A Framework to Support Requirements Analysis in Engineering Design

William Brace, Vincent Cheutet

A Framework to Support Requirements Analysis in Engineering Design

Complex system development activities such as requirements analysis to requirements specification, implementation and verification are well defined in the software engineering domain. Interests in using a model driven engineering have increased in this domain. System level requirements analysis and model driven engineering may result in a significant improvement in engineering design. This paper presents a Checklist Oriented Requirement Analysis (CORA) framework to develop and formalize requirements. CORA is an integrated framework that adopts a checklist concept and utilizes logical reasoning operation in conjunction with information management to analyze systematically the initial requirements statement. An underground work machine is used as an application example to illustrate the proposed framework.

Keywords: requirements analysis framework, requirements checklist, model-based requirements, requirements information, CORA-framework

1. Introduction

The engineering design process starts with a design problem expressed as a need (i.e. customer or initial requirements) that must be satisfied by the creation of a physical product or system. These needs provide the foundation for engineering design efforts but do not necessarily provide all the knowledge required for the subsequent design process and should thus be analyzed (Weigers 2003). As the design episode proceeds, informally expressed customer requirements are explored, developed and formulated to become abstract, unambiguous, traceable, and validatable, i.e. well-formed. Poorly analyzed customer requirements have been variously cited (e.g. Brooks, 1987, Hall *et al*., 2002) as leading to poor or inappropriate products, the inability to perform the required function, not to mention, leading to failure, unreliability, and by no means least, extra cost for a company.

The level of complexity of customer requirements has also increased due to increase in product complexity and distributed development. It is essential to tackle the complexity in a more cost-effective way to meet customer and environment needs and wishes (Pisano and Wheelwright 1995, Drejer 2008). Successful product

developments require multi-disciplinary approaches, which necessitate the integration of various engineers and specialists (Gupta *et al*. 2007). Therefore, a new form of current requirements analysis (RA) is an integration of all the disciplines and specialist groups into a team effort. However, there are difficulties associated with this approach. The stakeholders (i.e. customers, marketers and designers) involved, employ different sets of context to express the requirements. Differences in semantics and terminology impair the ability to communicate requirements, which have an adverse effect on collaboration (Greer *et al*. 2003).

At the same time, access to relevant information (i.e. the information required for RA) is required at an early stage for decision making during requirements analysis. Knowledge and information are central to decision making at all stages of product development. Moreover, the type of information used changes during the designing process (Lowe *et al*. 2004b). Various knowledge management (KM) systems exist and are used during the product development process to store and retrieve information. However, these existing methods are generally not compatible with the whole product design process as most are focused on detail design (Baxter *et al*. 2007). Despite the existence of KM systems, the designer wishes to be taken gently through the huge "minefield of information" to find suitable ones at the right time. Thus, managing and increasing accessibility to relevant information will be beneficial for the requirements engineer.

Interestingly, the commercial demands on software development have motivated a considerable amount of research into requirements development. Therefore, RA activities, implementation and verification are well defined in the software engineering domain with extensive computational application (van Lamsweerde 2000, Sommerville 2001, Parvianen *et al*. 2003, Grady 2006). In the

domain of mechanical engineering (hereafter engineering design), requirements analysis encompasses those tasks that go into determining the conditions to be met for a new or altered product, also taking into account possible conflicts (Pahl and Beitz 2007). Generally, most complex product development activities have engineering design at its forefront, and poorly developed requirements may impose constraints and narrow the solution space for other disciplines. Several procedures for analyzing design problems and creating requirements are suggested in literatures, e.g. (Pugh 1997, Ulrich and Eppinger 2004, Pahl and Beitz 2007). However, a brief scrutiny of current research, influential textbooks (e.g. Nuseibeh and Easterbrook 2000, Grady 2006, McAlpine *et al*. 2010), and empirical studies in various design projects indicate that these procedures have problems, which imply for us the following research motivations:

- Motivation 1: there is the lack of a formal process and a formalized collaborative work with experts as the concept of requirements analysis is not precisely and uniformly defined using IT tools as they are in the software engineering domain.
- Motivation 2: proposed approaches are document-centric and very often, labour intensive.
- Motivation 3: knowledge in mechanical engineering design is still more empirical (experiential) and not all aspects of requirements are consciously considered during analysis in existing methods. Since there are difficulties in re-use of available information due to lack of well defined and easily accessible information.

A necessary precursor to alleviate these problems is to create a framework for requirements analysis. The approach is to exploit requirements formalism in both the

engineering design and software engineering domain to create a framework to allow modelling and analysis with computer systems. To create this framework, we analyze both the scientific literature on the role and the use of requirements in a design project, and our design experience, mainly expressed in this study by our implication in a research design project (cf. section 4.1).

The remainder of this paper is structured as follows: existing concepts of requirements analysis and information access are given in sections 2 and 3. The research methodology is then discussed in section 4. This is followed by the proposed requirements analysis framework in section 5. To demonstrate the applicability of the framework, a case study example is presented in section 6. Section 7 concludes the research.

2. Existing concepts of requirements analysis

2.1. General principles in requirements analysis

According to Grady (2006), a requirement is an essential *attribute* or characteristics for a system. The attribute is coupled with *values* and *units* information by a *relation* statement. The following is an example: “weight is greater than or equal to 17965 kg” (see Figure 1). These are primitive requirement statements, often constructed through requirements analysis. Requirements are discriminated to functional requirements (FRs) and non-functional requirements (NFRs) or constraints.

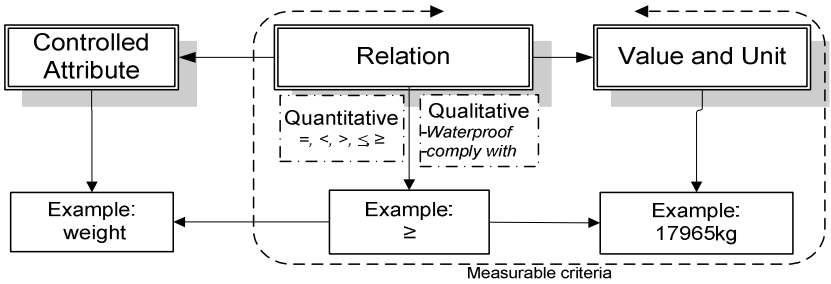

Figure 1. Primitive requirements statement structure with example (adapted from Grady 2006)

The process of requirements analysis is the transformation of an input to a desired output and communicating. The input is a combination of *informal* expression of the needs and information from several sources. The desired output is a *requirement specification* (i.e. requirements list), from which a design solution can be generated. This is a complex process, as the transformation can relate to any *aspect* and *source*. It can reflect the point of view of any person or multiple persons. It is an open source of information, which can be transformed for use in a variety of ways (Darlington and Culley 2002).

Several tools and activities have been proposed for this transformation. One such activity is *requirements engineering* (RE) (Sommerville 2001). Since the term, engineering was attached to requirements by Alford (1977 cited Jiang 2005), RE efforts have endeavoured to incorporate engineering approach to what was traditionally known as system analysis. A *systematic requirements analysis* is also known as requirements engineering. The RE-activity is divided into requirements development (RD) and requirements management (RM) which are the control of the whole requirements process. RD is composed of elicitation, analysis, documentation, verification and validation (McConnell 1996, Gilb 1997).

RE offers a number of techniques for evolving requirements. There are four general principles of RE techniques, which are of interest (Kotonya and Sommerville 1998):

- *Abstraction* involves ignoring the details and retaining relevant information for a particular purpose. For instance, when two different actions are taken and described as instances of the same general action, we are using abstraction.

- *Decomposition* involves breaking a problem into manageable parts to be analyzed independently. Such decomposition is not perfect because of tight coupling between parts, but they give insight into how things work.
- *Projection* deals with the adoption of a particular view or perspective and describes aspects important to that view.
- *Modularity* is finding structures that are stable over time and across different contexts.

The systematic use of decomposition, abstraction and projection allows complexity to be dealt with by making problems simpler. They are used by a requirements' engineer in a way to understand problem situations and to identify parts of the problem that can be structured. The idea of knowledge re-use makes modularity, which is important in design, equally vital for requirements analysis. It allows us to handle evolution of the requirements over time. Moreover, existing solutions and knowledge can be exploited when considering any new problem. These general principles are feasible and can be employed to resolve the complexity associated with requirements analysis. In the following two sub-sections, we look at how these general principles are employed in requirements analysis in different domains.

2.2. Requirements analysis in the engineering design domain

The process of requirement development in engineering design is in two phases. Typical working steps required for a requirements development process is as shown in Figure 2.

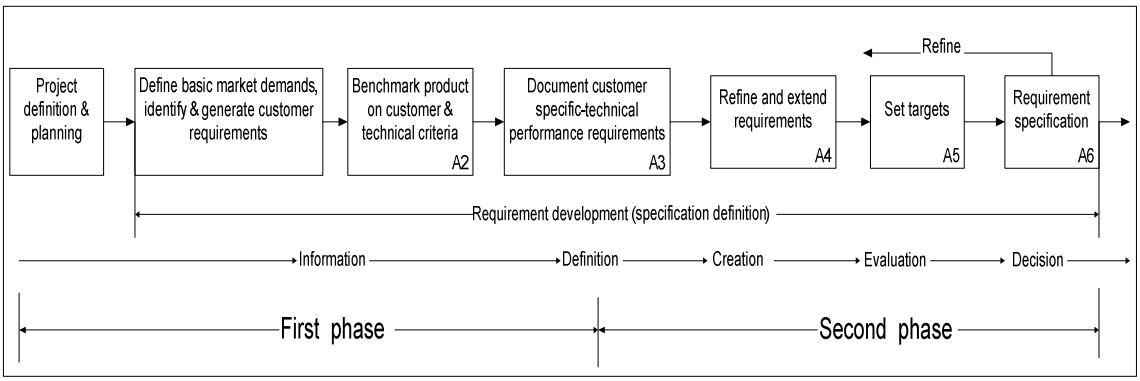

Figure 2. Main working steps for requirements development

In the first phase, information is gathered, and obvious requirements defined and recorded. Secondly, the recorded requirements are *refined* and *extended* using the information and special methods to generate requirement specification (Cooper *et al*. 1998, Otto and Wood 2002, Ulrich and Eppinger 2004, Pahl and Beitz 2007). This research work is concerned with the second phase. Extension in this context is the complete analysis to find elusive requirements. Refinement is the process of making the requirements less abstract and quantifying wherever possible (Ullman 2002). The requirement specification is a measurable behaviour of the system-to-be that will help, later in the design process, in determining its quality. Therefore, in order to measure the "quality" in the refinement process, requirements are presented to include criteria with specific qualitative and quantitative forms. Stored information provides the basis for types of criteria and values.

Several methods have been proposed for refinement and extension of requirements. Matrix based methods have been used extensively (Darlington and Culley 2002, Ullman 2002, Baumberger and Lindermann 2006, Short *et al*. 2009). Quality function deployment (QFD), a matrix base approach is used mostly in this domain (Akao 2004). Recently, there are works using QFD and modelling techniques to represent a relationship between customer requirements and design attributes of new products, e.g. (Amihud *et al.*, 2007, Chan *et al*. 2010). Nevertheless, the QFD

method is best for collecting and refining functional requirements hence the “F” in its name. According to Grady (Grady 2006), QFD, model simply establishes a relationship between information pairs and does not provide graphically expressed devices in models that encourage thoughts about specific aspects about the design problem. The author’s conclusion is that it is a requirements listing tool linked to a design implementation tool and most useful for incremental improvement situations in product design.

An alternative approach is the use of decomposition based on checklist, a generic list of major aspects and sources of requirements. Pugh (1997) considered thirty two checklists called “elements of product specification” to create an evolutionary document matching the characteristics of the final design as it develops.

Ullman (2002) used eight major types of checklists with sub-categories. The purpose was to reveal missing or elusive requirements, to develop questionnaires to ask in a survey (i.e. *elicitation plan*) and as a *key to information* that needs to be found before design begins.

Otto and Wood (2001) advocated the application of specification list generation that uses decomposition based on checklists developed by Franke (1975 cited by Otto and Wood 2001). However, this checklist considers mostly the technical and user aspects of a system and is used as guidelines.

Similar, checklists were used by Pahl and Beitz (2007) in their systematic approach to refining and extending requirements. In addition to the checklist method, they proposed the use of scenario creation. The checklists were in two forms. Seventeen major checklists with examples were advocated as guidelines for setting up a requirements list and fourteen checklists for implementing requirements in the embodiment design.

Dieter (2000) proposed checklists with four major lists of elements with several sub-categories that are to be found in a product design specification.

Sudin and others (2010) in their investigation on the *sources* and *aspects* leading to identification of requirements found twelve sources and seventeen aspects used by engineers as checklists.

