

Integration of functional, cognitive and quality requirements. A railways case study

M. Felici^{a,*}, M.-A. Sujan^b, M. Wimmer^c

^a*LFCS, Division of Informatics, The University of Edinburgh, James Clark Maxwell Building, The King's Buildings, Mayfield Road, Edinburgh, Scotland EH9 3JZ, UK*

^b*Institute for Computer Design and Fault-Tolerance, University of Karlsruhe, Karlsruhe, Germany*

^c*Institute of Applied Computer Science, University of Linz, Linz, Austria*

Abstract

The paper shows a SHEL oriented requirements engineering approach, which has been applied in a case study dealing with the definition of the requirements for a new railways traffic control system. The SHEL model provides an integrated view by considering any productive process or activity performed by a combination of Hardware, Software and Liveware resources within a specific Environment. A set of SHEL oriented requirements describes the different views in a complex system. The requirements are grouped into three main classes, namely, functional, cognitive and quality requirements. The paper points out the issue of integrating different types of requirements. © 2000 Elsevier Science B.V. All rights reserved.

Keywords: Requirements engineering; Software engineering; Human factors

1. Introduction

Requirements engineering [18,23] approaches still focus on the information technology to be designed. However, complex systems consist of various resources with different characteristics. Several studies [3,4,9] point out that the automation of work, introducing new information technology within a productive environment, may lead to serious dependability problems and often results in non-effective use of new systems. Accidents involving complex technology are often caused by a combination of organisational, managerial and technical factors [15]. Literature on accidents, incidents and near misses in safety-critical systems [14,16] stresses that both the weaknesses and the strengths of a system are in the interaction among resources.

Recent research [19,21] in requirements engineering encourages the adoption of hybrid methodologies already early in the design process. The relationships among resources co-operating in a productive process need further clarification in order to obtain an integrated requirements engineering approach.

The paper points out the issues of integrating different types of requirements, which are defined over Software,

Hardware and Liveware. The SHEL model [5] provides an integrated view over these resources within a specific Environment. Section 2 introduces an integrated requirements engineering approach. Section 3 shows a railways case study. The SHEL oriented requirements engineering approach has been applied for the definition of functional, cognitive and quality requirements for a new Railways Traffic Control (RTC) system. Section 4 discusses functional, cognitive and quality requirements and presents a template to integrate the different types of requirements. Furthermore, insights from eliciting requirements and integrating diverging views for the case study are discussed. Section 5 summarises our conclusions and further work.

2. A SHEL oriented requirements engineering approach

The SHEL¹ model [5] supports a systemic view defining any productive process as performed by a combination of Hardware (e.g. any material tool used in the process execution), Software (e.g. procedures, rules, practices, etc.) and Liveware (e.g. end-users, stakeholders, etc.) resources embedded in a given Environment (e.g. socio-cultural, political, etc.). The knowledge required to perform a specific process can be considered as being distributed among the

* Corresponding author. Tel.: +44-131-6508602; fax: +44-131-6677209.

E-mail address: mas@dcs.ed.ac.uk (M. Felici).

¹ The denominator SHEL stands for Software, Hardware, Environment and Liveware.

