Available online at www.sciencedirect.com

SciVerse ScienceDirect

Procedia Computer Science 16 (2013) 383 – 392

Procedia Computer Science

Conference on Systems Engineering Research (CSER'13)
Eds.: C.J.J. Paredis, C. Bishop, D. Bodner, Georgia Institute of Technology, Atlanta, GA, March 19-22, 2013.

Ontology for Systems Engineering

L.C. van Ruijven

Manager Technology Delvelopement, Croon TBI Techniek, Schiemond 20-22, 3024 EE, Rotterdam, Netherlands

Abstract

This paper presents an ontology for systems engineering, by means of a set of information models based on a modelling methodology derived from and a simplification of the ISO 15926 part 2 data model. During the engineering phase of several capital facility projects, the methodology has shown to be much more readable by and helpful in the communication between engineers than the mentioned data model. The ontology represents an interpretation of the Systems Engineering processes as described in ISO 15288 (System Life Cycle Processes) by the author based on years of experience with this standard.
Based on the information models and the set of relationships defined herein one can, in combination with a reference data library, specify, build or configure an information system or data exchange mechanism in order to support Systems Engineering processes or set up a specific product knowledge model. Implementation can be done by using a centralized information management system or a distributed one, based on data exchange between involved parties. Both implementations have been validated within pilot projects respectively based on the software tool Relatics and based on a RDF application using triple stores.

Keywords: Ontology, Systems Engineering, ISO 15926, ISO 15288, Information models, RDF

1. Introduction

The study of NIST "Cost Analysis of Inadequate Interoperability in the U.S. Capital Facilities Industry" quantified an annual cost of approximately $15.8 billion resulting from inadequate interoperability in the U.S. capital facilities industry (realizing process plants, ships, roads) over 2002, representing approximately 2 percent of industry revenue. Interoperability problems in the capital facilities industry stem from the highly fragmented nature of the industry, the industry's continued paper-based business practices, a lack of standardization and inconsistent technology adoption among stakeholders (NIST 2002).

A research on project results by USP Marketing Consultancy in the Netherlands also shows that the major reason for failure costs is lack of adequate information exchange and communication within projects. 25% of these failures arise during the design phase of a system which can be traced back to lack of interoperability between parties involved in the systems life cycle processes. This paper addresses the interoperability issue in the area of Systems Engineering also known as the Systems Life Cycle Processes as defined in ISO 15288.

Based on several experiences of the author with implementations of SE in some major infrastructure projects in the Netherlands, in general implementation of SE is generally lacking integrality and consistency. This is due to an island approach of the divers SE processes during development and implementation of these processes. Furthermore, every company interprets the abstract process descriptions of the ISO 15288 in a slightly different way due to the usage of natural language to describe these processes. The island approach and the different interpretations of the ISO 15288 lead to interoperability problems when several parties try to work together in a joint project, resulting in inefficiency and failure costs [5]. This is why the need arose to define these processes in an explicit, unambiguous way. A solution was found in modeling the processes defined in ISO 15288 by using the ISO 15926 standard. This was done by identifying relevant entities within the ISO 15288 processes and defining explicit relationships between associated entities, ending up in a set of information models built upon unambiguous defined entities and relationships. This approach has shown to significantly improve the integrality and consistency of the SE process descriptions and therefor minimizing the conceptual interoperability barrier [10].

Reviewers of this paper agreed that the topic of ontologies is important for systems engineering and that there is merit in examining the suitability of ISO 15288 as an ontological foundation. The reviewers agreed that the approach lacks justification and also lacks validation criteria which might be employed, and lacks proof of how those criteria were assessed. There is given little insight into how the ontology was used in practice at all. The author explains as a reply that the presented ontology originate from knowledge and years of experiences as a systems engineer within several projects based on ISO 15288 in the infrastructure and several research projects in the area of ISO 15926 and BIM. The presented ontology is currently used in major infrastructure projects in the Netherlands. The presented ontology was operationalized within these projects using the information management tool Relatics. Figure 19 shows a screen dump of Relatics with the implementation of the verification model as shown in figure 11.

2. Background of the information models

Outcomes of the European ORCHID workshop [1] showed that engineers and project leaders lack modeling skills and need to be supported in their project and or engineering knowledge modeling activities by a simple-to-use methodology that is close to the human way of thinking and communicating. The use of commonly shared modelling techniques would improve interoperability and therefore effective communication between and within the partner organizations in a project. Additionally, the presented models in this paper offer industry groups a way to set up their product model using the presented low-level modelling methodology based on 'facts' known from Gellish [4] which can be seen as a 'controlled natural language', enabling to describe the world in a unambiguous way.