Ward and colleagues (2003) used requirements checklist presented as a matrix to trace the life cycle against particular areas to identify requirements not captured by functional analysis. Table 1 shows an excerpt of types of checklists from several sources.

Table 1. An excerpt of checklist from several sources

Elements of product specification *(Dieter 2000)*	**Checklist** for requirements list *(Pahl and Beitz 2007)*	**Categories** for Spec. based on Franke, (1975) *(Otto and Wood 2001)*	**Types** of customer requirements *(Ullman 2002)*	**Elements** of product specification *(Pugh 1997)*	**Sources** SE requirements *(Bahill and Dean 1997)*
Functional performance -flow of energy -flow of information	Energy signal material operation	material signal operation	**Functional performance** -flow of energy -flow of information	Performance processes material	Input – Output performance
life cycle issues -useful life, reliability -maintainability -	- quality control -maintenance - recycling	- quality control - maintenance	**life cycle concerns** -distribution (shipping) -	- disposal - maintenance - quality reliability -	-system test - reliability
Human factors -aesthetics -ergonomics -user training	-ergonomics	-ergonomics	**Human factors** -appearance -force & motion control -	- customer - ergonomics -aesthetics	- intangibles (aesthetics, prestige) - common sense

In summary, the checklist is useful in providing a decomposition strategy valuable for analysis, documentation and computer application. However, current approaches (described above) do not enhance this strategy. An intelligent system could improve the effectiveness of this process. The checklist approach is seldom used in industries to structure requirements for analysis (Ullman 2002, Grady 2006, Pahl and Beitz 2007). Very often and in existing methodologies, the requirements are

set up based on subsystems (functions or assemblies). For instance, in the automobile industry, the requirements are set on sub-divisions into engine, transmission and bodywork development. Furthermore, many discussions of the requirements analysis process appeal only into functional flow analysis as the one approach and means to gain insight to needed system function and to extract requirements. However, this approach is not necessarily the single fruitful one (Grady 2006). It may prove useful to set up the requirements based on checklist on all levels of the system, and we exploit this potential in our research work.

2.3. Requirements analysis in other domains

Requirements Engineering (RE) is placed firmly in the requirement development of software-intensive systems. Therefore, it is regarded as a sub-discipline for software engineering (Pohl 1994, Zave and Jackson 1997, Parvianen *et al*. 2003). It is also considered as a branch of systems engineering (SE) concerned with the real world goals functions and constraints with software intensive systems (Bahill and Dean 1996, INCOSE 1998). In these domains, much emphasis is placed on functional requirements despite the increased concerns for inclusion of non-functional requirements (NFRs). This has resulted in the development of tools and methodologies to support the process of generating these requirements.

A literature review indicates that modelling is a fundamental activity in RE (Davis 1990, Luocopoulos and Kavakli 1995, Douglass 1999, Nuseibeh and Easterbrook 2000). Modelling is the construct of abstract descriptions at an appropriate level of detail that is clear, unambiguous and easy to understand by all stakeholders. RE methods, (e.g. Entity Relationship approaches, Structured Analysis approaches, Object-oriented approaches) focus on a particular modelling perspective of which the three main ones are information, function and behaviour. Since within

these three modelling perspectives a system's requirements are stated, which partially overlap, the perspectives must be somehow integrated to relate. Recently proposed object-oriented approaches (e.g. Unified modelling language-UML) provide some form of integration and abstraction mechanisms.

SysML, a flexible System Modelling Language has been customized from UML for SE application (Blanchard and Fabrycky 2006, OMG SysML, 2008). The SysML taxonomy includes standardized diagrams to support requirements, analysis, design, verification, and validation of a broad range of complex systems (Soares and Vransken 2008). SysML provides allocations to generate relationships between the various model elements and several aspects of the requirements such as traceability, verification, and validation has been taken into account (Follmer *et al*. 2010). Figure 3 shows the standardized diagram types recognized in SysML (Peak *et al.* 2007, Friedenthal *et al.* 2008, Weilkiens 2008).

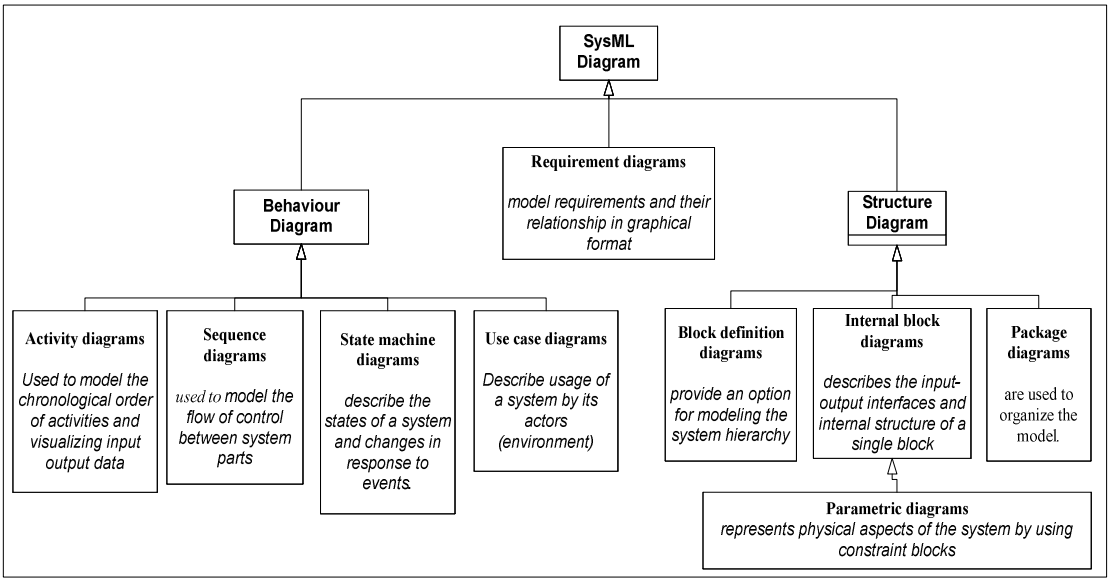

Figure 3. Diagram of SysML taxonomy (adapted from OMG 2008)

SE which has a requirements analysis analysis at its fore-front is very useful in handling and managing requirements by structuring and linking to the product. The hierarchical structures used are functions, systems or parts, and requirements are

allocated to these structures to enhance traceability (Svensson and Malmqvist 2001). SysML has greater flexibility and the standardized diagrams can be customization, as exploited to model requirements analysis in our research project.

3. Concept of Information access

The various design phases are a continuous process of transformation from one information state to another as a consequence of a *decision process*, driven by knowledge and available information (Hicks *et al.* 2002). Information access is one of the foundations of requirement development. The use of information for requirements does not cease, but continues through the other design phases. Engineers want to be pushed relevant information at the right time. Lowe and others (2004a) revealed that engineers spend 35% of their time searching for and interpreting information during design activities. Engineers then tend to draw about 40% of information from their own document store because of the difficulties in accessing official records.

Various information database systems are used in the product development process. A product data management (PDM) system handles much of the information created in the design phase. The system traditionally aims at managing part structure and product documentation. Others include enterprise resource planning (ERP) systems, which has the functionality needed to handle information for manufacturing products. PDM and ERP systems are used in different phases of the development process but requirements are used and propagated throughout the.

In current requirement development methods, product document searches are very often ignored. It is assumed that the requirements engineer has the knowledge (i.e. experience). Knowledge re-use is one approach to improve engineering design and remains a developing effort (Baxter et al. 2007). It is also the assumption that information from product source is needed more in subsequent design phases.

According to (McAlpine *et al.* 2010), concerns have as well been raised that it was impossible to track back through a project to the information source to understand why a particular decision was taken. In addition, a number of engineers are not able to identify other engineers within a company to collaborate with, in a project or to know the reason for a decision (Loftus *et al.* 2009).

In a typical industry, several information systems are used, in which function and part structures are forms of information storage (Svensson and Malmqvist 2001). However, in a complex system design, information related through functions and parts are not adequate. A new form of classification to include all relevant information should be adopted. Since data structure in PDM is from product-level, it provides less support for requirements analysis, which is on an abstraction level. Data from requirements can be referred to as *master data,* which is data used throughout the product life cycle. Well *structured requirements* can be an important basis for information system management and integration. Structured requirements stored in such systems will provide a valuable source of knowledge about previous projects and act as an information *search portal*. It can also be structured to point engineers to key individuals involved in a project. Since a requirements document travels throughout the entire product life cycle process and is available to all stakeholders, it is a valuable source for communication.

4. Research methodology

The research is a case study based on the University project [name deleted to maintain the integrity of the review process]. The goal of the project is to develop an existing specialist and relatively complex product. The case study method involves an in-depth, longitudinal examination of the project case. This provides a systematic way of looking at events, collecting data, analyzing information, and reporting the results

(Flyvbjerg, 2006). We gained as a result, a sharpened understanding of the requirements analysis process as it happened, and what might become important to look at more extensively in future research.

A case study should use as many sources as relevant to the study to enhance the validity and reliability (Yin, 1994). Therefore, the sources used includes, participation and observation in the research project and review of prescriptive and descriptive literature (described above). The project involves different groups of engineers and other stakeholders. Consequently, the use of participation aims to gain a close familiarity with the practices through an intensive involvement with the engineers in their natural environment over an extended period of time. The strategy involves a range of methods: informal interviews, collective discussions and analysis of personal documents. The main concern with this method is the potential bias as an active participant. However, the method allows for direct observation studies on-site to collect data. Nevertheless, the method has the drawback of selectivity, which might miss some facts.

The strength of the observation and interaction methods is the discovery of discrepancies not disclosed with a survey of only interview answers. The literature review done iteratively and simultaneously is also used to corroborate evidence gathered from these sources. The rationale of using multiple sources (participation, observation, literature review) is the triangulation of evidence. Triangulation refers to the use of several approaches to the investigation of a research question to enhance confidence in the ensuing findings (Stake, 1995). In the context of data collection, it serves to substantiate the data gathered from other sources.

4.1. Criteria for measuring the success of the method

During the investigating phase, a general research aim was formed. The objective was to use the following set of criteria (Table 2) found from the literature review to assess the current requirements analysis process as well as to observe the success of the framework after implementation.

Table 2. Set of criteria for the evaluation of current approach

Evaluation Criteria	
C1	Method and computer-based tools used
C2	Level of awareness and involvement
C3	Knowledge management and information access
C4	Quality of generated requirements
C5	Cost effectiveness of the applied RA process

Based on the findings of the research, more measurable objectives (i.e., criteria) were developed to help gauge the success of the method:

- First objective is *quality of output*: A successful method will generate requirements, which are well-formed (i.e., unambiguous, traceable and validatable) and solution independent (i.e., not specifying a solution to the problem.
- Second objective is *quality of the process*: the framework should have other characteristics such as usability, effectiveness and efficiency (i.e., reduced process time and cost).

The measurable objectives were defined to evaluate the framework after real implementation in industry. Therefore, an attempt was made to identify a number of key issues (Table 3) to evaluate and improve the method prior to implementation.

Table 3. Excerpt of key issues extracted to improve the requirements analysis process

key Issues	
K1	Efficient handling of complexities in requirements analysis process
K2	Stimulate a multidisciplinary approach to requirements analysis
K3	Encourage responsibility of each team member
K4	Prevent or discourage the fact that all aspects and sources of requirements will not be considered
K5	Encourage information retrieval and knowledge reuse
K6	Traceability in the requirements analysis process
K7	The ability to resolve inconsistencies in requirements
K8	Measuring the requirement analysis process for quantitative analysis
K9	Relevance, coherency, consistency and connectivity in the requirements model
K11	Balance between functional requirements and non-functional requirements

K12	Capability to trap requirements related defects early in the design phase
K13	Increasing maturity of stakeholders by requirements analysis activities

4.2. Problems observed in project requirements analysis

There was a critical elaboration of the method for developing requirements based on the criteria set (Table 2) and the research motivation as outlined in section 1. We applied the grounded theory methodology to analyze the data collected (Strauss & Corbin, 1998). The theory includes three different types of coding procedures: open, axial and selective coding. The coding was applied to identify and analyze problems associated with the requirement analysis process in the domain. During the open coding phase, we identified the following.

The initial techniques, which were manual recording and use of word processors, were conducted in isolation. There was no standard method, and the activities were ad hoc. The requirements documentation and specification were not sufficient and not well-structured. Owing to its inadequacy, the team decided to employ additional techniques such as a computer-based modelling tool (i.e., SysML). However, the tool was poor in aiding the requirements analysis process. With regards to the level of awareness and involvement, due to lack of collaboration and traceability, few people were sure of what the system requirements were. Team work and collaboration were reduced to creative group meetings and activation of experiences. The responsibility of generating requirements was assigned to one person who found it difficult to find experts to collaborate and negotiate with.