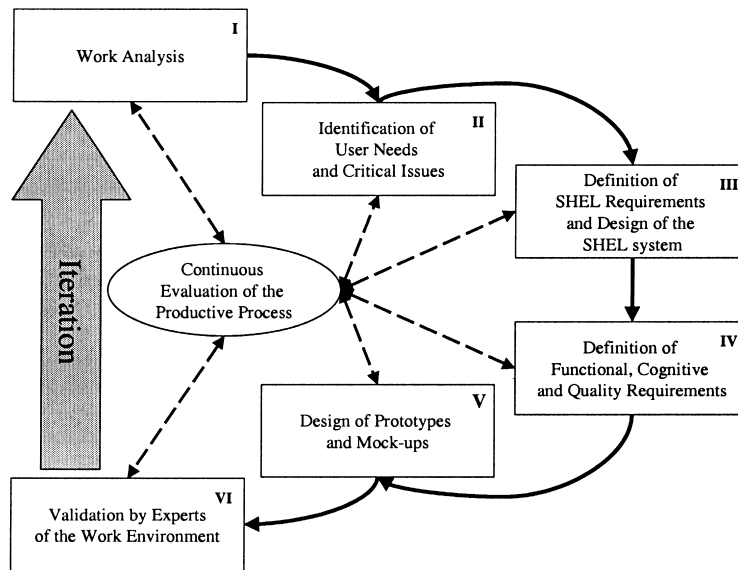


Fig. 1. The phases of the process adopted to define requirements.

system resources. Thus, a productive process may be regarded as an instantiation of the SHEL model for a specific process execution. The systemic view of the SHEL model encourages the definition of requirements not just for the system to be designed (Hardware and Software), but also for those aspects related to Liveware resources (e.g. human roles, interactions, help in breakdown-situations, etc.). Therefore, SHEL oriented requirements represent a trade-off among Hardware, Software and Liveware resources in a given Environment. Further discussion of the SHEL model can be found in Refs. [17,24].

Fig. 1 shows a SHEL oriented approach for the definition of requirements. The first step is a profound analysis of the work system of the specific environment. The knowledge required to perform a productive process, both in normal and abnormal conditions, is captured. The work analysis studies the way in which the productive process is performed taking into account the process itself and all the resources that contribute and interact in the process execution. The analysis uses different counterbalancing techniques of knowledge gathering, including observations, interviews, ethnography, heuristic analysis methods, video recording, and checklists. Representative models of processes, tools supporting the process performance, objects in the work processes, interactions, social and work practices, formal procedures, critical issues, and human skills and behaviours are produced to describe the existing work system with its critical issues and weaknesses. Reports and models for knowledge representation are, e.g. scenarios, narrative descriptions, templates and other task models. The generation of these reports and representative models involves a continuous iteration of analysis, modelling, and evaluation by the users until a satisfactory common understanding of the work system has been reached. In the analysis scenarios are an

important instrument to underline current weaknesses in certain problem situations, to communicate with end-users and get feedback from them and to link critical issues to user needs and later on to requirements. An overview of different potential applications of scenarios can be found in Refs. [8,20].

As soon as enough information is collected by the work analysis the focus changes to the weaknesses of the actual system by eliciting critical issues due to the knowledge distribution among the SHEL resources and their interaction. The results are organised in terms of the critical issues and user needs represented according to the SHEL model, which is also used to link critical issues within the productive processes, with functional, cognitive and quality requirements. The outcome of the second phase also provides a basis for alternative design considerations taken later on in the engineering process. In our understanding designing or re-engineering a system implies a redistribution of knowledge, which includes also dynamic task allocation and job design. The underlying idea of the second phase is to figure out suitable knowledge distributions, which enable an effective use of the resources in order to perform the specific productive process. The basic assumption is that there are no exclusive combinations of the three principal SHEL resources to shape a specific process, hence the same process can be performed using different possible combinations of the Hardware, Software and Liveware resources. Already early in the design process, this can be reflected by alternative design solutions represented by various prototypes and design models.

In the third phase of the process all the collected information contributes to defining the requirements and architecture of the system according to the SHEL model. The SHEL requirements and architecture of the system represent the starting point for defining functional, cognitive and quality

requirements, which is the fourth phase of the process. It is discussed in detail in Section 4.

The phase of specifying the system requirements is followed by a design phase, where the requirements are mapped into design patterns represented in prototypes, mock-ups, design models and scenarios.

The last phase of one iteration cycle in the SHEL oriented requirements engineering approach is the validation by the domain experts. The activity concerns a proof of the projected system in the real world, where system compliance with the requirements is evaluated, and where the requirements and the projected system are validated in the real system environment.

