ICED13
INTERNATIONAL CONFERENCE ON ENGINEERING DESIGN, ICED13
19-22 AUGUST 2013, SUNGKYUNKWAN UNIVERSITY, SEOUL, KOREA

AN EMPIRICAL INVESTIGATION OF REQUIREMENT EVOLUTION IN AN INDUSTRIAL PROJECT

Weili DAI, Marco AURISICCHIO
Imperial College London, United Kingdom

ABSTRACT
In the development process of complex systems, a range of design and system engineering methods are typically applied to analyse and validate requirements. This paper has reported an empirical investigation of documents which have been generated as a result of applying these methods. The investigation focuses on requirements evolution, one of the components of requirement analysis. Requirement evolution involves checking and structuring of requirements and leads to their refinement. The aim of this research is to understand how requirements evolve as a result of applying engineering methods, and to draw insights towards capturing requirement evolution. The motivation behind this work is to establish gaps in the current requirement analysis support tools in order to create an effective requirement analysis workflow. Four operations performed on requirements were used to characterise requirement evolution. The research has revealed that current computational support for design and systems engineering methods lacks means of visualising and capturing requirement evolution. The paper has also highlighted the opportunities to provide justification and clarification in the requirement analysis process.

Keywords: requirements, requirement analysis, design rationale, information management, communication

Contact:
Weili Dai
Imperial College London
Mechanical Engineering
London
SW7 2AZ
United Kingdom
wd206@imperial.ac.uk

1 INTRODUCTION

Requirement analysis consists of checking, structuring and evolving requirements (Dai, Aurisicchio and Armstrong, 2012). It is a dynamic and iterative process (Nuseibeh, 1996) that constantly requires stakeholders' attention. As part of this process, requirements are continuously refined from abstract needs to structured and detailed technical requirements as well as updated due to changes in resources, technology feasibility and stakeholders' preferences. Requirement validation is an aspect of requirement analysis, which consists in checking requirements. In the development process of complex systems, a range of design and system engineering methods are typically applied to validate requirements, e.g. Viewpoint Analysis and Quality Function Deployment. As a result of the application of these methods, a set of requirements evolves and an engineering team builds confidence in it. Tracing the evolution of requirements over the application of these methods and capturing the rationale for their evolution is important to support requirement analysis, communication of changes to requirements as well as product verification and validation strategies.

The aim of this research, in collaboration with the engineering business of a large power system company, is to understand how requirements evolve as a result of the application of methods for requirement analysis and validation, and to draw insights towards capturing requirement evolution. This paper reports an empirical investigation of the requirements identified and analysed by a team of graduate engineers from the collaborating company involved in the design of a portable machine to take material samples from surfaces. The requirement analysis undertaken by the graduates focuses on Voice-Of-Customer and systems requirements. Requirements were analysed using five methods known as Issue Based Information Systems, Function Analysis, Systemic Textual Analysis, Viewpoint Analysis and Quality Function Deployment. These methods were applied with computational support including the Decision Rationale editor (DRed) and MS Excel. The results indicate that when a set of requirements is analysed through the successive application of design and systems engineering methods, several types of evolution takes place. Yet, the evolution of requirements is often not captured and justified. This paper is of interest for academics and practitioners involved in the areas of system engineering and requirement engineering as well as for developers of software solutions to support requirement engineering practice.

2 BACKGROUND LITERATURE

This section reviews literature, which has inspected the concept of requirement analysis as well as methods and tools to support it.

2.1 Requirement analysis

In (Dai, Aurisicchio and Armstrong, 2012), a model of requirement analysis was introduced consisting of three phases: performing checks on requirements; structuring requirements; and evolving requirements as a result of the checks and structuring.

Requirement checking, also referred to as requirement validation, is a process to discover problems (Rzepka, 1989) and answer the question 'Have we got the right requirements?' (Kotonya and Sommerville, 1998). It includes validation of both individual requirements and the requirements as a set. Properties of individual requirements typically subject to checking include clarity, necessity and feasibility. Properties of the whole set of requirements that are subject to checking are completeness (Systems engineering fundamentals, 1991; Kotonya and Sommerville, 1998; Ott 2012), uniqueness (Heumesser et al., 2004), and freedom from conflicts (ISO15288, 2000). Robertson and Robertson (1999) make a similar distinction; they discuss a procedure known as 'Quality Gateway' to check requirements individually, and a separate procedure known as 'Stocktake' to review a set of requirements as a whole. Alexander and Stevens (2002) also distinguishes between checking a single requirement and checking requirements as a set.