A 'fact' can be defined as "that which is the case", a statement. A fact can be used to classify things as 'being the case'. Facts can be expressed in language as relationships between two separate roles. Each fact follows the same pattern: object (role 1) – relation – object (role 2), A fact must always be readable in two directions, from left to right and from right to left (figure 2). So each relationship has two names: the 'prescribed' name, reading from role 1 to role 2, and the reverse name, reading from role 2 to role 1. The usage of facts can replace complex data models and provides an extensible ontology with reference data for defining, harmonizing and communicating about systems. To achieve this goal, the methodology presented in this paper uses a taxonomy of relationships and a taxonomy of 'things' to describe the world in a consistent and explicit way. This is provided by the rule that everything in the real world must be classified by a class in a reference data library such as ISO 15926 part 4 [3]. In figure 1 the object airco xyz is defined as an 'air-conditioning unit' and capacity airco xyz is formally defined as a 'nominal cooling capacity' and is quantified as 100 kW. The above-described approach is comparable with the Resource Description Framework (RDF). Section 3 will explained what RDF is and how RDF is used to implement the presented information models within a pilot project in the shipbuilding industry in the Netherlands. The aim of this project is to enlarge the effectiveness of product data exchange within the supply chain in the area of shipbuilding.

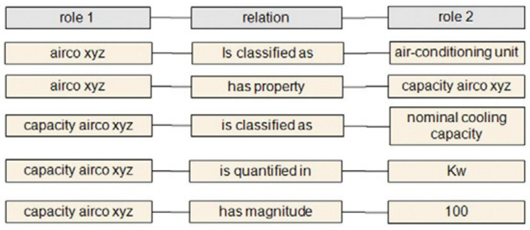


Fig. 1. Example of a set of facts defining a property of an airco

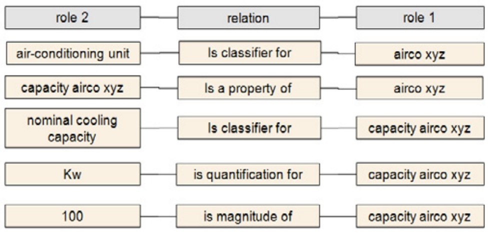


Fig. 2. Reverse names of the relationships named in figure 1

Based on the process description in ISO 15288, a set of information models are presented in this paper which together forms an integrated and normalised representation of Systems Engineering processes as described in the ISO 15288. The set of information models can be seen as an ontology for Systems Engineering in the context of infrastructure projects in the Netherlands which has been validated within several infrastructure projects in the Netherlands (e.g. Sluiskiltunnel) using the of the shelf, WEB enabled sematic information management system Relatics. Within these projects, all project partners (multidisciplinary) use and share the same way of defining a risk, interface, deviation, requirement, verification, deviation, the System Breakdown Structure and the Work Breakdown Structure which improved significantly the interoperability within these projects.

The information models also enable product manufacturers and/or suppliers to create ‘product knowledge models’ of their products in such a way that product information can be easily integrated in a systems engineering environment based on the same information models.

The model based approach of SE in this paper is described by means of information models representing:

- the process side and physical side of a system (figure 5)
- the breakdown structure of the physical side of a system (figure 6)
- the characterisation of objects by means of properties (physical quantities) and status (figure 7)
- the interactions between system elements mutually and between system elements and the environment and or stakeholders (figure 8)
- the failure mode and effect analysis of systems (figure 9)
- the requirement analysis of a system (figure 10)
- the verification of a requirement specification (figure 11)
- the risk management information model (figure 12)
- the contract change management process (figure 13)
- the work breakdown structure of a system (figure 14)
- the information model of a measure (figure 15)
- the information model of a assumption (figure 16)

The used relationships in the information models are derived from ISO 15926 part 2 entities and can in general be seen as short cuts in the context of the ISO 15926 part 2 data model which is a complex, highly normalized and abstract data model which is difficult to understand, even for IT specialists.

For reason of readability the presented information models have been simplified by ignoring the difference between types of entities (object types and types of relationships between object types) and instances of these types. ISO 15926 makes importunately a difference between those. Also for reason of readability only the prescribed name of each relationship is given in the models. The models can easily be expanded with private classes and relationships, according the shown method. Important to know is that each entity and relationship in the presented information models has a formal definition which will be defined in the ISO 15626 standard (respectively part 4 and part 11).

All relationships used in the information models can be dependent of the application (e.g. capturing knowledge or real project data) in one of the modes ‘shall be the case’, ‘can be the case´ or ‘is the case’.

Besides the data model of ISO 15926, the so called ‘Hamburger model’ principle (Functional Unit and Technical Solution model, figure 3) is used as a base to build the information models. The hamburger model has been developed in the early days of ISO 10303 (STEP) as the General AEC Reference Model (GARM) [8]. This concept makes a clear distinction between a certain abstract object (represented by e.g. a symbol on a drawing) and the materialized version of it (as delivered by a supplier and installed in place). The Functional Unit can be seen as representing all the requirements (functional, technical, performance, constraints and other aspects including e.g. safety, esthetical and maintenance). A Functional Unit captures a “design problem” and “exists” in principle as long as the life cycle of the total system lasts (a ‘whole life individual’ in ISO 15926 terms). The Technical Solution can be a principle or technology or a tangible piece of equipment, described with a set of characteristics which fulfil all the requirements of the Functional Unit. The selected Technical Solution can be replaced more than once during the life cycle of the total system. In the presented approach is a Functional Unit can be either a Functional Object or a Functional Physical Object in the context of ISO 15926.

Based on the Hamburger model one can build a hybrid System Breakdown Structure representing the Functional Objects, design principles and or Technical Solutions on the various levels of the break down structure (figure 4). This approach doesn’t require an additional functional break down with a mapping to the physical break down structure anymore as traditionally is done.