The holistic approach employs a continuous evaluation and verification of the analysis results, requirements and alternative design solutions during the whole design phase. Some influence due to the introduction of a new system within a productive process can be predicted, but the real impact can only be evaluated during testing and application in practice. The use of prototypes [2,10] together with user involvement are effective instruments to validate possible modifications before the final implementation. Simulation using mock-ups and prototypes together with a set of pre-defined scenarios [6,22] ease the communication and feedback with the users in order to improve and validate the requirements and to anticipate bad design solutions or impacts to the environment. Various architectural solutions and implementations are evaluated with the user by refining the prototypes and repeating the analysis and design parts in an iterative way. Evaluation takes into account directly measurable criteria such as performance (e.g. number of trains dispatched, maintenance work completed) and also criteria such as usability, cognitive workload and level of cognitive support.

3. A railways case study

The case study consists of a RTC system for which the requirements have been defined in co-operation with the Italian National Railways (Ferrovie dello Stato-FS). The system is to support the dynamic rescheduling of the railways resources (e.g. train paths, vehicles, maintenance units, etc.) by human agents in case of unplanned events. The planning activity as currently conducted by FS is divided into seasonal time-table planning, contingency planning to take into account special short term demands, and real-time planning to react to unforeseen events. The project focuses on the real-time planning on a particular line of the Italian railways network which can be characterised by the need for high responsiveness, rapid decision taking and the existence of multiple conflicting targets. Important requirements for the RTC system to be developed during the project are:

- increased quality of service (e.g. rescheduling the rail

resources via real-time planning in the case of disturbances, providing customer information, managing breakdown situations);

- more efficient use of infrastructure, personnel and rolling stock;
- dynamic management of transport resources.

The RTC system concerns the control and reactive management of the train movements within a certain control zone. The FS have a heterogeneous infrastructure and information technology support. There exist mainly three work environments where the control process is carried out by different systems. In the first one the work is carried out by a quasi manual RTC system, in which the train positions are graphically registered by hand on a specific paper table. The oral communication is the principal means of interaction even during emergencies. It takes place by face-to-face interaction, microphone or telephone. In the second environment the operator works with more technical instruments providing an automatic monitoring system for the railways traffic. In the third environment the control system allows the operator to act directly on the line by switching junctions and setting protection signals of stations. In this case, stations can also be unmanned, and so the stations are managed remotely.

During normal activities, the control operator observes the train movements and computes the deviations of the real data from the planned time schedule. The assessment of the deviations is based on the gravity of the deviations, formal rules, operator's experience and on other environmental factors. The overall goal of the operator's work is to make decisions for rearranging the productive resources and objects based on their assessment of deviations in order to avoid or minimise inconveniences or even incidents that may lead to economical damage or harm to humans. In order to react appropriately, especially in emergency cases, the operators need to have in any moment an understanding of the dynamic system. They use several artefacts for keeping the necessary knowledge alongside.

The operator has the supervisory control of the objects in use in their control zone as far as the movement is concerned [1]. Yet, the direct responsibility and management of the different resources (e.g. humans or machines) is assigned to other departments as defined in the organisational structure of the FS. Hence, the operator has to collaborate and coordinate with colleagues of other departments. Further characteristics of the operator's work are economical and safety related criticality, high dynamism and little predictability, unforeseen events. These events trigger and require quick decision-making, involvement of many objects, need of sufficient knowledge, skills and experiences, consideration of decisions in the context, varying workload.