Different groups of people in the project are responsible for collecting information, product benchmarking and determining customer requirements. However, there was intermittent information flow between the research team members. Therefore, the requirements analysis was initially conducted with a severe lack of information and stakeholder awareness. The quality of generated requirements

was poor as the links between the process model, information, requirements and subsequent design phase could not be established and clarified. The current requirements analysis approach failed in respect to the quality of cost-benefit analysis because it was not strategically taken into account. The approach used was not cost effective. The various problem concepts are grouped into categories. According to Strauss and Corbin (Strauss & Corbin, 1998), grouping categories into categories is important as it enables the analyst to reduce the number of units to work with.

In the axial coding phase, we identified the relationship between the categories. This is also important as according to Strauss and Corbin (Strauss & Corbin, 1998), discovering the ways that categories relate to each other helps an analyst to contextualize the phenomenon under study.

During the selective coding phase, we discovered a central category based on the need for techniques contributing to good requirement practices and project success according to Macauley (Macaulay, 1996). The themes used are, process, human involvement, knowledge development, documentation and management. Strauss and Corbin (Strauss & Corbin, 1998) emphasize the identification of the central category as this has the analytical power to pull the other categories together to form an explanatory whole. Therefore, the problem concepts identified are presented as:

(1) The requirements analysis process itself,

(2) Human communication and collaboration within the requirements analysis process,

(3) Knowledge development and information awareness,

(4) Structure and documentation of requirements,

(5) Use of appropriate computer based modelling tool.

To overcome these causes, a framework for effective and efficient requirements analysis is proposed in section 5 and described in details.

5. The requirements analysis framework

From the foregone discussions, it is obvious that requirement development is a recurrent activity which involves multiple levels across many teams. Therefore, the document-driven process in the engineering design domain is examined more closely from a system perspective. Two alternative paradigms (i.e., hard and soft system thinking) exist for systems thinking (Checkland 1981). Hard Systems Thinking refers to SE methods. This method relies mostly on *technical view* and computer application to handle requirements more efficiently. Soft System Thinking seeks to incorporate multiple stakeholder views and other situations perceived as problems in problem analysis (Moores and Gregory 2000). These paradigms are applied to the traditional requirements analysis approach and form the underlining principle for the proposed approach.

The novelty of our approach, therefore, resides in placing the document-driven requirements analysis process into a systematic and structured format. This is done by integrating several known and useful methods such as checklist decomposition, graphic modelling and data structuring with adequate format based on systems thinking to allow for computer application.

5.1. The scenario

To convey a sense of the utilization of the approach to be proposed, and to act as an example to anchor the details which will be presented in the later sections, we show a scenario illustrating the use of the requirements analysis framework. Initial requirements may be presented to the requirement engineer in one of the following forms:

- one sentence or in the form of a short narrative statement
- or, very detailed with a profusion of texts, graphs, or even formulas.

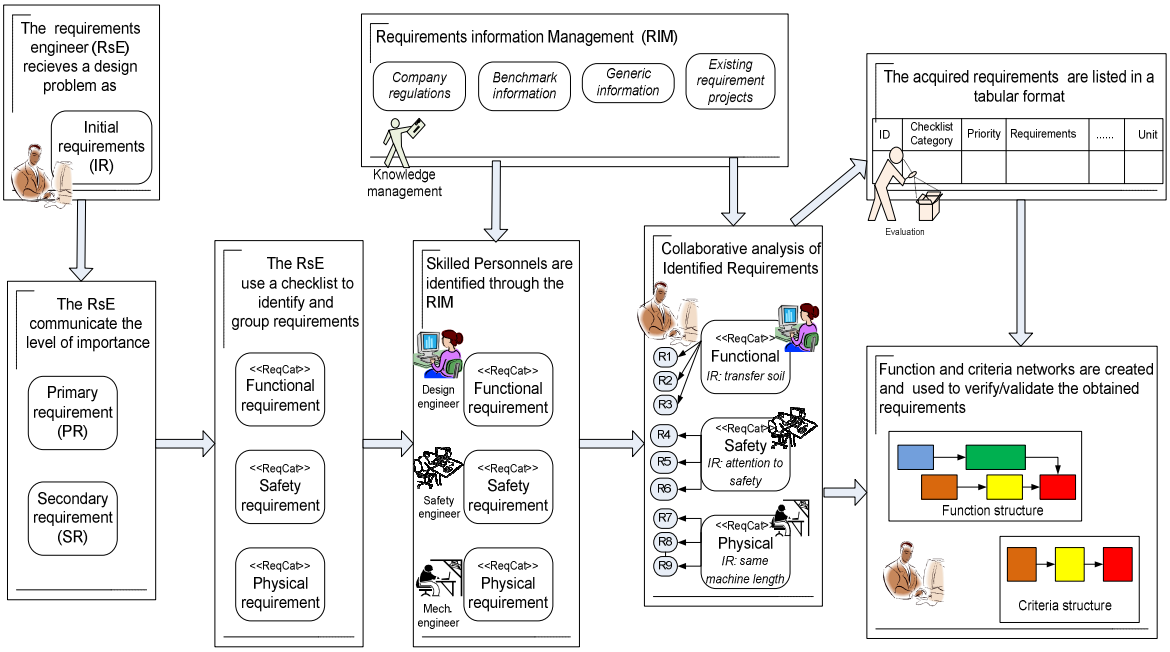

Figure 4. A roadmap of requirements development

The abstract nature of our scenario (Figure 4) is based on the assumption that the design is taking place in a multi-disciplinary environment. The requirement engineer receives the initial requirement in the form of a short narrative statement. The statement needs to be reviewed in order to identify omissions and inaccuracies. Therefore, the requirement engineer together with others prioritizes and decomposes the narrative statement, refine, extend, create dependencies and validate the identified requirements.

The key to this roadmap is the implementation of a *Checklist Oriented Requirements Analysis* (CORA) framework. The basic building block of the CORA-framework is the unified checklist (see excerpt from Table 1). A checklist can be thought of as a “reminder” and an “organizer” in the requirement development process. Checklist is the simplest kind of rational design method which unlike creative methods encourages a *systematic approach* to design (Cross 2008). In the sections

that follow, we will describe the basic units of the CORA-framework as well as an example to demonstrate the interactions in the scenario and validate and evaluate the framework.

5.2. An overview of the CORA-framework

The basic units of CORA are the unified checklist, requirements information management, specification drivers, functional and non-functional analysis, and criteria analysis (Figure 5).

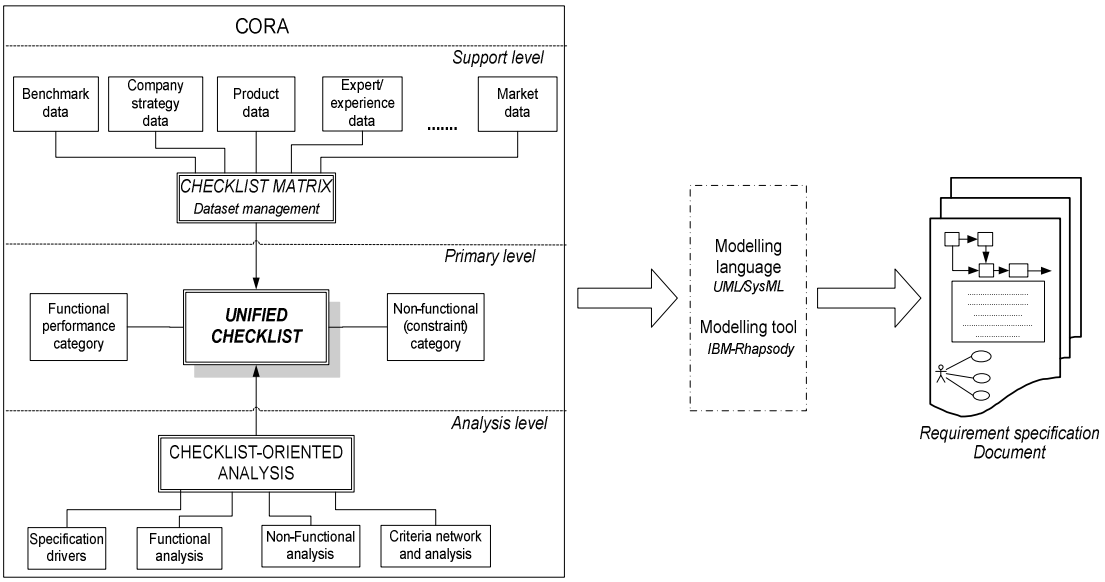

Figure 5. The CORA/SE framework

These units are different in their approach, and they help together to ensure that a good requirement specification is drafted. To make the approach model-centric, the CORA framework is formalized with SysML (Figure 3). The basic units are briefly described in the following sub-sections.

5.2.1. Unified checklist

The unified checklist aims at managing requirements by creating one kind of hierarchical structure to represent different requirement views (Figure 6).

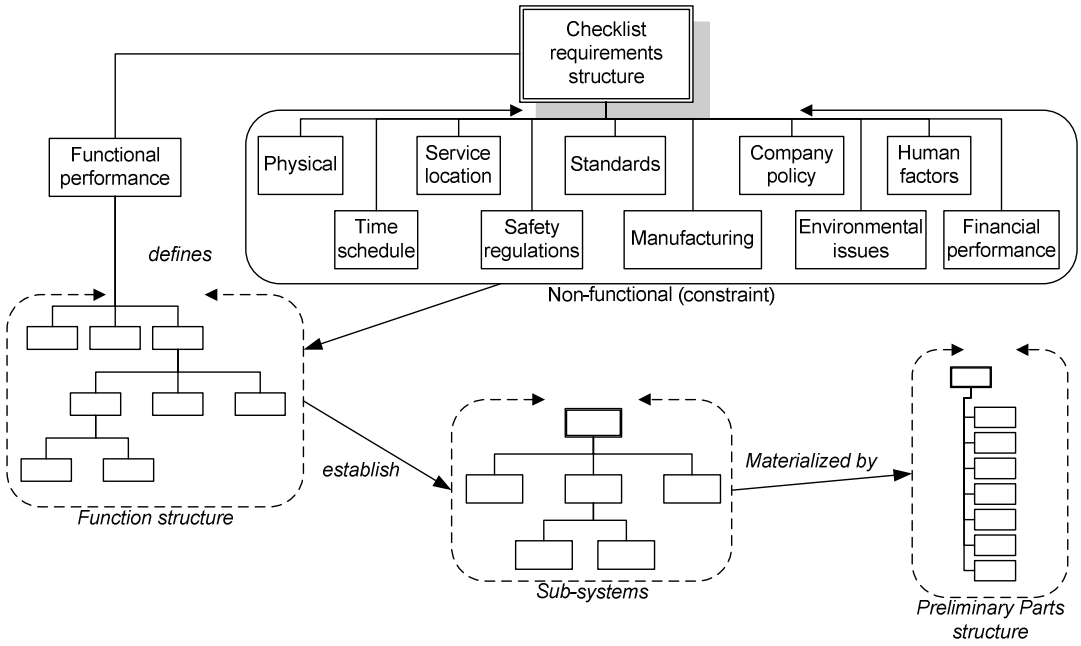

Figure 6. Checklist requirements structure model

The checklist is intended to support a structured analysis process as well as to manage data. Structure analysis is an organized method requiring a broad appeal to knowledge, for partitioning a complex problem into smaller problems better matched to human and team proportion to solve (Grady 2006). The checklist model is based on four structures (see Figure 6). On top is the checklist requirement structure which discriminates the requirements into functions and non-functions with various categories. Functions are logically decomposed into smaller entities and interactions with non-functional categories established. A function structure is composed to encourage a clearer formulation of the requirements. The function structure is also used to decompose the complex product to set up sub-systems (i.e. assemblies) to induce more requirements based on the checklist. The sub-system is materialized by the preliminary parts structure which helps in linking requirements to design solutions in subsequent design phase. One important aspect during the requirements analysis process is verifying that the requirements are satisfied in every phase. In SysML, the unified checklist is modelled as stereotypes (i.e. user defined notations) (Figure 7.