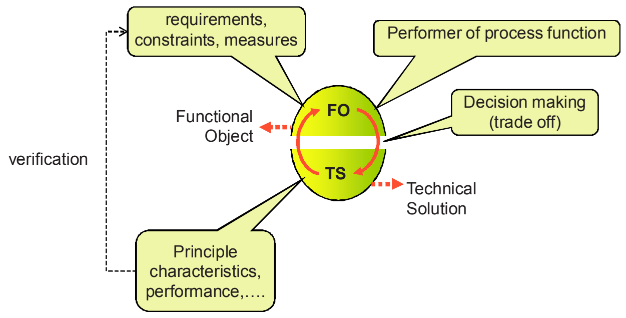

requirements, constraints, measures
Performer of process function
Functional Object
FO
Decision making (trade off)
verification
TS
Technical Solution
Principle characteristics, performance,….

Fig. 3. The Functional Object – Technical Solution principle from the General AEC Reference Model (GARM).

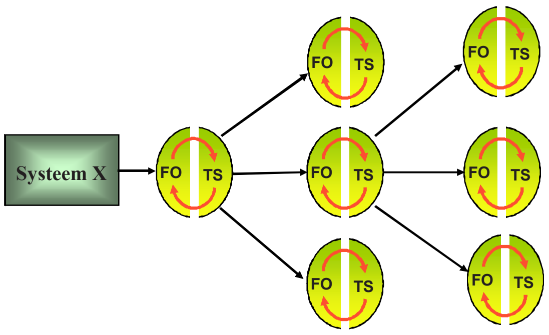


Fig. 4. The Functional Object – Technical Solution principle used in a System Breakdown Structure.

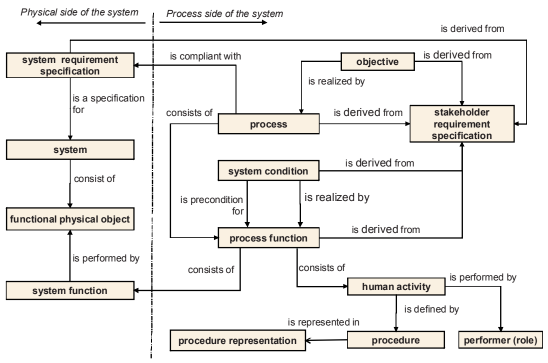
3. Information models of SE processes

In this section the most important information models will be explained which together represent all the basic entities that are needed to describe and communicate about Systems Engineering activities.

In principle, development of a system starts with an objective for that system (figure 5). There is at least one objective for the target system and one objective for the (enterprise) system that realizes the target system. After objectives are stated, the system life cycle processes (e.g. the operation process or the design process) are needed to achieve the objectives (according to In 't Veld, [12] objectives need processes to be realized). A process is defined as a coordinated set of activities (ISO 15926-4 [3]).

The next step is to define process functions, which are a kind of sub processes that will allow the processes to perform in such a way that the objectives will be achieved. On the one hand process functions realize process conditions and on the other hand there are process conditions that are preconditions for process functions. A process function can be performed by a functional object and/or human activity. At this point of a design an initial decision is made on the level of automation of the process function and thereby the responsibility of the operator/maintainer in achieving the objective of the system.

In principle, systems consist of functional objects which perform one or more (technical) system functions. System functions contribute to process function. Objectives, processes, process conditions and process functions and requirement specifications are derived from stakeholder requirements. Figure 6 shows the information model for the System Breakdown Structure (SBS).



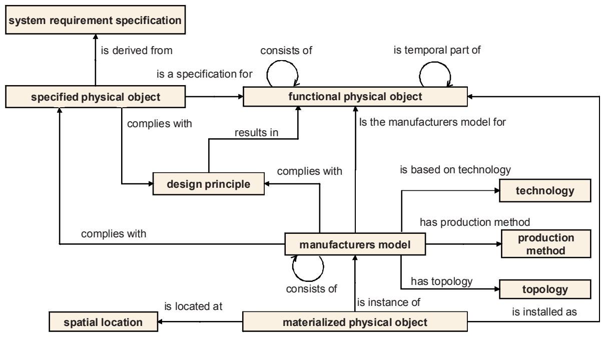


Fig. 5. Information model representing the process side and physical side of a system

Fig. 6. Fundamental physical system elements

In figure 6, four levels of realisation of a physical object are recognised:

- functional physical object (e.g. a pump as presented on a P&ID)
- specified physical object (e.g. pump specification)
- manufacturer's model (e.g. pump model xyz from manufacturer abc)
- materialised physical object (e.g. pump of model xyz with a specific serial number)

The manufacturer's model as a chosen solution for a functional physical object will be valid during a temporal part of the functional physical object and will have a unique instance during a temporal part of the life cycle of the system (e.g. pump model xyz which is replaced by another model). A manufacturer's model complies with a design principle and is based on a specific technology, has a specific production method and has a topology.

In the 'Hamburger model' the functional physical object and specified physical object are combined in one object: the Functional Unit. The design principle can be seen as a high level Technical Solution. Another design principle will lead to another set of new Functional Units on the next level of the system breakdown.