Notice that the control process involves Hardware, Software and Liveware resources. Examples for Hardware resources in the less advanced RTC system are the time/station maps, pencils with different colours, the

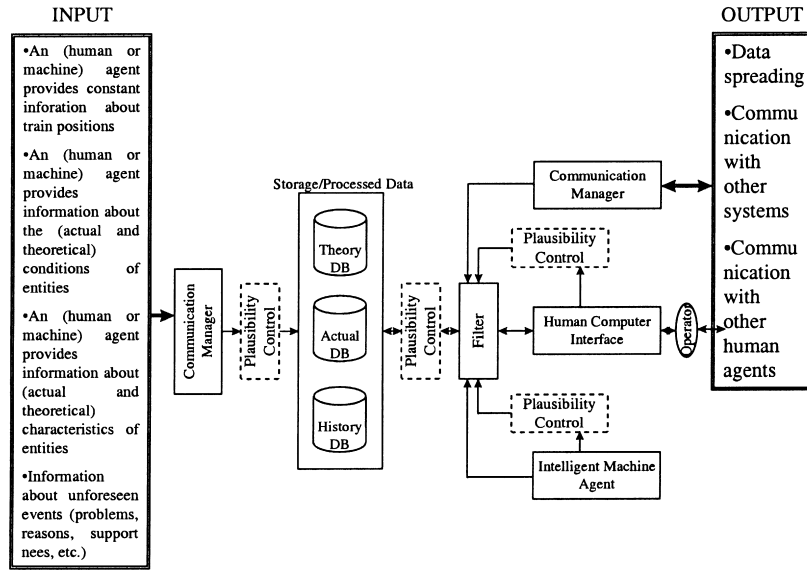


Fig. 2. A SHEL oriented architecture.

microphones and telephones. The procedure of how the station manager communicates a train position to the control operator may be seen as an example for a Software resource. The control operator or the station manager are examples of Liveware resources. All these resources embody knowledge that contributes to performing the RTC process. As changes are implemented to perform the control process, the knowledge distribution changes among the resources in the overall system environment. Indeed, the second work environment we observed is equipped with a monitoring system visualising the actual train positions on the line to be controlled to the control operators. The knowledge embodied in the time/station maps, pencils and rules are substituted by the monitoring system. Knowledge in the Liveware has changed too. The way the train trajectory is represented has changed to actively moving numbers instead of lines. The colours have changed too. The operators' activity has changed: they monitor the control system instead of continuously hiring and registering the train positions.

Over time the availability, adaptability and flexibility of the resources within the environment affect the control process. Improving the overall system, therefore, requires a careful consideration and re-distribution of knowledge among the resources as well as an effective design of the interactions. Requirements may not only be defined for functionality of the resources in isolation, but also for their interaction, for breakdown or non-availability. As human operators represent a crucial resource in the control process, special focus needs to be placed on cognitive requirements supporting user needs especially in critical situations. Moreover, the complex interrelations among the resources require careful consideration and necessitate the definition of quality requirements (e.g. reliability) for the control system.

4. Integration of functional, cognitive and quality requirements

The section describes shortly the functional, cognitive and quality requirements defined for the case study as well as the attempt of integrating them by the means of the SHEL model. Fig. 2 shows a SHEL oriented architecture representing system aspects as well as work aspects. The SHEL oriented architecture emphasises the boundary of the system and its interaction with other systems or external human agents.

4.1. Functional requirements

Functional requirements describe the functionality each system component needs to contribute to achieve the process goals. From the knowledge distribution point of view this means describing the process knowledge allocated to a specific system resource according to the SHEL model. In addition, "interaction" requirements describe the interaction among system components.

In general, functional requirements are extracted from an exhaustive analysis of the existing work environment with elicitation of critical issues and weaknesses of the current knowledge distribution among the SHEL resources. Moreover, the definition of the overall system architecture and the specification of general system requirements give a frame for detailing specific functionality in the assigned system component. Other sources bypassing the functional requirements specification are expert knowledge from the different members participating in the design team [12], technological advances influencing the projected system, as well as overall project goals driving the organisation to

Table 1
Internal reliability and target values for the System Monitor

System component	Quality characteristic	Quality subcharacteristic	Quality subsubcharacteristic	Attribute	Target
System Monitor	Reliability	Estimated reliability Correctness Test accuracy System evolution trend	Test coverage	Mean time between failures Module fault density Function coverage Branch coverage Trend of module modification Trend of requirement modification	$\geq 1 \times 10^3$ h ≤ 10 faults per module = 100% $\geq 70\%$ To be defined on first changes to be defined on first changes

change the existing system. Table 2 shows an example of the functional requirements for the RTC case study.

