PLC) could be made in accordance with the PLC programmer according to an agreed convention that could be exploited for an automatic matching with a device catalogue or ontology	The PWAL includes mechanisms to associate custom PLC symbols to ontologies selected within the ebbits environment.
475	Functional	Devices	PWAL should adopt a lock and semaphore-based policy to the access of PLC memory	3	Since different applications could eventually be interested in a common variable, the PWAL must assure	When the PWAL tries to read or write some critical variable onboard the PLCs, all variables exposed are safe

					its access is controlled in order to avoid collisions in concurrent requests, as well as possible locks or restrictions to specific applications	
476	Functional	Devices	PWAL should implement an error control strategy to assert correct data type and values written to the PLC	3	Errors in writing variables to the PLC must be avoided at all cost, since they can lead to a halt in the running program. The PWAL has to adopt a suitable error control strategy in order to assert data has been introduced correctly (this eventually would require a control logic in the PLC program as well).	The PWAL always writes on the PLC memory the appropriate data types, consistently with variables and respecting the appropriate value ranges.
477	Functional	Architecture	PWAL should implement a heterogeneous multi-data aggregation in single events.	3	Event-driven data acquisition can easily generate scalability issues if single events are generated per sample. Thus aggregation of several samples in a single or few events has to be devised.	It is possible to aggregate events coming from the same source at PWAL level
478	Functional	Architecture	PWAL should expose basic feature extraction and sensor fusion functionalities (e.g., moving average, decimation, filtering, etc) in order to minimize scalability issues	3	Some variables gathered through the PWAL could require high sampling frequencies and maybe just some feature of the acquired signal is of interest, so in order to	The PWAL offers methods to extract features and to exploit the sensor fusion

					save some bandwidth and avoid scalability issues, the PWAL could offer some basic feature extraction and sensor fusion capabilities.	
479	Functional	Devices	Multiradio devices should be able to detect which LinkSmart Network Manager to connect/migrate to, according to the current network interface active	3	Devices with multiradio capabilities should be able to switch interface, and therefore network, without compromising the connectivity to the LinkSmart layer, this means that when migrating to a new interface, the device should register itself to the closest Network Manager available in that network	Devices accessing ebbits by using non-corporate or external networks (e.g. 3GPP) can detect which border network manager they must connect to.
480	Functional	Architecture	Multiradio devices should avoid re-generation of HIDs when migrating to a different LinkSmart Network Manager	3	Devices with multiradio capabilities should be able to migrate to different networks without affecting the functionalities at the LinkSmart layer, including identificability, thus when registering to a new Network Manager, the device should try to register itself with the previous HID, triggering an unregistration request to the previous Network Manager	Devices with multiradio capabilities don't affect the LinkSmart layer while switching radio interface, thanks to a proper HIDs assignments' mechanism.
481	Functional	Architecture	Multiradio devices should be use local data caching	3	Devices with multiple radio interfaces and in	ebbits allow multiradio devices to use local data

			and delay tolerance networking		general devices with delay tolerance networking capabilities may experience periods with no networking, where some events may have happen, thus a local caching of its data may prevent any loss of data.	caching and delay tolerance networking.
482	Functional	Architecture	Multiradio devices with local data caching should implement suitable application specific data-expiration policies in order to prevent cache overflows	3	Devices with local data caching for delay tolerance networking, may exhibit cache overflows if they generate big amounts of information and/or experience large offline periods, thus data-expiration policies need to be applied in order to prevent this.	The platform prevents the data loss of devices with multiple radio interfaces (e.g. in case of loss of network intervals)
483	Functional	Architecture	Multiradio devices should be able to gather information about their network interfaces needed for the selection policies.	3	Multiradio devices must be able to collect and expose some information about their interfaces, like throughput, energy consumption, cost of traffic, quality of service, between others. Such information will be useful for defining interface selection policies.	Multiradio devices can expose information about network interfaces
484	Functional	Architecture	] Multiradio devices should select the most proper network interface according to the application requirements	3	Depending on the properties of the information (e.g., importance, quality of service, timeout, etc),	Multiradio devices are able to select most proper network interface consistently with the application implemented

					multiradio devices should select the network interface most suitable to the requirements of the application accessing it, which could be a energy or cost saving policy for instance, or an urgent event that should be transmitted at all costs	
485	Functional	Communication	6LoWPAN networks should include frequency agility features in order to enhance the overall system reliability	3	The ability to jump to a different channel automatically according to the channel occupancy or interference seems a promising solution in order to cope with the high electromagnetic pollution present in manufacturing scenarios, thus a frequency agility service should be included in 6LoWPAN networks	The ebbits platform includes frequency agility features in order to increase overall system reliability