Requirement structuring is a process to build a structure around requirements and decomposing them. Structuring requirements into hierarchical levels is recommended in (Andersson, Sutinen, and Malmqvist, 2003; Stoller, 1988). Hull et al. (2010) describe structuring as making requirement documents into hierarchies, with sections and subsections. Robertson and Robertson recommends to structure requirements according to the Volere template (Robertson and Robertson, 1999). The template organises requirements in a hierarchy of categories and sub-categories. The use of hierarchies to structure requirements is necessary so that complex and large volume requirements can be managed.

A similar view is shared by Grady (1993), who advocates that decomposition is central to the process of requirement analysis that transforms customer needs into system requirements.
Requirement evolution is 'a process to recognise change through continued requirement elicitation, re-evaluation of risk, and evaluation of systems in their operational environment' (Easterbrook and Nuseibeh, 1995). Essentially it is a process of making refinements, including refining customer and performance objectives (Systems engineering fundamentals, 1991). Refinement can be either to correct the scope of a requirement or to correct an inconsistent requirement. Heumesser et al. (2004) argue that refinement can either decrease the scope of a project through specialisation or increase its scope through tailoring. Kotonya and Sommerville (1998) recognise refinement as resolving unnecessary, conflicting, or infeasible requirements. Regardless of the purpose of refinement, the basic operations to be performed to refine requirements include addition, deletion, and updating of requirements. 'Typical changes to requirement specifications include adding or deleting requirements, and fixing errors' (Nuseibeh and Easterbrook, 2000). Refining requirements is closely associated with clarification – also an activity in requirements checking. Robertson and Robertson (1999) describe updates as 'clarification' and 'revision', and deletion as 'discarding' of requirements.

2.2 Techniques and tools used for requirement analysis

A common method to analyse requirements consists of structuring them as a tree is. It normally begins by grouping an arbitrarily ordered set of requirements. It allows, among others, checking a requirement set for missing or conflicting requirements. Methods commonly used for this purpose are the Affinity Diagram and Viewpoint Analysis (Burge, 2011). The Affinity Diagram is a method to sort requirements in a tree structure following elicitation during a brainstorming session. It is often used as a precursor to more formal recording of requirements (CIRI, 2011; Crow, 2011). Viewpoint Analysis (VPA) has a more stringent format that separates requirements into types such as external and internal, and functional and non-functional. The tree structure of the VPA shows a hierarchical decomposition of the system functions. It also shows the impact and constraints exerted by non-functional requirements. VPA has two purposes. First, it helps stakeholders build a common view of the structure of requirements. Second, it highlights requirements missing from the original customer requirements, because customers rarely provide a complete set of system requirements. In (Dai, Aurisicchio and Armstrong, 2012) an approach was introduced to map non-functional requirements with a tree structure using the IBIS notation, which put emphasis on the capture of design rationale to justify requirements.
Apart from the tree structure, requirements can also be represented in tables and matrices. In Systemic Textual Analysis (STA) (Burge, 2004), a table is used to sort requirements into columns including operational, functional and non-functional. The table is generally used to identify missing requirements as there is usually one or more functional requirements associated with a non-functional requirement. According to the guide for this technique in (Burge, 2004), a textual analysis is applied to identify customer requirements, which are typically expressed as textual statements with no particular order or grouping. It is worthy to note that during the interpretation of these statements it is permitted to expand, clarify and decompose identified customer requirements. Quality Functional Deployment (QFD) employs a relationship or traceability matrix. It is used mainly to ensure that the voice of customer is fully acknowledged and translated in the technical requirements, and consequently into the design. Robertson and Robertson (1999) claim that QFD's matrix of interdependencies can help identify conflicting requirements.

3 REQUIREMENT ANALYSIS IN AN ENGINEERING PROJECT

This section describes the research methodology, introduces the Decision Rationale editor (DRed), provides an overview of the engineering design project studied, and presents an analysis of requirement evolution.