4.2. Cognitive requirements

Cognitive requirements are non-functional requirements, which derive from the psychological demands of the humans working within the system. These are frequently neglected and often have led to incidents or system under-utilisation. It is generally agreed that humans are an essential part in the operation of complex systems, because they are better able to handle unforeseen events than machines are. However, frequently they are considered as part of the environment of a computer system to be designed, rather than part of the system itself. This fails to take into account the importance of the interaction of the humans and the various artefacts. More promising is the Distributed Cognition approach [11], which maintains that a complex system is composed of humans, hardware and software artefacts. The knowledge is distributed over all the components of the complex system. Cognition cannot be understood by analysing or observing the human in isolation, but rather by analysing the complex socio-technical system as a cognitive entity. This allows understanding that work processes are performed by a dynamic combination of the different system resources (i.e. Hardware, Software and Liveware).

The conception of distributed cognition leads to a variety of new, cognitive requirements, which the system has to meet. Since the knowledge required to perform a process is distributed over the resources, a generic cognitive requirement is that the required knowledge for a human to react to unforeseen events needs to be available to him/her. Examples of cognitive requirements are “Status of the support system should be obvious at all times”, or “Train path monitoring should be done such that data is available after possible support system breakdown”. The cognitive, non-functional requirements will map eventually onto functional requirements, some of which may be implemented by a dynamic function allocation scheme, e.g. “All functions offered by the support system concerning the selection of train paths need to be executable also manually”, or “All functions offered by the support system need to offer a facility to switch them on or off”. Another example (not implemented by dynamic function allocation), with respect to the above non-functional requirement is “Train path monitoring should be done redundantly on a permanent hardware medium”. Such a redundant distribution of knowledge allows performing the process even after the breakdown of some of the involved resources.

4.3. Quality requirements

The quality requirements for the control system have been extensively discussed in Ref. [7]. The software architecture for the control system consists of three main components (four with the operator), the System Monitor, the Man Machine Interface, and the Decision Support System. Each

Table 2
An example of a classification of SHEL requirements and types of requirements

Function ID	SHEL requirements		SHEL space resources				Type of requirements		
	Description		S	H	L	SHL	F	C	Q
1		Provide information				*	*	*	
	1.1	Reliable information		*	*			*	*
	1.2	Required information	*	*	*		*	*	*
	1.2.1	Characteristics of trains		*			*		
	1.2.2	Characteristics of tracks		*			*		
	1.2.3	Characteristics of stations		*			*		
	1.3	When it is required — to have it at any time	*	*	*		*	*	*
	1.3.1	As quickly as possible	*	*				*	*
	1.3.2	Long term (as feedback)	*	*			*		
	1.3.3	Constant/permanently available information	*	*	*		*	*	
	1.4	Adaptive information, when it is needed	*	*				*	
	1.5	Adaptive — how to display	*	*				*	
	1.6	Provide feedback from actual status of the entities involved in the system	*	*	*		*	*	
	1.6.1	For theoretical planning	*				*		
	1.6.2	For decision making (from history of movements)	*	*	*		*	*	
	1.7	Visualise deviation from actual to theoretical	*				*		*

component has a different task, hence different quality requirements.

According to the standard ISO 9126 [13], the quality model consists of a set of important quality characteristics for the final product. Quality subcharacteristics and attributes refine the quality model. Attributes are directly measurable software properties that quantify the quality sub-characteristics. Quality characteristics and sub-characteristics can be internal or external representing respectively the design view and the end-user view. Internal quality attributes are measurable properties of the software product on its development process that influence the final product quality. The internal attributes are related to one or more external quality characteristics of the quality model according to the quality view that is suitable for the specific project.