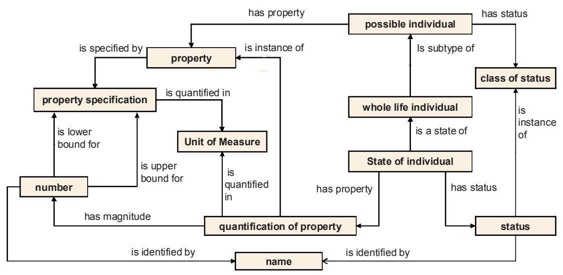


Fig. 7. The property and status model of possible individuals

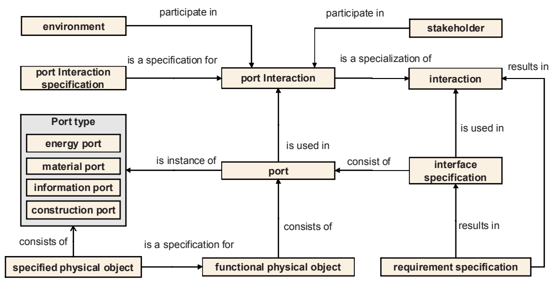


Fig. 8. Port principle to model all relevant interactions within and between systems and the outside world of the system

Figure 7 shows the information model for properties of any element ('possible individual') of a system, with a specific approach for unique instances of classes (a 'whole life individual' in a particular state, being the physical condition). As long as an element of a system (such as a physical object, activity or event) exists on the class level or on the specification level (see figure 5) a property is accompanied by a property specification, defining the range (upper and lower boundary) and the unit of measurement in which the property is expressed. The property specification can be used in a product knowledge library to specify several potential permissible expressions of a property value. Every system element which is in a specific state can be described using a property that has a specific magnitude. A property can be expressed in a unit of measurement or in a string value. In the last case the number can be identified by a specific name; by means of the property specification several numbers can be defined in an ordered manner, each with a given name as possible value for that property (e.g. expressing hazard classification in terms of category I, II or III). More or less in the same manner, the possible status (a situation at a particular time) of a system element on class level can be defined by defining a class of possible values of the status, linked to a name. On the instance level of a system element the status of an individual can be identified by one of these values and or name.

Figure 8 shows the port principle as used in this methodology. Physical objects interact with their environment, human or other system elements by means of an interface which consist of one or more ports. All interactions that occur during the operational life cycle stage of the system can be defined using this port principle (also derived from the ISO 10303 STEP developments).

The interface of a physical object exists of ports. There are four basic types of ports: material ports, energy ports, information ports and construction (3D) ports. Interactions by means of a port can take place between functional objects and between functional physical objects and the environment and/or stakeholders. An interaction with corresponding ports becomes a physical connection when a functional physical object is realised by a materialised physical object. An information port for user interaction can be materialised by e.g. a push-button panel. An energy port defined as electrical power can be materialised by a power plug, connected by a cable, a material port by a flange, a construction (3D) port by mounting a cabinet on a wall with concrete plugs.

Based on the models presented in figures 6, 7 and 8, manufacturers can set up their own product knowledge model that can be used and integrated in the systems engineering process. Besides, organisations can set product knowledge models in a specific domain that can be reused in comparable projects e.g. a complete tunnel design. This approach to capturing and consolidating product information can be a means to achieve configuration management for a specific system or product.

Figure 9 shows the information model for failure modes of a system. This model is based on determining possible malfunction of a system function performed by a physical object. A failure mode is a reason for defining one or more maintenance activities, test activities and/or safety measures that are defined in a procedure and/or document. Failure modes that are detected by a control system signal result in a maintenance alert and/or an operator alert, which will function as input for a maintenance or operator activity. To execute a maintenance activity on an instance of a manufacturer's model, it may be necessary to use a spare part that is also a manufacturer's model in itself.

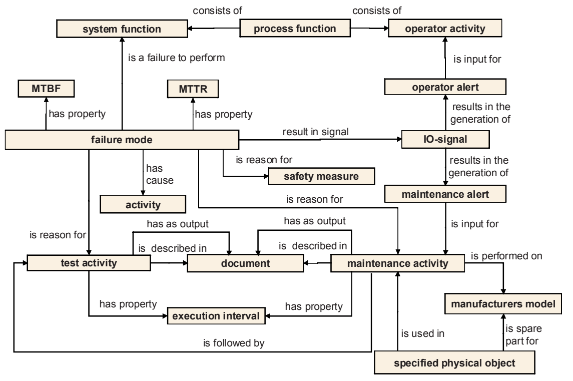


Fig. 9. Information model supporting the failure mode and effect analysis process (FMECA)