3.1 Research methodology

The approach taken to understand requirement evolution aligns with the Design Research Methodology (Blessing and Chakrabarti, 2002), which distinguishes four research stages: Research Clarification, Descriptive Study I, Prescriptive Study and Descriptive Study II. The research consisted in studying engineering design practice in the collaborating company including use of requirement analysis methods with their software support. It, therefore, started from and focused on the Descriptive

Study II stage with the aim to characterise practice. Subsequent steps to be undertaken in the Prescriptive Study stage would consist of extracting requirements to improve computational support available to engineers.
Design data were collected from a team of graduate engineers involved in a Design and Make project part of the collaborating company's training programme. This particular project was chosen because it followed an approach to requirement analysis similar to that employed on core business programmes by system engineering experts in the collaborating company. Two types of data were collected from this project. The first is a set of project documents including DRed and MS Office files. Among these, five documents were used by the design team to perform requirement analysis and validation. The second type of data consists of a transcript from an interview with a member of the project team. The interview was used to understand the purpose of each project file and their order of creation. It was conducted over the phone and the conversation was supported by the use of the project files previously shared. During the interview, the overall workflow of requirement management was discussed.

3.2 The DRed tool

The Decision Rationale editor (DRed) is now introduced because the graduates chose to use it to support some of the requirement analysis methods adopted, and because of the aspiration by the authors to use it in the long term as a platform to support a multi-method approach to requirement analysis and validation. The Decision Rationale editor is a software tool which was developed to enable graphical capture of design rationale (Bracewell and Wallace, 2003; Bracewell, Ahmed, and Wallace, 2004; Bracewell et al, 2009). The tool uses diagrams for rapid mapping of design information, seeing patterns and achieving insights (Eng, Bracewell, and Clarkson, 2009). Following its successful application for design rationale capture, it was found that the tool provided benefits also in laying out root cause analyses undertaken during failure investigation (Bracewell et al., 2009). Subsequent research has focused on the capture of the functional interactions between the physical elements of a system (Aurisicchio, Bracewell, and Armstrong, 2012). More recent work has explored its applicability for the capture of requirements and their rationale (Aurisicchio and Bracewell, 2012). DRed differs from other IBIS-based tools as it does not need a dedicated database, which makes it compatible with existing document management practices.

3.3 The engineering design project studied and its requirement documents

The aim of the project undertaken by the graduates was to design a portable machine for material sampling. The project was accomplished by a team of four engineers over a twelve-week period. Initially, the team was given three requirements from the customer. In order to elicit, analyse and validate the requirements, the team applied several design and systems engineering methods and analysed existing products that were similar to the desired solution. The requirements that the team was able to identify are presented in Figure 1 according to the method used to study them. It can be seen that five requirement documents were used known as Issue-Based-Information-System (IBIS), Function Analysis (FA), Systemic Textual Analysis (STA), View Point Analysis (VPA) and Quality Function Deployment (QFD), see Figure 1.
The process of creation of the five documents is illustrated in Figure 2. The individual requirement documents are now presented explaining how they were used and relating their use to the requirement analysis model introduced in section 2.1.
The first requirement set created by the team is the Issue-Based-Information-System (IBIS) map, see Figure 1and Figure 2. This set of requirements originated from a project brief, which contained only requirements given by the customer. Subsequently, this set was expanded through interviews with stakeholders. The final set was formatted in a tree structure using DRed. IBIS was, therefore, used to structure requirements as soon as they were elicited. Grouping requirements into categories helped check the consistency of the requirements as a set. However, it is noteworthy that the members of the project team did not justify the requirements using the argumentation-based rationale approach behind DRed (Dai, Aurisicchio and Armstrong, 2012).
The requirements were not just elicited from interviews with the customer and the stakeholders. Rather some requirements were elicited through function analysis of an existing machine for material sampling. This analysis was also captured using DRed, and it is referred to as Functional Analysis (FA) map, see Figure 1and Figure 2. The map has a section showing the decomposition of the parts of the existing machine, and another section showing the decomposition of the functions. A subset of the