A set of quantitative targets identify specific boundaries on the internal quality attributes, which are also used to monitor the project progress. Once a product is undergoing final testing or has been released to a limited user population, it is necessary to measure the actual quality achievement. Thus, the measures over the external quality attributes are compared with the foreseen targets to confirm that the software product meets the requirements. These requirements, usually referred to as quality requirements, describe the user needs and are quantified in terms of values (targets) for specific external attributes.

Table 1 shows the internal reliability defined for the system monitor of the case study together with the target

values for the attributes. We do not describe the entire quality model for the sake of brevity. In order to identify the quality characteristics and to define the quality requirements for the project, the work analysis was quite useful. Work analysis results allow to better understand the integration of the system into the work environment. It is possible to analyse deeply the work performed by the operator and the system. The user viewpoint has been used to refine the characteristics of the quality model. The quality requirements are based on the system aspects pointed out from the user.

4.4. Requirements integration

The section shows all the issues that we encountered to integrate the different types of requirements of the case study. Integration of functional, cognitive and quality requirements is not an easy task. The discussions we had during the project were characterised from the different viewpoints each of us had — biased from our background discipline. We took into account separately functional, cognitive and quality requirements. Hence, the first phase of the work was to encounter the different types of requirements as described above. During these discussions it was also important to realise and accept the different viewpoints on system characteristics the others had developed. Finally, our discussions aimed to identify a suitable way for integrating soundly and reasonably our work.

Table 2 shows an example of the SHEL requirements

classified in terms of type of resources² (i.e. Software, Hardware and Liveware) and type of requirements (i.e. Functional, Cognitive and Quality). The requirements are the result of the work analysis and the synthesis of user needs. The SHEL view and the classification of requirements provide a holistic design perspective, which is useful not only for designing the control system, but also for defining new activities and procedures that need to be designed together with the new system.

The following points summarise our experiences and lessons learnt for integrating functional, cognitive and quality requirements:

- Classical requirements engineering approaches do not support the integration of different types of requirements.
- Most of the requirements engineering approaches focus on the process of defining requirements without taking into account specific product aspects.
- The attempt of integrating different methodologies (e.g. work analysis and software quality) dealing separately with functional, cognitive and quality requirements resulted to be costly.
- The relation between different types of requirements (e.g. functional–cognitive, quality–cognitive and functional–quality) is not clear.
- The different types of requirements represent redundancies, which may increase inconsistencies. An effective representation of integrated requirements can improve our ability to assess their completeness and correctness.
- An effective representation of integrated requirements could improve our ability to manage changes in the requirements. The origins of requirements evolution are due to the different stakeholders involved in the development process. Hence any change in the specific environment will affect requirements. Managing integrated requirements can be an effective way to obtain change tolerant processes and requirements.

5. Conclusions and further work

The paper points out a set of issues to integrate different types of requirements, which have been defined by a systemic requirements engineering process taking into account a holistic view of the system. The identified issues represent a checklist for comparing requirements engineering approaches. Currently most of the methods in requirements engineering do not support all the features required for an integrated methodology. Moreover, the set of issues for integrating requirements represents a work plan suggesting possible research directions and updates for requirements engineering methodologies. We strongly suggest to

tackle all the issues identified in order to obtain integrated requirements specifications fitting the holistic expectations.

Acknowledgements

We are grateful to Alberto Pasquini (ENEA, Italy) and Antonio Rizzo (University of Siena, Italy) as well as some people of the OLOS Network (Human Capital and Mobility Programme of the European Community Third Framework Programme) for their support, supervision and comments to this work and our attempt to integrate software, hardware and human aspects in requirements engineering. A special thank goes to the Italian National Railways (FS), who provided the case study and allowed us to interact with system operators spending their time.