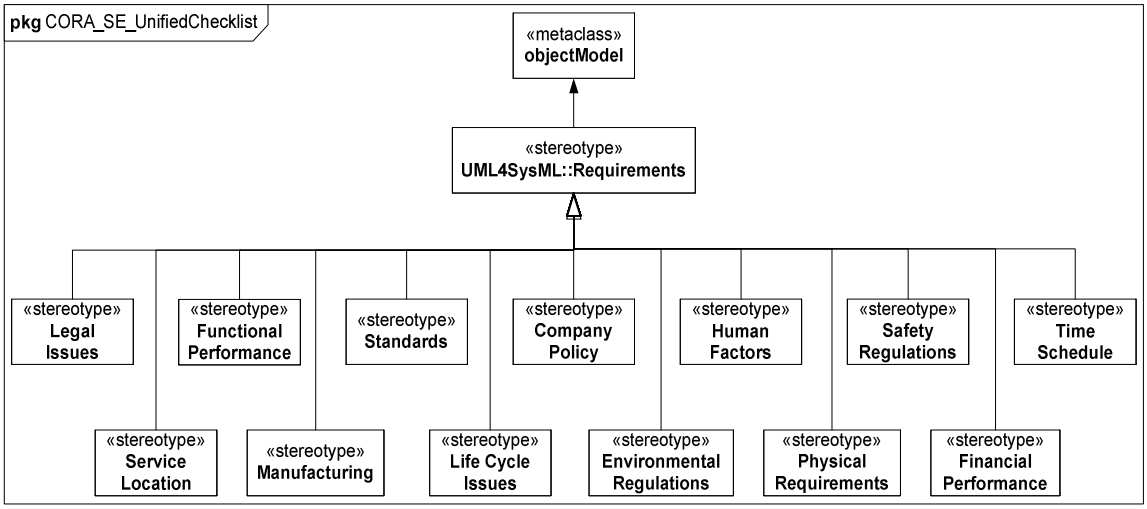

Figure 7. SysML requirements checklist stereotypes

5.2.2. Requirements information management

The checklist provides a structure which is used to store and manage information to support decision making. Information collected for requirements analysis can be classified into two major sources (Sudin *et al.* 2010):

- Human: customer, stakeholder, end user, market analysis, colleagues, expected solution, designers own documents.
- Artefacts: existing specification, proposed solution, existing product (i.e. benchmark), previous projects, design guidelines, user guidelines.

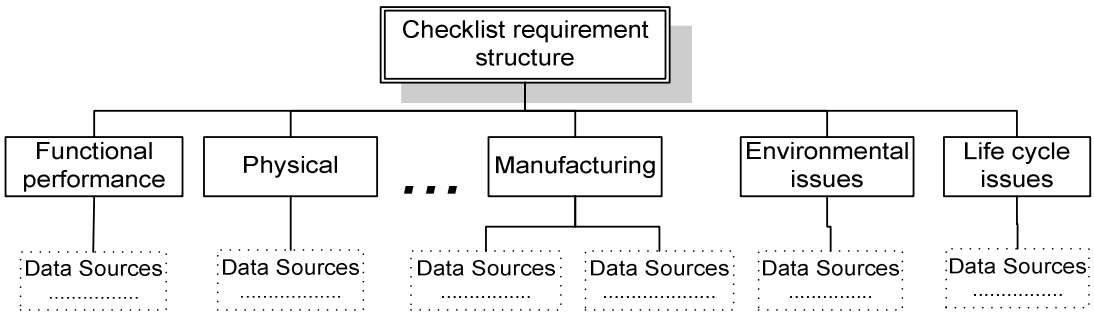

Figure 8. Checklist structure with associated data source

The checklist is used as a key to identify these sources. Both formal and informal data captured from these sources are organized with the checklist structure where each checklist is related to one or many other data sources as shown in Figure 8.

From our viewpoint, this form of information structuring should help to integrate other database systems (i.e. PDM, ERP) into a single database unit. Therefore, the unified checklist becomes the master structure and the most important source of information accessible to all. The checklist model (Figure 6) and a PDM system both have function and parts structure and can easily be integrated. Several other information systems (i.e. ERP, standards, and regulations) can be integrated through subsequent categories in the unified checklist. In existing PDM, to find information, it is necessary to understand the parts classification system and this is not an easy task. Hence, the part structuring paradigm in PDM can be substituted with checklist structure to include and manage all information.

In addition to information structuring, engineers involved in requirements analysis should also be organized effectively to establish links between them. Organizational links may be aligned with functions or projects (Ulrich and Eppinger 2004). Where, an organization function is an area of responsibility usually involving specialized education, training, or experience. According to the checklist structure, individuals are linked together based on organization function, and their expertise is explicitly stated for others to access. Typically, project managers share responsibilities in a project and are accountable for organizational links. Nevertheless, in this context, the matrix organization structuring is used to point requirement engineers to key individuals with the right expertise to collaborate with. Matrix technique is the underlining method adopted for information management, more specifically by adopting the House of Quality matrix based on the QFD-methodology as shown in Figure 9 (Akao 2004).

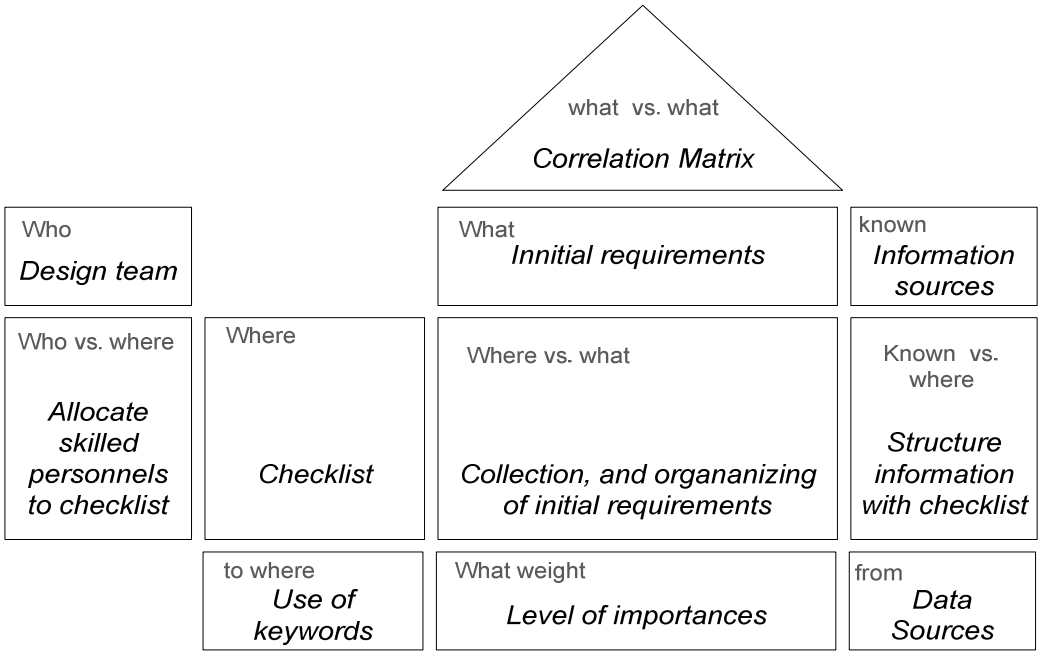

Figure 9. The checklist matrix based on house of quality method (adapted from Akao *et al.* 2004)

The matrix can also be used as a key to identify needs and to discriminate initial requirements under the various checklists. The outcome of matrix analysis becomes the input for subsequent stages in the requirement analysis. Keywords identified from the description and examples of each checklist can be useful in the matrix analysis. The keywords are also convenient during modelling and automation when an intelligent system is applied.

In SysML, the information type, are modelled as UML class diagrams or as blocks in a model library package. These are linked to the information system or repository and allocated to the various requirements checklist to direct the designer to the right information. As an example,

Figure 10 shows a UML model of design (i.e. maintenance) guidelines.

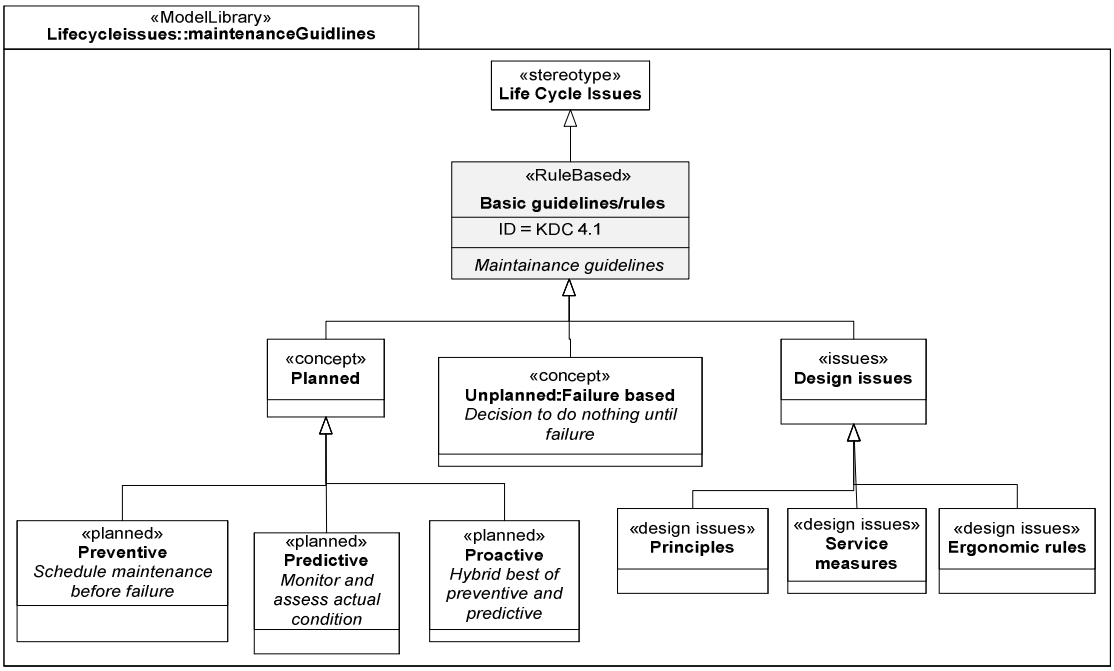

Figure 10. Information block diagram and UML class diagram of maintenance guidelines

5.2.3. Specification drivers

A significant task in requirements analysis is to identify requirements that are salient to get right, clarify what is important and what needs to be determined at what time. Sometimes the level of importance is evident, from the initial (i.e. customer, market, or business view) requirement statement. However, the level of technical (i.e. engineering, physical, or technological views) importance must also be defined. Thus, the two *fundamental views* are the *customer view* (i.e. business, environment), and the *engineering view* (i.e. technical constraint). Each of this view is an independent source of risk for the system, and must be coupled to ensure who drives whom. A *design driver* (referred to here as a specification driver) is the method used to analyze and distinguish between the two fundamental equilibrium conditions that must be satisfied (Otto and Wood 2001). The specification drivers are meant to be abstract representations of actual design variables. Consequently, they can take on different forms in distinct concepts.

As an example, we consider the specification drivers of a mobile work machine. What are they? Thinking as financial analysts and having in mind the environment aspect, we start with “Profit” and determine important factors that affect it. This forms the business case loop and for its development, there should be a clear understanding of environmental regulations (i.e., involvement of environmental engineers, regulators), customer and market needs (i.e., market analyst, customer liaison personnel).

The second step is to construct the technical constraint loop. Thinking as engineers, the reason we need power is to lift and transfer material from one point to another. Thus transfer of material is the fundamental constraint equation, and we determine other factors that affect it. Combining the business and technical loops into a diagram, we have Figure 11. Intersecting the loops by the common variables gives an understanding of the drivers.