functions was later converted into functional requirements for the product to be designed as they had not been identified through the interviews with stakeholders. According to the interview with the member of the project team, FA was useful to identify, check and agree the complete functional behaviour and structural feasibility of the desired solution. Therefore, it can be said that FA helped create shared understanding of the system to be designed and decompose abstract functions into a sub-system of functions (Systems engineering fundamentals, 1991).
Following the elicitation of requirements using interviews and function analysis, further analysis was performed to structure and refine them. This involved use of the techniques such as Systemic Textual Analysis (STA), Viewpoint Analysis (VPA), and QFD, see Figure 1 and Figure 2. STA was used mainly to check for missing requirements. STA has three steps: separate and sort identified customer requirements; identify missing requirements; and clarify and refine requirements (Burge, 2004). STA, therefore, covers all the components of the requirement analysis model. STA is generally applied to unstructured requirement statements. However, in this case the statements were already grouped in the IBIS map with categories different to those proposed for STA, i.e. operational requirement, non-functional implementation requirements, non-functional performance requirements and functional requirements. This means that the application of STA required the development of a new structure compared to the previous one. VPA also served the role of facilitating checking, but it shifts the emphasis on visual structuring of the system requirements. QFD facilitates the checking of requirements as a set and the exploration of the relationships between non-functional and functional requirements. One way it does this is by capturing target values for requirements making them more quantified.

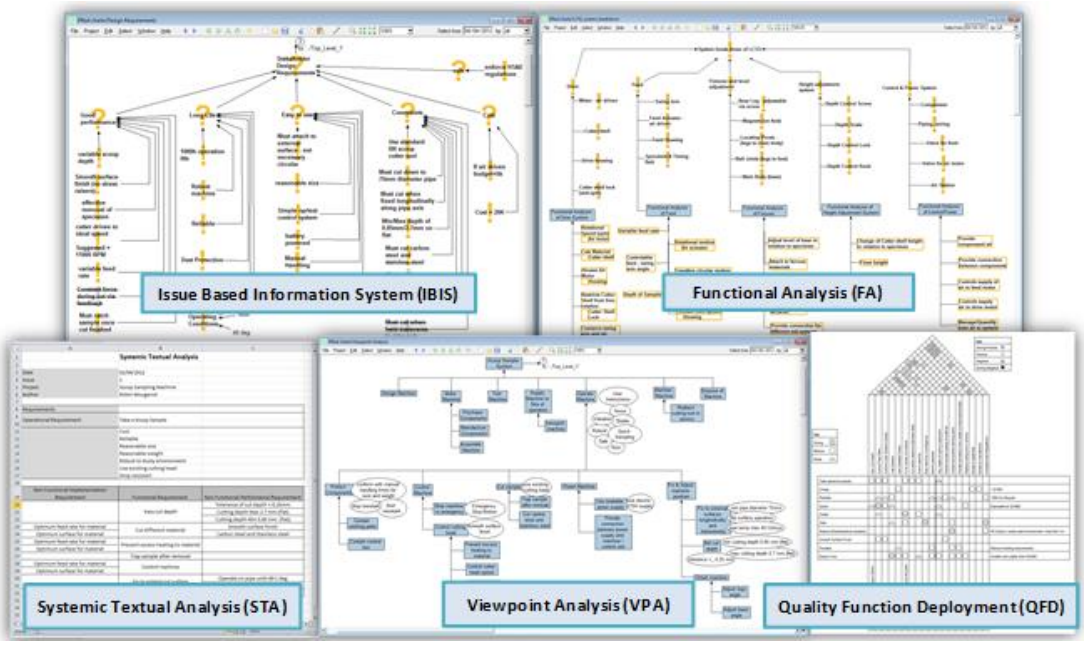

Figure 1 Requirement documents produced by the project team (This Figure is low resolution due to the confidentiality of its content)

3.4 Analysis of requirement evolution

To investigate requirement evolution across the five documents, the requirements were analysed using an Excel spreadsheet. This allowed comparing and contrasting the requirements as well as understanding how they evolved. Data analysis followed three steps. First, the requirements captured in each document were captured into a single column of the spreadsheet. Second, the requirements from all sets were matched to identify those requirements which were transferred from a previous document and those that were newly introduced. Third, all sets of requirements were categorised into non-functional and functional in order to understand how these two fundamental types evolved.
Figure 3 shows the final look of the spreadsheet at the end of the analysis process. Each column present the requirements generated through a requirement document, e.g. column three captures the requirements studied through the Systemic Textual Analysis. Each row of the column holds one requirement statement. The requirements with the grey background are non-functional and the

requirements with the orange background are functional, see Figure 3. If a requirement was present in two sets it was stored in the same row indicating that it was considered also when creating the subsequent requirement document. This analysis required interpretation as over time requirements were refined.