References

- [1] L.J. Bannon, S. Bodker, Beyond the interface. Encountering artifacts in use, in: J.M. Carroll (Ed.), *Designing Interaction: Psychology at the Human–Computer Interface*, Cambridge University Press, Cambridge, 1991, pp. 227–253.
- [2] R. Budde, *Approaches to Prototyping*, Springer, Berlin, 1994.
- [3] C.W. Clegg, P.B. Warr, T.R.G. Green, A. Monk, N.J. Kemp, G. Allison, M. Landsdale, *People and Computers: How to Evaluate your Company's New Technology*, Ellis Horwood, Chichester, 1989.
- [4] K.D. Eason, *Information Technology and Organisational Change*, Taylor and Francis, London, 1988.
- [5] E. Edwards, Man and Machine: Systems for Safety, Proceedings of British Airline Pilots Associations Technical Symposium, British Airline Pilots Associations, London, 1972 (pp. 21–36).
- [6] T. Erickson, Notes on design practice: stories and prototypes as catalysts for communication, in: J.M. Carroll (Ed.), *Scenario-Based Design: Envisioning Work and Technology in System Development*, Wiley, New York, 1995.
- [7] M. Felici, A. Pasquini, S. De Panfilis, Software quality in user-centred design, Proceedings of ESCOM-ENCRESS 98, Rome, Italy, 27–29 May 1998, pp. 239–247.
- [8] D. Filippidou, Designing with scenarios: a critical review of current research and practice, *Requirements Engineering* 3 (1) (1998) 1–22.
- [9] P. Hornby, C.W. Clegg, J.I. Robson, C.R.R. MacLaren, S.C.S. Richardson, P. O'Brien, Human and organisational issues in information systems development, *Behaviour and Information Technology* 11 (3) (1992) 160–174.
- [10] S. Houde, C. Hill, What do prototypes prototype?, in: M. Helander, T. Landauer, P. Prabhu (Eds.), *Handbook of Human–Computer Interaction*, 2nd ed., Elsevier, Amsterdam, 1997.
- [11] E. Hutchins, *Cognition in the Wild*, MIT Press, Cambridge, MA, 1995.
- [12] ISO/DIS 13407, Human-centred design processes for interactive systems. ISO, 1999.
- [13] ISO/IEC 9126, Information Technology — Software Quality Characteristics and Metrics.
- [14] T. Kletz, *Lessons from Disaster*, Gulf Publishing Company, Houston, 1993.
- [15] N. Leveson, *Safeware*, Addison-Wesley, Reading, MA, 1995.
- [16] C. Perrow, *Normal Accident: Living with High-Risk Technologies*, 2nd ed., Princeton University Press, Princeton, NJ, 1999.
- [17] A. Rizzo, A. Pasquini, P. Di Nucci, S. Bagnara, ShELFS: Managing critical issues through experience feedback, *Human Factors and Ergonomics in Manufacturing* 10 (1) (2000) 83–98.

² Requirements classified as SHL are those which involve all the SHEL resources. The classification depends on the granularity of the representation of requirements.

- [18] I. Sommerville, P. Sawyer, *Requirements Engineering: A Good Practice Guide*, Wiley, New York, 1997.
- [19] I. Sommerville, P. Sawyer, Viewpoints: principles, problems and a practical approach to requirements engineering, *Annals of Software Engineering* (1997) 101–130.
- [20] A. Sutcliffe, Scenario-based requirements analysis, *Requirements Engineering* 3 (1) (1998) 48–65.
- [21] S. Viller, I. Sommerville, Social analysis in the requirements engineering process: from ethnography to method, Technical Report CSEG/14/1998, Lancaster University, 1998.
- [22] K. Weidenhaupt, K. Pohl, M. Jarke, P. Haumer, Scenarios in system development: current practice, *IEEE Software* 15 (2) (1998) 34–45.
- [23] K.E. Wiegers, *Software Requirements*, Microsoft Press, 1999.
- [24] M. Wimmer, A. Rizzo, M. Sujan, A holistic design concept to improve safety related control systems, *Proceedings of the International Conference on Computer Safety, Reliability and Security (Safecom99)*, LNCS 1698, Springer, 1999, pp. 297–309.