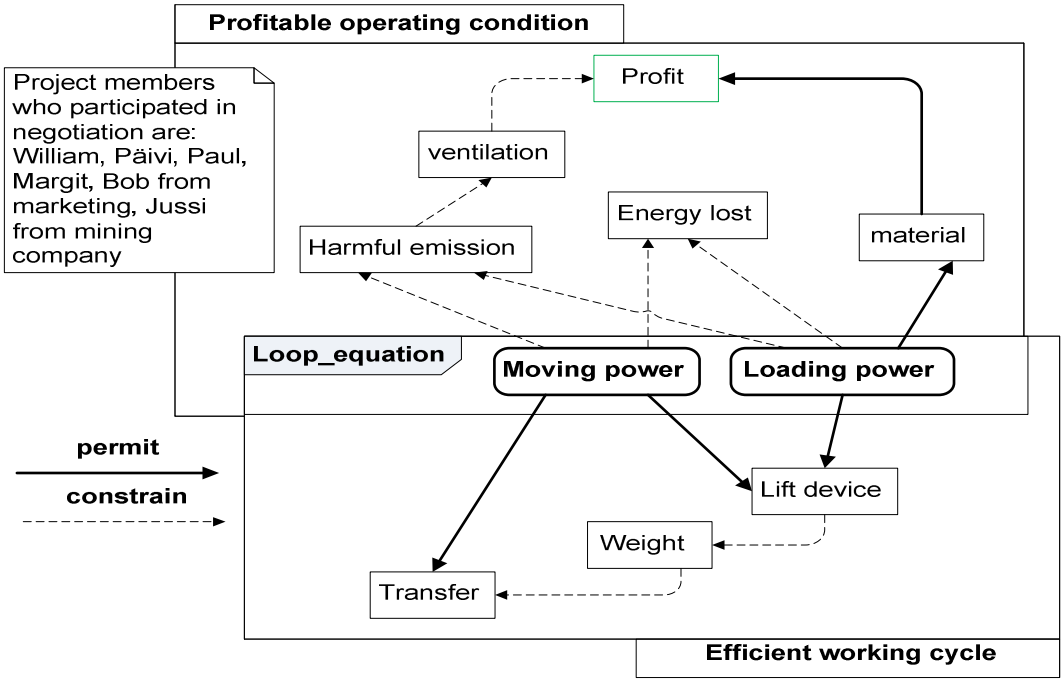

Figure 11. Specification driver model of underground mobile work machine (adapted from Otto and Wood 2001)

The overall analysis result is that decision variables are established earlier. The specification driver loop “equations” are not obvious and can be constructed in several ways. Different variables and performance criteria come to light during construction. The specification driver is a platform for negotiation and setting the context about the requirements and design at an early stage. The collaboration work is effortless as responsible engineers are easily identified from the checklist (Figure 9). In the SysML, context diagram, which refers to a user-defined usage of the internal block diagram are set to define the boundary in which to negotiate. Top level and goal level Use Case diagrams are constructed and are used together as a platform to clarify and understand the design problem to discriminate level of importance. The specification driver loop is modelled using *Internal Block Diagram.*

5.2.4. Functional and non-functional analysis

Requirements under the functional performance checklist are analyzed by logically arranging sub-functions based on priority. To help explore the functional requirements, a FAST (Function Analysis System Technique) approach is applied. As an example the load function of the mobile work machine is considered (Figure 12).

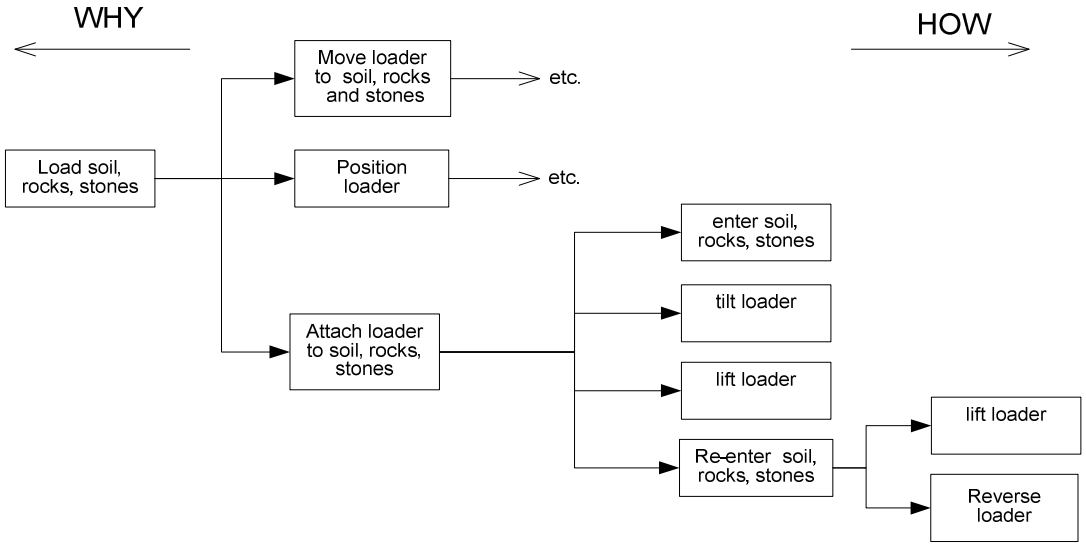

Figure 12. A (partial) FAST diagram, showing the decomposition of load function in mobile work machine

The approach is a diagram assisting designers to prioritize the activities or functions of a system. It is used to display functions in a logical sequence and to test their dependency (VAI 1993). In the FAST approach, for each sub-function, the question, “how is this to be met” is asked. “How” is answered by moving from left to right on the diagram; “why” is answered by moving from right to left. The diagram is constructed by asking “how” until the lowest level elemental function is established.

By extending the FAST diagram, important functions may be expanded into separate *function structures*, which identify the input parameters (i.e. flow of energy, material or information) and the output response of each function. The function decomposition helps the designer to highlighted requirements comprehensively. The separate function structure is then classified into function modules and used to define sub-systems to obtain more requirements with the checklist. In Sysml, the process is modelled with *Activity Diagram* and *Block Definition Diagram. Use Case Diagram* is used to create scenarios to help in this analysis process. In the next step, non-functional requirements are analyzed.

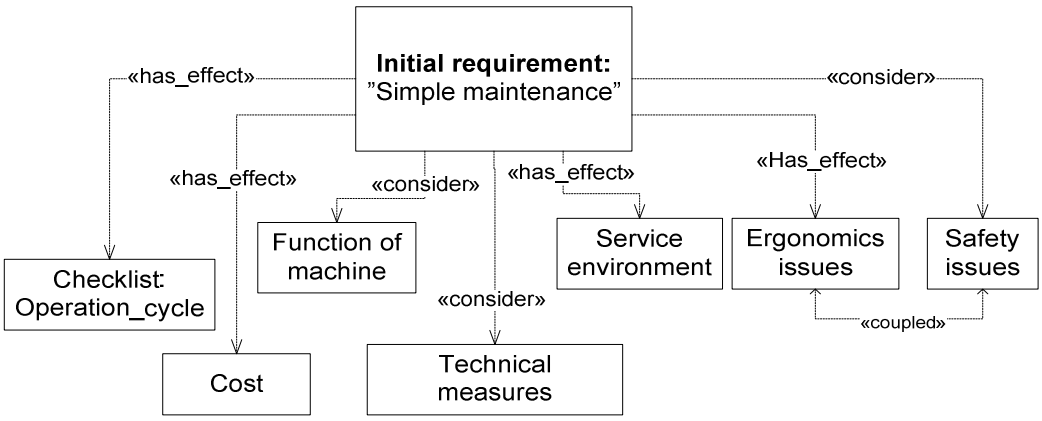

Figure 13. Analysis to identify information and dependencies between requirements

Most non-functional requirements are often unbounded, i.e. making relative statements that cannot be verified, as for example “simple maintenance." These requirements have to be restated to define specific bounds in order to transform them to be validatable. In this case, the requirement is first modelled (Figure 13) to identify

dependencies between other requirements in the checklist to extract information. The requirement statement is then developed through decomposition and refinement in a systematic manner. The analysis for the requirement statement “simple maintenance” is done mainly by applying design principles and guidelines (

Figure 10) as the main reasoning techniques. The SysML diagrams used in this analysis are the *Use Case* and *Internal Block Diagrams*. The analysis processes are iterative, and the process is applied with other sets of initial requirements to make the requirement specification well-formed.

5.2.5. Criteria network and analysis

Target values with unit information that the system should be designed to satisfy are progressively attributed to quantify the derived requirements (see Figure 1). This step is knowledge intensive, and the reasoning method is usually case-based. Knowledge from information obtained in RIM is used (Figure 9). The use of existing information is crucial to help in decision making. If, for instance, we have two related criteria and there is the need to find the target value of one based on the other, then RIM is valuable (consider the example in Figure 14).

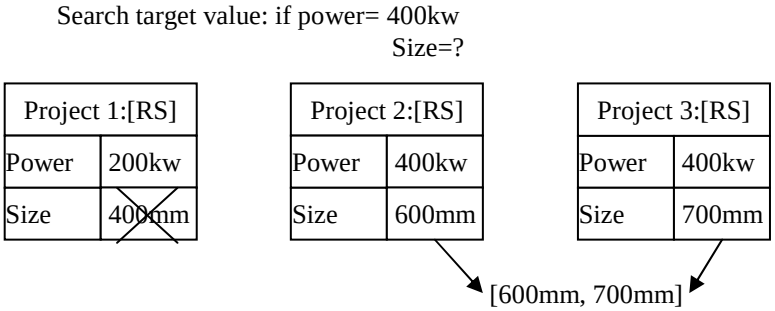

Figure 14. The use of existing requirement specification to set target values

In addition to setting target values, *physical quantity* symbols and *units dimensions* are also assigned to each criterion.

Several criteria may be proposed for a single requirement statement. The target value should be practical and since not all the criteria may be relevant, there should be more analysis to refine. For that reason, first the criteria are connected

together to form a network (criteria network). This helps to manage the criteria relationships and identify criterion, which appears in multiple requirements (Claros Salinas *et al.* 2008). Furthermore, the *dimensional variables* of units of measurement are extracted and connected to create a network. The dimensional variable networks can be modelled using P-calculus and dimensional analysis to analyze the criteria (Brace *et al.* 2009). *Block Definition Diagram*, *Parametric Diagram* and *State Machine Diagram* are used to model the criteria networks in SysML. In the following, we show an application of the framework and modelling with SysML.

6. Application example and evaluation

6.1. Application example

The proposed framework can be applied to different types of design problems. The design problems may apply to distinct design modules namely, original (novelty), adaptive (using established solution principles) or variant (modular) design.

The research is taking place in the context of adaptive design. An existing specialist and relatively complex product is to be redesign. The physical product is systematically analyzed to gain knowledge. Competitive products are benchmarked. The product specification and documents are also available. The requirement engineer received a design problem in the form of a short narrative statement as follows:

"The task is to develop an underground mobile work machine for loading, transferring and dumping soil, rock, and stones. The aim is to reduce the energy consumption and harmful emissions in the machine. The primary market area is mining companies globally. The development should be based on the existing machine. The outer geometry of existing machine does not change. The project is to last for five years in two phases. The project cost should not exceed allocated budget.

The developed machine should be easy to maintain and safety issues should be considered."

Applying CORA framework, information from various sources is managed with the requirement checklists as discussed in section 5.2.2. Using this structured information, the narrative statement is analyzed to generate requirements through the following steps. First, the statement is set into the initial requirements and discriminated under the various checklists using the matrix checklist template (Figure 9).

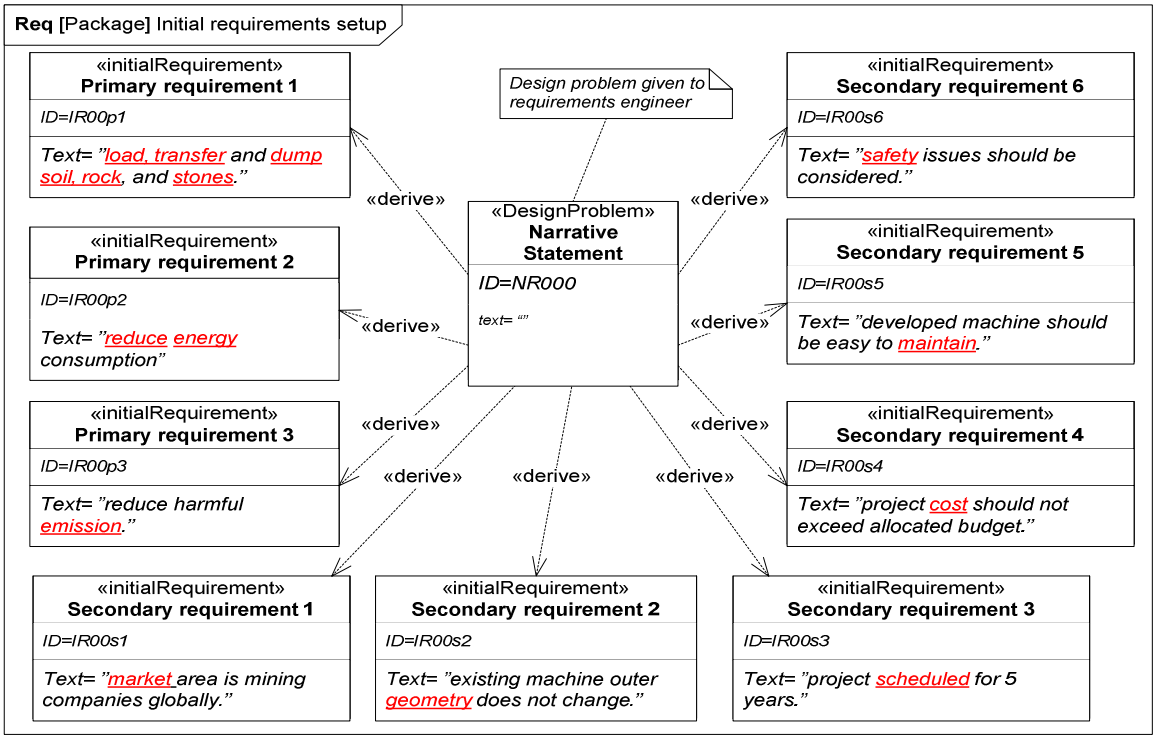

Figure 15. Initial requirements model in SysML

Keywords (i.e. underlined and shown as red in Figure 15) are identified to help in modelling. The specification driver is modelled as shown in Figure 11. The initial requirements are then structured into primary and secondary goals (i.e. level of importance) as shown in Figure 15.