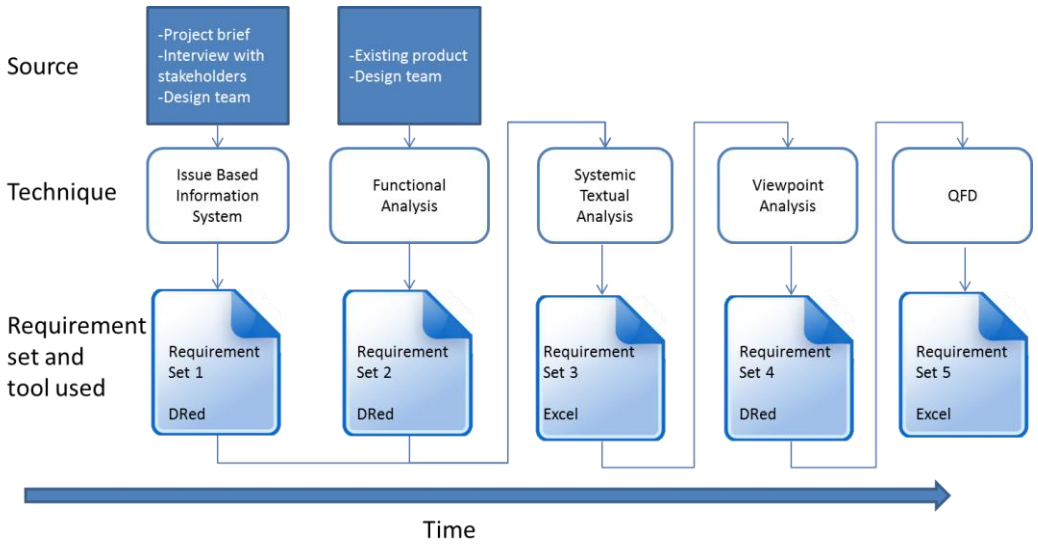

Figure 2 Requirement analysis workflow used to produce five project documents

Figure 3 Excel spreadsheet showing requirement evolution across the five documents

Based on the analysis of the five requirement documents it was found that four types of operations were carried out on requirements: new introduction, transfer as-is, transfer with refinement, and drop. Figure 4 presents a model illustrating these operations. New requirements were introduced due to changes in the project scope and resource availability. Existing requirements were transferred as-is to reflect their recognised validity and suitability to be part of the next analysis stage. Transfer with refinement was found to be needed to clarify, quantify and decompose requirements as well as to make them conflict-free. Existing requirements were dropped because deemed unnecessary.

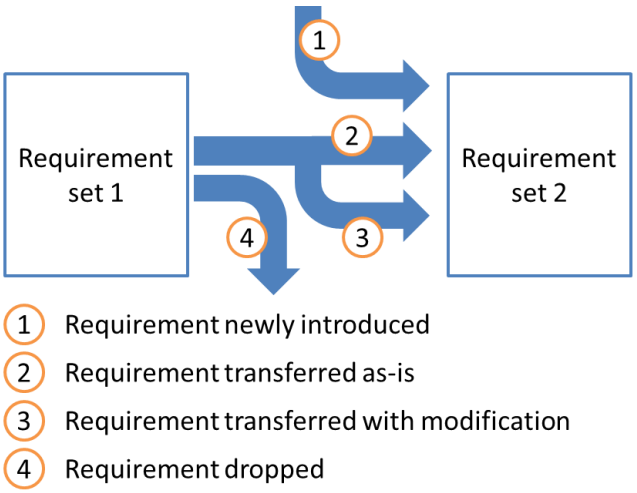

Figure 4 Model showing the evolution of requirements between consecutive documents

A quantitative analysis of requirement evolution is presented in Figure 5. The five requirement documents are shown as orange boxes ordered according to their sequence of creation. Each box has percentages to its left, showing proportions of incoming requirements; and percentages to its right, showing proportions of outgoing requirements.