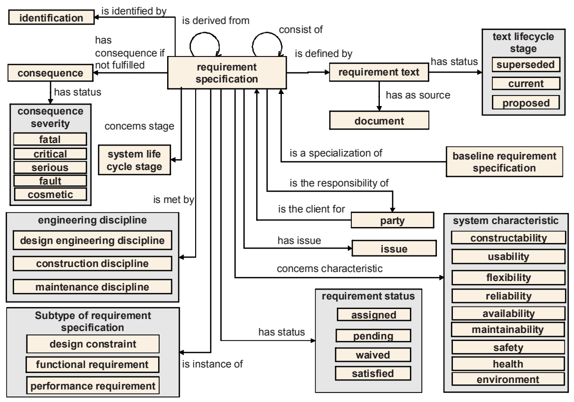
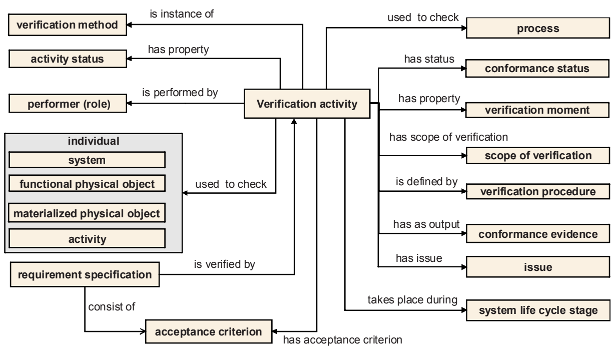


Fig. 10. Information model of a requirement specification

Figure 10 presents the requirement analysis process. Requirement specifications are classified in terms of severity, engineering discipline, type of requirement and the system characteristic addressed by the requirement specification. The requirement specification is allocated to a party that functions as 'client'. In general, a requirement specification will be a design constraint, functional or performance requirement.

A requirement specification is defined as a piece of text that has a status in the context of its life cycle and has a source. A requirement specification itself also has a status. Possible status is e.g. waived when the requirement is no longer relevant or fulfilled when it has been proved satisfactory.



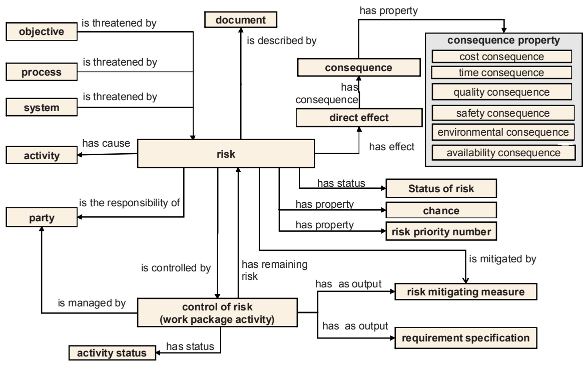


Fig. 11. Information model of a verification activity

Fig. 12. Information model for a risk and risk control activities

Figure 11 shows the information model of the verification process. The fulfillment of a requirement specification must be demonstrated to the client in order to verify that the product or system will be or has been built in the right way. In principle, each requirement specification has one or more related verification activities, depending in which system life cycle stages verification is required. One requirement may have to be verified in the design stage and also in the construction stage; another requirement may only need verification during the maintenance stage of the system. Each verification activity is related to the specific item that is checked. This can be an individual, specific system, object or activity (see also figure 6 and 14). Also a process (which is a type of object) can be verified. This means that if a specific requirement specification related to a specific system is checked, in for example, three life cycle stages of that specific system, three verification activities exist, each having its own method (e.g. review, calculation, simulation, inspection), scope (e.g. full scale, sample, typical) and procedure (test protocol).

For each verification activity it might be necessary to specify an acceptance criterion due to the vague character of the requirement specification. In fact, the specified acceptance criterion is in that case an extension of the requirement specification and becomes a part of it. The verification activity may be dependent on a conformance status, a source for a contract deviation, e.g. when the requirement specification appears not to be feasible. A specific verification activity always has a status (e.g. active, finished) and will be performed by a role (e.g. design manager). For validation activities a model as in figure 11 can be used, only verification has to be replaced by validation and requirement specification has to be replaced by stakeholder requirement specification and or objectives.

Figure 12 represents a model for risk management. Risk management requires control of information that characterizes a risk and information that comes with mitigating a specific risk. In principle, a risk threatens an objective and the objective-related system and/process. A risk has a cause and effect and is the responsibility of a person or organization. The effect leads to consequences concerning the cost, time, quality, safety, environment and/or availability of the system. Rating these consequences and multiplying the mathematical sum of them by the chance gives rise to a risk priority number which can be used to prioritize a set of risks. The risk is managed by a person or organization and ends up in one or more mitigating measures or one or more requirement specifications. Fulfilling these measures and requirement specifications leaves a residual risk with a new chance and set of consequences resulting in a new risk priority number, which must be significantly lower than the risk priority number of the primary risk without the measures and/or requirements.

Figure 13 represents the process of contract change management. A central entity is a contractual delivery which is stated to be delivered in the contract. The contract is between a principal and a contractor where sometimes a third party is involved (stakeholder, preferred supplier etc.). A contractual delivery must be compliant with one or more requirement specifications (the 'what') and consist of work package deliverable items (see figure 14). The whole set of work package deliverable items should satisfy the set of contractual deliverables. In order to be able to phase the work, several milestones will be defined where a part of the contractual deliverables will be accepted and hence lead to payment. These financial milestones are specific milestones as shown in figure 13. An issue relating to a contractual deliverable can, after clarification, be reason for a contract change proposal described by a document that may result in a contract extension defined in a document.