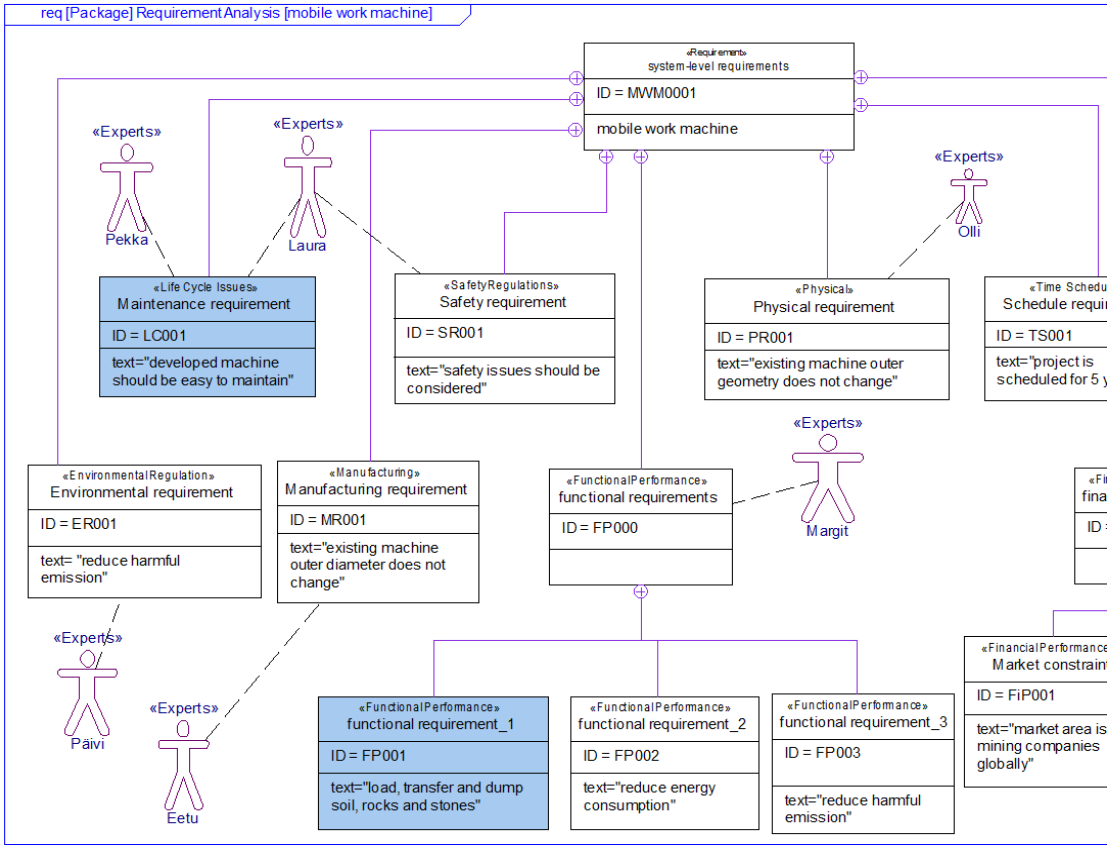

Figure 16. SysML Requirements diagram of mobile work machine

The identified initial requirements are set under the various checklists and modelled as shown in Figure 16. Dependencies can already be established at this early stage as some checklists contain similar requirements. Expert engineers or personnel to collaborate with are also pinned to the checklists to direct designers.

The next step is to analyze the requirements set under the checklists to derive and establish more requirements and dependencies. The functional requirement statement “Load, soil, rocks and stones” are analyzed in this example. The top-level operational activity is first modelled (Figure 17a). A Block Definition Diagram of the main function is modelled based on the FAST approach (i.e. as discussed in section 5.2.4). The interaction between system and environment is modelled with Use Case Diagram using information from RIM (Figure 17b). An operator is modelled as the external environment to control the load activity.

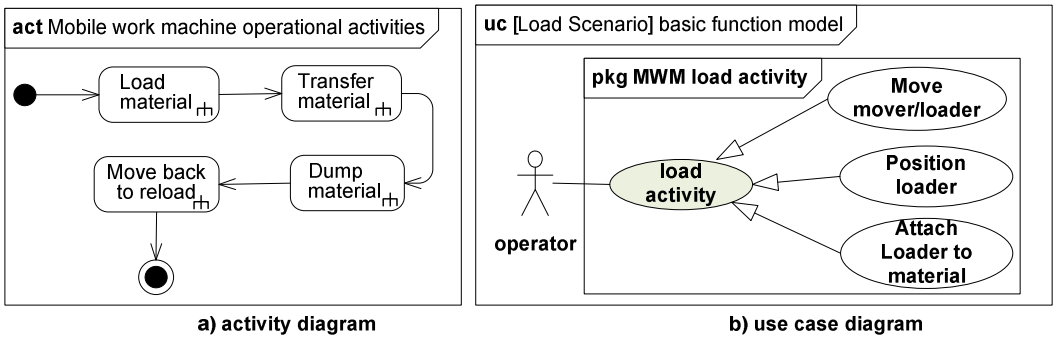

Figure 17. SysML activity diagram, internal block diagram, and use case diagram to analyze functional requirement

The entities (e.g. operator, loader, material) and the functions (i.e. move, position, attach) are expanded and further analyzed to derive more requirements. At the same time, dependencies between the various checklist requirements (i.e. functional requirements and non-functional requirements) are identified (Figure 18). The requirement diagram is updated with newly identified requirements.

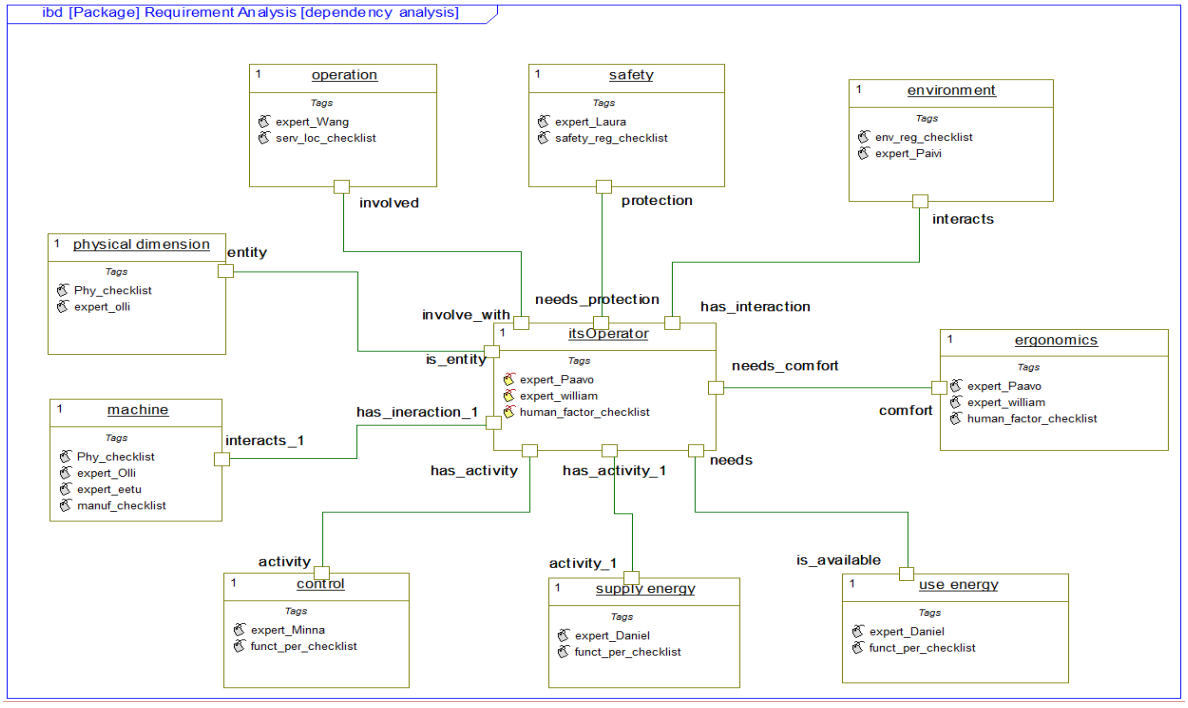

Figure 18. Internal block diagram of functional requirement entity “operator” showing dependencies

Next we analyze the non-functional requirements. As an example we analyze the requirement statement “developed machine should be easy to maintain” (see

Figure 16). First, we model to identify dependencies with other checklist and to identify sources of relevant information and designers to collaborate with (Figure 19).

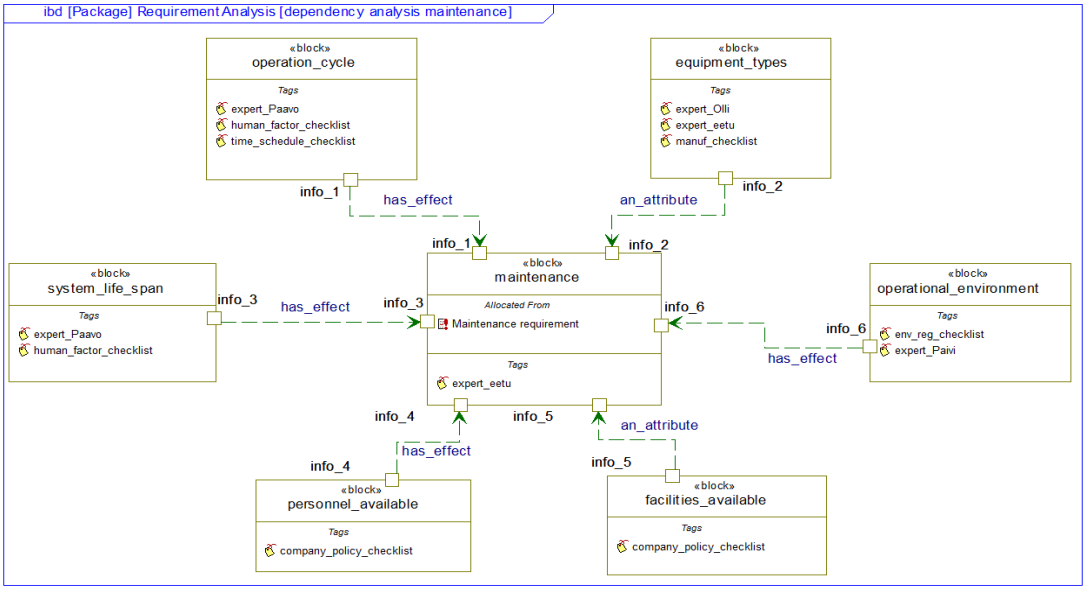

Figure 19. Internal block diagram of non-functional requirements “maintenance block” showing dependencies

The maintenance guidelines (
Figure 10) together with knowledge from identified sources (Figure 19) are applied to analyze and derive more requirements (Figure 20).

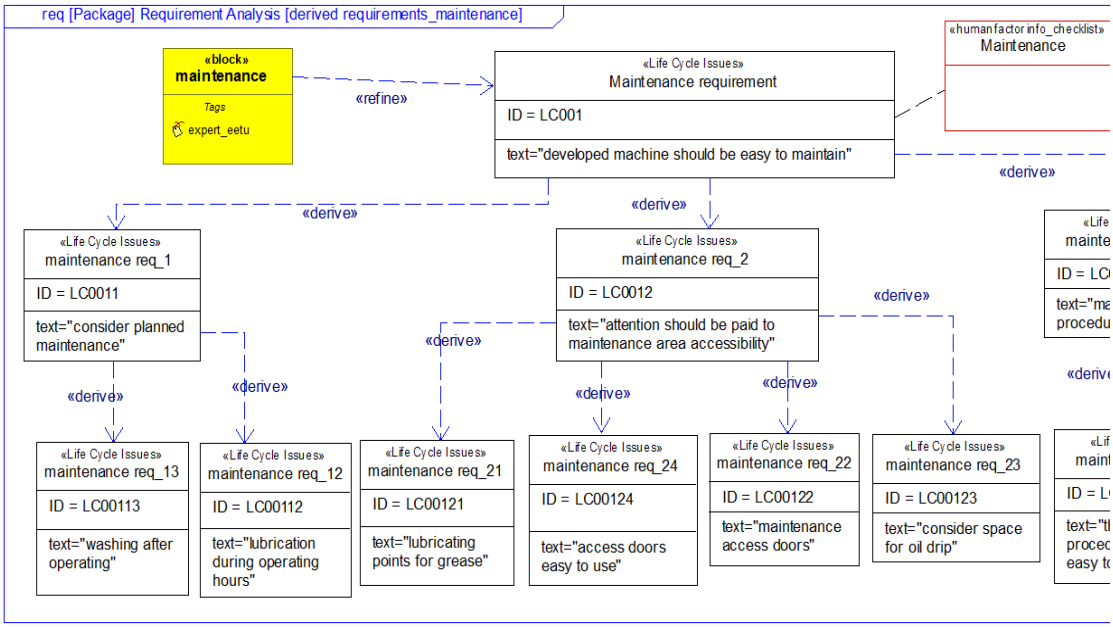

Figure 20. Derived maintenance requirements SysML diagram of mobile work machine

This procedure is continued to obtain all relevant and identifiable requirements with other non-functional requirement statements and to update the requirement diagram.