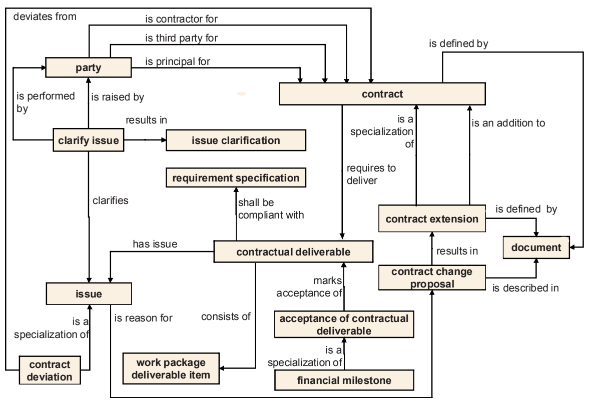


Fig. 13. Information model for contracts and contract changes

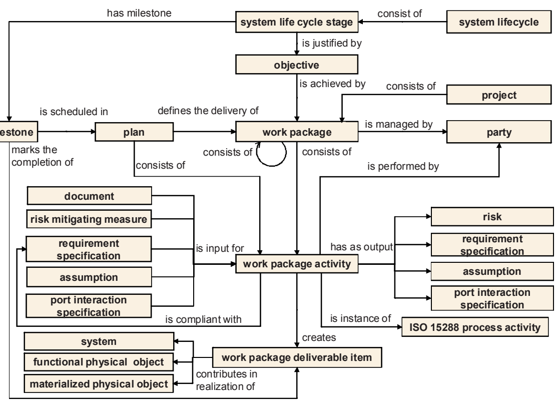


Fig. 14. Information model for the items relevant in the context of a work breakdown

Figure 14 shows the information model for the Work Breakdown Structure (WBS) of a system, capturing the whole system life cycle, which is typically divided into several stages. Each stage is justified by one or more objectives ('what has to be achieved when the stage is finished'). These objectives are achieved by realizing one or more work packages. A work package comprises one or more activities creating a deliverable contributing to a subsystem, a specific functional physical object or materialized physical object. Both life cycle stages and work packages can have milestones, each of them scheduled and described in a timetable. Special milestone are financial milestone as agreed upon in the contract. To execute a work package activity that creates a work package deliverable, input will be needed from documents, the risk management process (measures), requirement analyses process (requirements) and design process (assumptions and port interactions specifications). The activity will also lead to new risks, requirements, assumptions and port interaction specifications. Every work package activity will in principle be an instance of an activity in the context of one of the ISO 15288 processes. By making different views one can get a total overview of a specific type of activity e.g. all welding activities, sorted per life cycle stage.

For several entities in afore given information models an information model can be made for that specific entity itself. The following two examples are given: a measure (figure 15) and an assumption (figure 16).

A type of measure is a 'risk mitigating measure' as introduced in figure 12. A measure is characterized by a specific workflow: it will be proposed by the party that is managing the controlling activity of the risk. The measure will be approved by those parties that will be affected by the measure. The measure will get a status e.g. 'approved' and there will be a party that is responsible for executing the measure. There will be a specific work package activity that will really execute (perform) the measure. When the measure is successfully executed the status of the measure e.g. to 'executed'. In that case in principle there must be a document that proves the execution of the measure. During the workflow statements can be made by the various parties to explain the course of the measure. A measure can be classified by a property that represents the effectiveness.

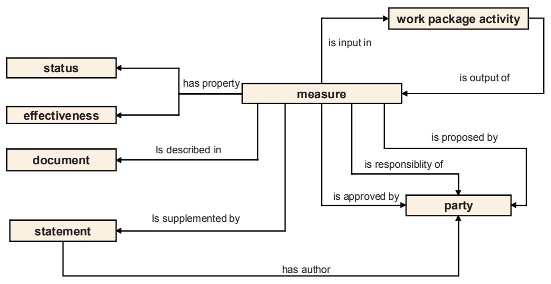


Fig. 15. A basic information model for a measure

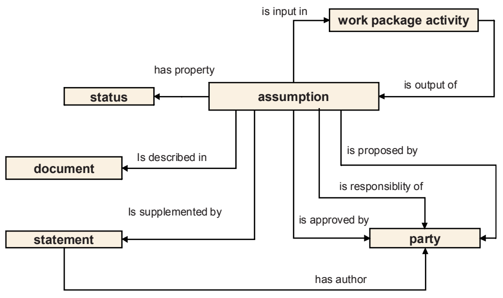


Fig. 16. A basic information model for an assumption

An entity assumption is introduced in figure 14. In figure 16 the information model is presented of an assumption. An assumption is also characterized by a specific workflow: it will be proposed by the party that is executing a work package activity e.g. a design activity. The assumption will be approved by those parties that will be affected by the assumption. The assumptions than will get a status e.g. 'approved' and there will be another work package activity that will have the assumption as input. When the assumption is successfully processed the status of the activity changes e.g. to 'executed'. In that case in principle there must be a document that proves the execution of the assumption. During the workflow statements can be made by the various parties to explain the course of the assumption. Assumption can e.g. be used to clarify requirement specification as output of a requirement analysis activity (a specific work package activity) and to describe a system design of human activity.

Despite of the fact that the information models of both a measurement and assumption looks like the same, they represent each another concept, meaning and role within projects. For this reason both were not combined in one model.

An important entity within SE is a decision. In the presented models decisions are represented by design principles (figure 6), measures (figure 15) and assumptions (figure 16): they all in essence represent decisions. They all have a relation with work package activities and therefore with milestones, so major decisions can be defined by introducing specific milestones.

Trade-off studies including the final design principles can be described in documents that are related to the relevant work packages and Technical Solutions.

4. Implementation technology

Due to lack of software tools that could handle the presented approach in a distributed environment including exchange of product information, within the shipbuilding industry in the Netherlands, the presented SE approach is implemented by using RDF technology in a WEB environment. Resource Description Framework (RDF) [9] is a formal language that makes it possible to describe the semantics of information and is very similar to the mechanism described in the previous sections. RDF is a general-purpose language for representing information on the Web. It defines a language for describing relationships ("predicates") among resources in terms of named properties and values. RDF provides no mechanisms for describing properties, nor does it provide any mechanisms for describing the relationships between properties and other resources. That is the role of the RDF vocabulary description language, RDF Schema (RDFS). RDFS defines classes and properties that may be used to describe classes, properties and other resources. These resources are used to determine characteristics of other resources, such as the domains and ranges of properties. RDFS vocabulary descriptions are written in RDF. The abstract model of RDF comes down to four simple rules:

1. A fact is expressed as a Subject-Predicate-Object triple, also known as a statement. It is like a basic, explicit sentence in English.
2. Subjects, predicates and objects are given as names for entities, also called resources (dating back to RDF's application to metadata for web resources) or nodes (from graph terminology). Entities represent something – a person, website or something more abstract such as states and relations.
3. Names are URIs, which are global in scope, always referring to the same entity in any RDF document in which they appear.
4. Objects can also be given as text values, called literal values, which may or may not be typed using XML Schema data types.

The simplicity and flexibility of the triple, in combination with the use of URI's for globally unique names, makes RDF unique and very powerful. It is a specification that fills a very particular niche for decentralized, distributed knowledge and provides a framework to enable computer applications to answer questions we would not consider asking computers today. The underlying structure of any expression in RDF is a collection of triples, each consisting of a subject, a predicate and an object. A set of such triples is called an RDF graph. An example is given in figure 17, a triple being described as a node-arc-node link. An RDF triple is conventionally written in the order subject, predicate, object. The direction of the arc is significant: it always points towards the object. The assertion of an RDF triple says that some relationship, indicated by the predicate, holds between the things denoted by subject and object of the triple. The assertion of an RDF graph amounts to asserting all the triples in it, so the meaning of an RDF graph is the conjunction of the statements corresponding to all the triples it contains (see figure 17).

De formal W3C standard of RDF [9] only knows a few predicates. In this paper a set of relationships is presented that can be used to expand these few original ones by defining the relationships as a subclass of rdf:property. By making use of the TRIX protocol defined by W3C based on the usages of Named Graphs, one can implement the presented models in a WEB environment making use of software developed by the internet community resulting in a flexible and powerful information integration and exchange platform for SE projects. TRIX also support provenance and digital signing of a Named Graph containing a set of triples ('facts').

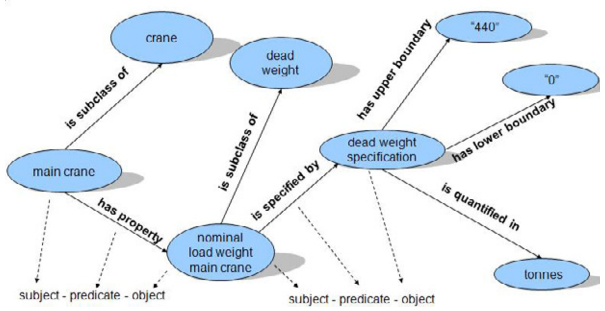
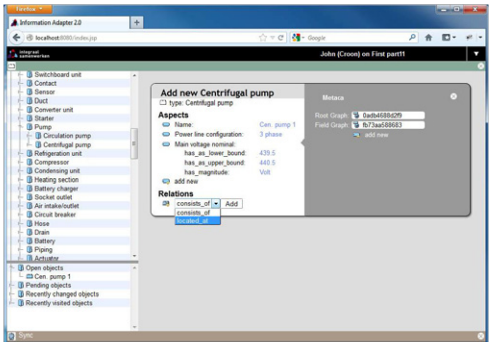


Fig. 17. Example of RDF graph representation of set of triples describing a property of a physical object.



Add new Centrifugal pump
type: Centrifugal pump

Aspects

- Name: Cen. pump 1
- Power line configuration: 3 phase
- Main voltage nominal:
  - has_as_lower_bound: 439.5
  - has_as_upper_bound: 440.5
  - has_magnitude: Volt
- add new

Relations

consists_of Add
consists_of
located_at

Metaca
Root Graph: 0adb4688d2f9
Field Graph: fb73aa588683
add new

Fig. 18. Screen dump of the WEB application based on the presented ontology, build for a pilot project within the shipbuilding industry in the Netherlands.