So far, the requirements identified are in the qualitative form. The next step is to quantify wherever possible and to validate. First, a list of attributes is coupled with the requirements derived. Several attributes may be necessary to reflect completely a single requirement (

Table 4). Second, the units of measurement, target values with tolerances are also related to each attribute. The analysis in this step is knowledge sensitive and involves the use of existing information. Therefore, available information from the various sources is compounded to set the target values for the attributes.

Table 4. Excerpt of concept requirements list

ID	Category type	Priority	Requirements statement	Attributes	Measurable criteria			Physical quantity	
					Target value	Target value (tolerance)	Units	symbol	dimensions
FP0021	FP000	primary	the machine shall load soil, rock, and stones	transfer torque	428Nm at 1500rpm	+/-3% of value	Nm	T	
				energy losses through - friction -heat -noise	 >15.6% >62.4% >22%	 reduce by at least -50% at least -50%	 N J J	 f Q Th	 $[MLT^{-2}]$ $[ML^2T^{-2}]$ $[ML^2T^{-2}]$
				power transfer source	diesel motor output power 90KW	consider a hybrid power source	W	P	$[ML^2T^{-3}]$
				material weight	17965kg	maintain n. value	N	w	$[MLT^{-2}]$
				loading force	9072kg	maintain	N	Fl	$[MLT^{-2}]$
				loading time: Rising lowering	 5sec. 3sec	 ≤ 5sec ≤ 3sec.	s	t	[T]
LC003M11	LC001	secondary	maintenance intervals between operating hours	operation time interval	12,000 hours	≈18,000 hours	s	t	[T]
				cost per hour	22£	≤22£	£/s	tc	$[£T^{-1}]$
				number of personnel	3	<3	£/s	tc	$[£T^{-1}]$

The SysML models in Figure 18 and Figure 19 also helps in identifying attributes and target values. The derived requirements (

Table 4) which are in the primitive format (Figure 1) are written in complete sentences and organized to satisfy the requirement specification format. Each requirement statement in the specification must satisfy several characteristics, including the appropriate use of shall, will and other keywords. They must also have

proper grammar and rigid compliance with a format (i.e., company format). Following is an example (from Table 4):

"The machine shall be able to load soil, rock, and stones with weight less than or equal to 17965 kilograms"

This step is also iterative and continues until a well-formed requirement specification is established.

Validating the requirements means ensuring that 1) the set of requirements is complete, correct and consistent, 2) a model that satisfies the requirements can be created for i.e., simulation purposes and 3) a real-world solution can be built and tested to verify the requirements (Bahill and Henderson, 2005). The measurable criteria should be dependent variables and practical with the proper unit (see

Table 4). For instance, the requirement statement "if 1500 < material weight <17965, load machine" is incomplete. What should happen if weight of material is less than 1500? The requirement is in-consistent, what should happen if material weight is equal to 1600? The requirement is also incorrect because the units are not given. Is the material weight in grams or kilograms? The dimensions (

Table 4) are used to create the criteria network for dimensional analysis (DA) modelling and Petri Nets (PN) behavioural simulation (Brace et al. 2009). Practical application of DA and PN typically relies on a combination of interactive and automatic simulation, visualization, functional analysis and criteria analysis. These activities justify that the derived requirements have desired attributes and target values. In addition, a high degree of confidence and understanding of the requirements has been obtained early in the requirements analysis phase.

6.2. Theoretical evaluation

Since empirical evaluation requires the real implementation of the framework in an industrial setting, it is impossible to conclude the evaluation at this point. However, two comprehensive theoretical validation procedures are used for theoretical evaluation: relative validation, absolute validation. The following sections will explain the validations respectively.

6.2.1. Relative validation

Key issues (Table 3) are used for the relative validation based on the results of our case study example. The case study demonstrates the difficulty of precise and comprehensive *handling of the complexities in a requirements analysis process*. Developing any large specification is a challenging exercise in managerial and organizational terms. CORA-framework attempts to manage complexity by providing carefully structured methods for effective analysis of the requirements. Shared understanding and negotiation between the project stakeholders is essential to *stimulate a multidisciplinary approach to requirements analysis*. The specification driver is a platform to set the requirement context at an early stage thereby *increasing* stakeholder *maturity* and shared understanding. The right person is easily identifying from the checklist structure, therefore, encouraging a *responsibility approach for each designer*. The checklist is also intended to manage data, and this helps in *information retrieval and knowledge re-use*. Requirements that have no direct bearing on the functionality of the product and its essential characteristics are identified and omitted.

As indicated in the case study, the modelling approach creates links and dependencies between requirements in the various checklists. This form of dependency is missing in most existing methods, where the link is mostly through part structure. The supporting tool helps to apply an interactive method to the systematic method of the framework.

The application of SysML facilitates to support for traceability and validation through linking with other simulating tools. SysML attempts to impose formalisation of the requirements process through transition between the sequential thinking and structural thinking. The design team has at their disposal an integrated design tool in addition to the intuitive methods, creativity and experience. The formalized and system-level modelling is a step towards an automated requirements analysis process to eliminate a labour-intensive approach. One of the major advantages of the tool support is the common pictorial view of the requirement process. In short, the requirements analysis process supports all the key issues as indicated in Table 3. As a result, the proposed framework is validated according to the relative validation.

6.2.2. Absolute validation

This validation is in two steps. The first step is based on the Three Dimensions of requirements engineering. We considered how the CORA method achieved the suggested scale as shown in Table 5.

Table 5. The Three Dimensions of RE (adapted from Pohl, 1994)

Fundamental dimensions	Goals	Suggested scale
Specification	Developing as complete as possible system specification out of opaque views existing at the beginning of the process	From opaque through fair to complete
Representation	Providing integrated representation formalisms and supporting the transformation between them	From informal, semi-formal and formal formats
Agreement	Allowing various views and supporting evolution from personal views to common agreement on final specification	From personal view to a common view

A justified assessment for the specification dimension is between fair and complete. The systematic approach behind CORA decomposes and analyzes the need from an opaque to a well-formed requirement. The representation format was the clearest since the CORA framework suggests semi-formal representations and augments them with a formal SysML approach. The framework supports a multi-disciplinary approach with a negotiation platform thus advocating a common view in the agreement dimension.

The second step is based on the requirement-related project risk factors related project risk factors (Nikula 2002). CORA addresses these factors as shown in Table 6. It tries to alleviate the likelihood of these factors and makes the presence of risks apparent to take proper action.

Table 6. Absolute validation with requirements related risk factors (adapted from Nikula, 2002)

Risk factor	CORA method to mitigate the risk factor
Misunderstanding the requirements	Requirements documentation Requirements modelling Specification driver platform for negotiation
Lack of adequate user involvement	Establishing links with checklist structure Use of visual and semantic modelling tool SysML to enhance user interaction
Failure to manage end user expectations	Interacting with users through modelling tool Documenting requirements Prioritizing requirements Validating requirements with criteria network
Changing scope/objections	Modelling and documenting goals, context, and requirements Managing requirements Requirement information management, change request
Lack of frozen requirements	Use of Checklist decomposition Documenting requirements Change management
Conflict between user departments	Matrix checklist to allocate skilled personnel's Multidisciplinary approach Use of modelling tool for concurrent work Encouraging responsibility
Incomplete requirements and specification	Checklist decomposition technique Checklist analysis technique Criteria network and analysis Validating requirements
Ambiguous and vague requirements	Building shared understanding through specification driver Modelling requirements Checklist oriented decomposition and analysis Validating requirements

This proves that there are precautions in the proposed framework to eliminate risk factors. Therefore, the framework is theoretically validated against risk factors. Consequently, the absolute validation is concluded with the completion of this second step.

6.3. Discussion

There were multiple evaluations made on the framework. Evaluation with designed key factors indicates the CORA process supports all key factors. Evaluation against the Three Dimensions shows the process lies in the middle showing a balanced

approach to the different areas of RE. CORA placed precautions to eliminate project risks as noted in the risk factor evaluation. It is concluded that following the CORA framework should introduce new requirements analysis practices in the engineering design domain and consequently, improve the quality of requirements analysis in general.

However, it was also clear that there are limitations. Clearly, the current flat structure of CORA framework may be too rigid and heavy-weight for some practitioners, but we maintain that this is not a unique problem as a light-weight approach tends to be more document-centric. The *responsibility approach* involves a distributed process where designers have to perform requirements analysis prior to switching to design solution search. We should all recognize that this approach puts a lot of stress on the designer. Design engineers are creative people. Design work is creative work. It is probably true that the creative engineer will have difficulties adapting to an organized requirements analysis methodology. Nevertheless, the authors believe that we will realize a better mix of requirements quality and synthesis excellence by involving them. In addition, the structured process has similar aims to the creative method, such as widening the search space for a potential solution.

Another drawback is that, the framework can be applied rigidly and in ways that are improper and consumes company money and time. The nature of the design product and design activities differs in engineering design. Therefore, it is not clear whether the framework is suitable for other domains (i.e. software engineering, electronics engineering), and has to be investigated in the future.

The application of the framework also depends on the existence of problem/solution bias in design strategy and on product complexity. The framework may not be suitable for a design strategy that does not begin with an analysis of the

needs. Nevertheless, in a solution-oriented design strategy, it can be linked to the strategy to create simultaneously, a set of ideas and requirements. For non complex products, there may be some difficulties encountered as the framework cannot be used to its fullest extent. The circumstances surrounding a design product have a bearing on the actual complexity of the product. Therefore, the use of the framework can be influenced by the experience that a particular company and individual have on the product.

7. Conclusion

We have addressed the requirements analysis from an engineering design perspective with the aim of exploiting a systematic and model-driven approach. We define a checklist oriented requirements analysis (CORA) framework for deriving requirements. However, a problem arises when attempting to compare CORA with other works, e.g. (Parvianen *et al.* 2003). The framework cuts across many existing approaches to support a process of deriving requirements. Short of reviewing all methods, it is difficult to provide a detailed comparative analysis of the approach. Nevertheless, the CORA framework is substantially different from many existing approaches and will improve a design project. Key points distinguishing our approach are used to answer the three main research questions.

7.1. CORA framework for formal requirement analysis process

In our first research question, the emphasis was on the lack of a formal process. Current requirements analysis processes in the engineering design domain rely extensively on informal processes, largely based on a free-style strategy. In reality, this approach is for experienced engineers already familiar with the product line appropriate to the customers' need. This strategy carries with it the danger of possible incompleteness due to lack of rigor in the analysis process. Therefore, it does not

provide the desired level of assurance for the derived requirements. Furthermore, there is a significant lack of effective methods and tool support for the requirements analysis in comparison to detail design. Interaction between requirements can be hard to identify, let alone validate.

A formal basis for the proposed approach was by combining requirements analysis practices in engineering design with current software and systems engineering practices, such as SysML, and knowledge management. The formal method behind the framework helps provide interaction, rigor and validation as demonstrated in the case study. It provides a simple validation check, as it forces a level of explicitness far beyond that needed for informal representations. Furthermore, it demonstrated the potential of SysML as a flexible tool. The formal process generated function structures together with the requirements. This encourages a clearer formulation, and leads to the identification of new requirements. A well-defined function structure is also useful in later phases during the search for a solution. Function structure together with criteria network allows failure analysis early in the design phase.

7.2. CORA framework for more expressive requirements

In the second question, the concern is about the document-centricity and labour intensiveness of the traditional approach. Our answer is demonstrated in the potential of system-level modelling as a method to support and facilitate an inter-disciplinary and multi-level engineering. Through the use of SysML which defines the notation (visual representation) and semantic (meaning) used to construct the model. Modelling and simulation are indispensable when dealing with complex engineering products. It enables an essential assessment before products are built, can alleviate the need for expensive experiments and can provide support in all stages of a project.

Very often, system modelling in the engineering design domain means constructing a descriptive and explanatory mathematical model of the system. In the field of software engineering, there is an overwhelming tendency to see pictures and diagrams as a form of the model that helps in visualizing and communicating. Visually presented information is comprehended more easily by the human mind compared to the difficulty in comprehending the meaning of 100 words. This level of visualization and communication is quite evident in the later phases of engineering design, but in the requirements analysis phase, it still remains a challenge. Modelling the requirements analysis process provides an opportunity to address many of the limitations of the document-based approach.

7.3. CORA framework for requirement information management

The third question exposes the empirical nature and difficulties in the re-use of available knowledge and information. We answered through the use of a requirement information management. The approach makes it possible to develop knowledge-based systems and to link and use stored data and methods (PDM, CIM). The CORA framework creates an important basis for information system management and integration. It also acts as an information search portal and as a valuable communication source. Novice engineers have ready access to information, thus eliminating personal preferences and improving effectiveness and productivity. The framework is also useful in pointing out expertise thereby improving negotiation and collaboration, which are pre-requisites for requirements analysis. The application of the checklist along with the capability to store and apply additional information will significantly enhance the re-use of knowledge.

7.4. Perspectives

The framework is demonstrated with an on-going research project, but more work needs to be done in the following direction. Since a first justification has been established, future work will complete this analysis by focusing on the exploitation of machine readable knowledge representation that will favour partial automation. There will be a further evaluation of the CORA-framework in several industrial settings to complete the evaluation. Customize existing information management systems based on the checklist structure and the co-operation with PDM systems are other areas of future consideration.

References

Akao,Y., 2004. *Quality function deployment: Integrating customer requirements into product design*. 1st ed. Cambridge: Productivity Press.

Amihud, H., Kasser, J.and Weis, M., 2007. How lessons learned from using QFD led to the evolution of a process for creating quality requirements for complex systems. *Systems Engineering and Management, 10* (1), 45-63.

Bahill, A.T. ,Dean, F.F., 1996. What is Systems Engineering? A Consensus of Senior Systems Engineers. *Proceedings of the Sixth Annual Symposium of the International Council on Systems Engineering* (INCOSE), 7-11 July 1996 Boston. USA: INCOSE, 1, 503-508.

Bahill, T. A.; Henderson, S. (2005). Requirements development, verification, and validation exhibited in famous failures. *Systems Engineering*, 8 (1), 1-14.

Baumberger, C. and Lindemann, U., 2006. Requirement oriented process planning and configuration. *Proceedings of NordDesign*, 16 – 18. August 2006, Reykjavik. Iceland: 244-254.

Baxter D., Gao J., Case K., Harding J., Young B., Cochrane S., Dani S., 2007. An engineering design reuse methodology using process modelling. *Research in Engineering Design,* 18(12), 37-48.

Berkovich, M., Leimeister, J. M.and Krcmar, H., 2009. An empirical exploration of requirements engineering for hybrid products. *17th European conference on Information Systems.* Verona, Italy.

Blanchard, B. and Fabrycky, W., 2006. *Systems engineering and analysis*. 4th ed. Prentice hall: International series in industrial and systems engineering

Brace W., Coatanéa E., Kauranne H., Heiska M., 2009. Early Design Modelling and Simulation of Behaviours: Case study of mobile work machine. *Proceedings of ASME- IDETC*, 30 August – 2 September 2009, San Diego, USA.

Brooks, F., 1987. No Silver Bullet – Essence and Accident in Software Engineering. *IEEE Computer*, 20 (4), 10-19.

Chan K.Y., Kwong C.K., Dillon T.S., & Fung K.Y., 2010. An intelligent fuzzy regression approach for affective product design that captures nonlinearity and fuzziness. *Journal of Engineering Design*, 0954-4828.

Checkland, P., 1981. *Systems thinking, systems practice*. Chichester, UK: Wiley.

Claros, Salinas M. P., Prudhomme G. and Brissaud D., 2008. Requirement-oriented activities in an engineering design process. *International Journal of Computer Integrated Manufacturing*, 21(2), 127 – 138.

Cooper, R., Wootton, A.B. and Bruce M., 1998. Requirement capture: theory and practice. *Technovation*, 18(8–9), 497–511.

Cross, N., 2008. *Engineering Design Methods Strategies for Product Design*. 4th ed., West Sussex, England: John Wiley & Sons.

Darlington, M. J. and Culley, S. J., 2002. Current research in the engineering design requirement. *Proceedings of Inst. Of Mechanical Engineers*, IMechE, Part B: Journal of Engineering Manufacture, 216 (3), 375-388

Davis, A. M., 1990. *Software requirements: analysis and specification*. Englewood Cliffs, NJ: Prentice Hall.

Dieter, G., 2000. *Engineering design: A material and processing approach*. 3rd ed. international Editions: McGraw- Hill

Douglas, S., 2006. Model-Driven Engineering. *IEEE Computer*, 39 (2), 25-31.

Douglass, P. B., 1999. *Doing Hard Time: Developing Real-Time Systems with UML - Objects, Frameworks, and Patterns*. Addison-Wesley.

Drejer, A., 2008. Are you innovative enough? International Journal of Innovation and Learning. 5(1), 1-17.

Effendi I., Henson B., Agouridas V., De Pennington A., 2002. Methods and tools for requirements engineering of made-to-order mechanical products. *Proceedings of ASME- IDETC*, 29 September – 2 October 2002, Montreal, Canada.

Flyvbjerg, B. 2006. Five Misunderstandings About Case-Study Research. *Qualitative Inquiry* , 12 (2), 219-245.

Follmer, M.; Hehenberger, P.; Punz, S.; Zeman, K., 2010. Using Sysml in the Product Development Process of Mechatronic Systems. *International Design Conference - Design 2010*, 17 - 20May 2010, Dubrovnik. Croatia.

Friedenthal, S., Moore, A., and Steiner, R., 2008. *A practical guige to SysML the systems modeling language*. Morgan Kaufmann: Elsevier Science

Gilb, T., 1997. Viewpoints: towards the engineering of requirements. *Springer Computer Science*, 2 (3), 165–169.

Grady, J., 2006. *System Requirements Analysis.* international Editions: Elsevier Inc.

Greer, J., Stock, M., Stone, R. and Wood, K., 2003. Enumerating the component space: First steps toward a design naming convention for mechanical parts. *Proceedings of ASME- IDETC*, 2-6 September 2003, Chicago, USA.

Gupta, A., Pawara, K. and Smart, P., 2007. New product development in the pharmaceutical and telecommunication industries: A comparative study. *International Journal of Production Economics*, 106(1), 41–60.

Hall, T., Beechem, S. and Rainer, A., 2002. Requirements problems in twelve software companies: an empirical analysis. *IEE Proceedings Software*, 149, 153-160.

Hicks B.J., Culley S.J., Allen R.D., Mullineux G., 2002. A framework for the requirements of capturing, storing and reusing information and knowledge in engineering design. *International Journal of Information Management*, 22(4), 263-80.

INCOSE, 1998. *Terms Glossary Version 0*. INCOSE SE.

Jiang, L., 2005. *A framework for the requirements engineering process development*. Thesis PHD. University of Calgary.

Kotonya, G. and Sommerville, I., 1998. *Requirements Engineering: Process and Techniques*. John Wiley & Sons.

Loftus, C., Hicks, B. and McMahon, C.A., 2009. Capturing key relationships and stakeholders over the product lifecycle: an email based approach. *6th Int. Conf. on Product Lifecycle Management*, PLM'09, 6-8th July 2009. Bath, UK.

Loucopoulos, P. and Kavakli, E., 1995. Enterprise Modelling and the Teleological Approach to Requirements Engineering. *International Journal of Intelligent and Cooperative Information Systems*, 4(1), 45-79.

Lowe, A., McMahon, C. and Culley, S., 2004a. Information access, storage and use by engineering designers - Part 1. *The Journal of the Institution of Engineering Designers*, 30 (2), 30-32.

Macaulay, L. 1996. Requirements for requirements engineering techniques. *Proceedings of the Second International Conference on Requirements Engineering -IEEE*, Colorado Springs, CO , USA, 157 – 164.

McAlpine H., Cash P., Howard T., Arikoglu E.S., Loftus C., and O'Hare J., 2010. Key Themes in Design Information Management. *International Design Conference - Design 2010*, 17 – 20 May 2010, Dubrovnik. Croatia.

McConnell S., 1996. *Rapid development: taming wild software schedules*, 1st ed., Redmond, WA: Microsoft Press.

Moores, T. T. and Gregory, F. H., 2000. Cultural problems in applying SSM for IS development. *Journal of Global Information Management*, 8(1), 14-19.

Nuseibeh, B. and Easterbrook, S., 2000. Requirements engineering: a roadmap. *Proceedings of the Conference on the Future of Software Engineering*. ACM Press, 35-46.

Niksula, U. 2002. *BaRE-A Ready to Use Method for Requirements Engineering*. Licentiate Thesis, Lappeenranta University of Technology, Finland.

OMG, 2008. *Systems Modeling Language V1.1*. OMG Available Specification.

Otto, K. and Wood K., 2001. *Product Design: Techniques in Reverse Engineering and New Product Development*. Upper Saddle River, NJ: Prentice Hall.

Pahl, G. and Beitz, W., 2007. *Engineering Design. A Systematic Approach*. 3rd ed. Wallace K. and Blessing L., translation and edition Berlin-Heidelberg: Springer.

Parviainen P., Hulkko H., Kääriäinen J., Takalo J., Tihinen M., 2003. Requirements engineering. Inventory of technologies. *VTT publications 508*. Espoo Finland.

Peak R.S., Burkhart R.M., Friedenthal S.A., Wilson M.W., Bajaj M., Kim I., 2007. Simulation-Based Design Using SysML Part 1 and Part 2. *INCOSE Intl. Symposium*, San Diego, USA.

Pisano, G. and Wheelwright, S., 1995. The new logic of high–tech R&D. *Harvard Business Review*, 73 (5).

Pohl, K., 1994. The three dimensions of requirements engineering: A framework and its applications. *Information Systems*, 19 (3), 243-258.

Pugh, S., 1997. *Total design: integrated methods for successful product engineering*. Essex UK: Addison-Wesley Longman Ltd.

Rechtin, E., 2000. *Systems Architecting of Organizations*. CRC Press.

Short, T.D., Garside, J., Appleton, E., Morris, A., McEachran, H., Beeley, E., 2009. Matching the Voice of the Engineer to the Voice of the Customer: An Evolution of QFD. *In*: Norell Bergendahl, M., Grimheden, M., and Leifer, L., eds. Proceedings of ICED'09. *17th International Conference on Engineering Design ICED'09*, 24-27 August 2009 Stanford.USA: Design Society.

Soares, M. S., and Vrancken, J., 2008. Model-Driven User Requirements Specification using SysML. *Journal of Software*, 3(6), 57-68.

Sommerville, I., 2001. *Software Engineering*. 6th ed. Addison-Wesley.

Stake, R. 1995. *The art of case research*. Newbury Park, CA: Sage Publications.

Strauss, A.,Corbin, J. 1998. *Basics of Qualitative Research: Techniques and Procedures for Developing Grounded Theory*. 2nd, ed. Thousand Oaks, CA: Sage Publications.

Sudin, M.N., Ahmed-Kristensen S. and Andreasen, M. M., The role of specification in the design process: A case study. *International Design Conference - Design 2010*, 17 – 20 May 2010, Dubrovnik. Croatia.

Svensson, D., and Malmqvist, J., 2001. Integration of Requirement Management and Product Data Management Systems. *Proceedings of DETC'01*, 9-12 September 2001, Pittsburgh, PA, USA.

Ullman, D., 2002. *The Mechanical Design Process*. 3rd ed. New York, NY: McGraw Hill.

Ulrich, K. and Eppinger, S., 2004. *Product Design and Development*, 3rd ed., McGraw-Hill

VAI, 1993. *Value analysis, value engineering, and value management*. Clifton Park, New York: Value Analysis Inc.

van Lamsweerde, A., 2000. Requirements engineering in the year 2000; a research perspective. *ACM Press*, 5-19, New York.

Ward J., Shefelbine, S. and Clarkson P. J., 2003. Requirements capture for medical device design. *International conference on engineering design 1ced 03*, 19-21 August 2003, Stockholm.

Weilkiens, T., 2008. *Systems Engineering with SysML/UML: Modelling, Analysis, Design.* Morgan Kaufmann.

Wiegers, Karl E., 2003. *Software Requirements 2: Practical techniques for gathering and managing requirements throughout the product development cycle*. 2nd ed., Redmond: Microsoft Press.

Yin, R. 1994. *Case study research: Design and methods*. 2nd, ed. Thousand Oaks, CA: Sage Publishing.

Zave, P. and Jackson, M., 1997. Four Dark Corners of Requirements Engineering.

ACM Transactions on Software Engineering and Methodology, 6(1), 1-30