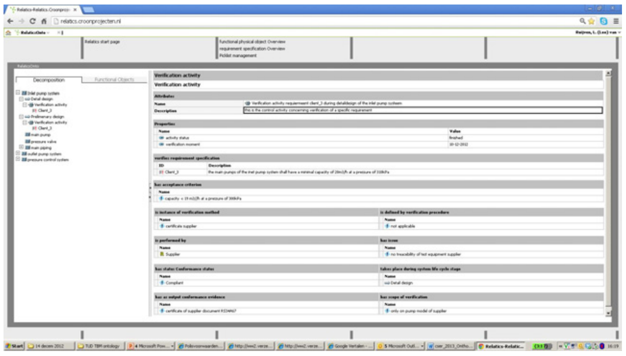
5. Conclusion

The main problem within projects delivering a system is that even though the used technology itself is mostly not very complicated, the process of engineering and realization is complex. This is caused by the number of parties involved, the many different interpretations of SE resulting in lack of interoperability, the fragmentation of the total system life cycle with related contacts and information handovers, and by the wide variety of types of system objects to handle. One major step to improve the project environment would be the availability of a common ontology representing SE defined by standardized information models as presented in this paper to support the SE processes. Using this ontology, project organizations would be much better able to set up an adequate project management system. This would prevent verbal chaos between the various parties involved in such a project and will improve interoperability.

The ontology presented in this paper, based on and a simplification of ISO 15926 intents to be a practical approach of model-based Systems Engineering. Based on several experiences implementing an ontology as described in this paper requires executive management support as well as project leaders and engineers with adequate skills and competences to handle abstract and subjective matters that come with working with a taxonomy and ontology.

In a market research there proved to be a very few ‘off the shelf’ information management systems available that can handle the semantically richness of the total of all the presented information models including the flexibility to add new relationships when needed. In the Netherlands an implementation of the presented ontology in several infrastructure projects requiring Systems Engineering is done by using the information management tool Relatics

RDF Named Graphs has shown a promising technology to implement ontologies by expanding RDF with a set of relationships and because of the ability to add meta data to ‘facts’. An implementation based on RDF has been validated in a real shipbuilding project. A next step could be to construct a well-formed ontology in OWL.


relatics.croonprojecten.nl

Relatics start page | functional physical object Overview | requirement specification Overview | Picklist management

Decomposition | Functional Objects

- Inlet pump system
  - Detail design
    - Verification activity
      - Client_3
  - Preliminary design
    - Verification activity
      - Client_3
  - main pump
  - pressure valve
  - main piping
- outlet pump system
- pressure control system

Verification activity

Verification activity

Attributes

Name	Verification activity requirement client_3 during detaildesign of the inlet pump system
Description	this is the control activity concerning verification of a specific requirement

Properties

Name	Value
activity status	finished
verification moment	10-12-2012

verifies requirement specification

ID	Description
Client_3	the main pumps of the inlet pump system shall have a minimal capacity of 20m3/h at a pressure of 310kPa

has acceptance criterion

Name
capacity < 19 m3/h at a pressure of 300kPa

is instance of verification method	is defined by verification procedure
Name	Name
certificate supplier	not applicable

is performed by	has issue
Name	Name
Supplier	no traceability of test equipment supplier

has status Conformance status	takes place during system life cycle stage
Name	Name
Compliant	Detail design

has as output conformance evidence	has scope of verification
Name	Name
certificate of supplier document RS34A67	only on pump model of supplier

Fig. 19. Sceendump of the tool Relatics that is used to operationalize the presented ontology within infrastructure projects in the Netherlands. Specifically is shown the way the verification model is implemented by presenting the System Breakdown Structure (left window) and for each system element one or more applicable lifecycle stage are introduces as a placeholder for verification activities, each handling the verification of a requirement (right window).

References

1. CEN ORCHID workshop; www.cen.eu/go/ORCHID, 2010.

2. ISO 15288; System Life Cycle Processes, ISO, Geneva. 2008.

3. ISO 15926; Integration of lifecycle data for process plants including oil and gas production facilities, ISO, Geneva. 2010.

4. RENSSEN, A. VAN; Gellish, A generic extensible ontology language, Delft, 2005.

5. NIST Cost Analysis of Inadequate Interoperability in the U.S. Capital Facilities Industry, U.S. department of Commerce, Galthersburg, Maryland, 2002.

6. RUIJVEN L.C. VAN; U. NIENHUIS; Requirement management, traditional and a second generation scenario, COMPIT, 2006.

7. WITTGENSTEIN, LUDWIG; Philosophische Untersuchungen, Cambridge, 1953.

8. GIELINGH W.; SUHM A.; IMPACT Reference Model for Integrated Product and Process Modeling, Delft, 1993

9. W3C; Resource Description Framework (RDF); http://www.w3.org/RDF/, 2004.

10.RUIJVEN LC VAN; Ontology for model based systems Engineering; Workshop Formal Ontology Meets Industry, Faculty of Technology, Policy and Management, Delft University of Technology, 2011.

11. RUIJVEN L.C. VAN; Ontology for model based systems Engineering; Workshop Formal Ontology Meets Industry, Faculty of Technology, Policy and Management, Delft University of Technology, 2011.

12 . IN HET VELD J.; Analyse van organisatieproblemen, Noordhoff uitgevers, Groningen, 2007.