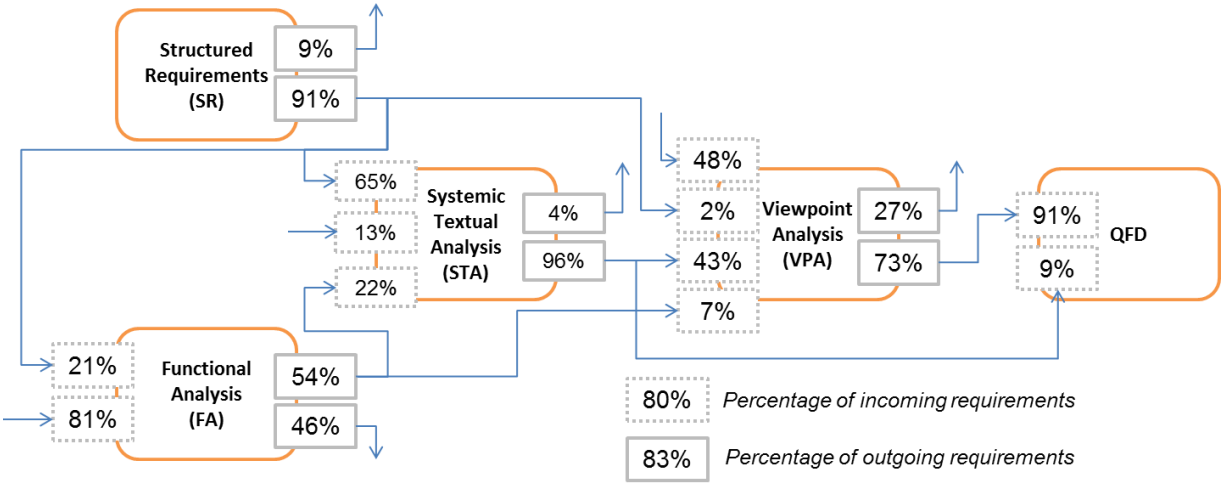

Figure 5 Diagram showing quantitative aspects of requirement evolution between project documents

In Figure 5 the proportions of transferred requirements are shown by the arrows linking a source to a destination requirement document, e.g. 65% of the requirements in the STA document were transferred from the IBIS document. The proportions of dropped requirements are shown by the arrows with no destination, e.g. 27% of the requirements in the VPA document were dropped and not used in the QFD document. Finally, the proportions of newly introduced requirements are shown by arrows with no origin, e.g. 48% of the requirements in the VPA document were newly introduced.

From Figure 5 three important findings emerged. First, there are always requirements in one document being transferred to another document. Second, there are always requirements being dropped. Third, there are at times newly introduced requirements. These findings are now presented and discussed in the context of the project studied.

From Figure 5 it can be seen that only 21% of the requirements in the IBIS document were also present in the FA document. Little overlap between the IBIS and FA documents confirms that these were two independent sources through which requirements were elicited. Figure 5 also shows that the majority of requirements in the STA document was transferred from either the IBIS or FA documents. However, 46% of the requirements identified through the FA document were not carried over indicating that they were judged not suitable. Inspecting these requirements a range of reasons can be hypothesised for why they were left behind including, for example, their dependency on a specific architectural design decisions made by the project team. If the rationale for dropping these requirements was captured it would have been easier for stakeholders to interpret the process followed

by the project team. After the STA document, the team generated the VPA document, see Figure 5. The requirements transferred from the STA to the VPA document are only 43% of the total. This is somehow surprising as at this stage of the requirement analysis one would have expected the requirement set to become firmer. Interestingly 48% of the requirements in the VPA document were newly introduced. Comparisons between the requirement statements in the STA and VPA documents show that they differ in breadth and depth. In the STA document, the requirements mostly describe product level attributes and functions. Differently in the VPA document, the project team considered a broader set of design aspects including assembly, testing, and transportation. In addition, the requirements in the VPA document are very detailed including specification at component level. From Figure 5, it can also be seen that 91% of the requirements in the QFD document were transferred from previous work in the VPA document. It is noteworthy that these were only 73% of the VPA, indicating that the VPA is a larger set. The remaining 27% of the VPA requirements that was not transferred to the QFD includes mostly requirements about the broader design aspects previously mentioned, indicating that the QFD contains only requirements related to the product in operation.

Finally two issues important to understand the above model of requirement evolution are noted. The first, not observable from Figure 5, is that compared to the VPA, the QFD contains more quantified statements. A reason for this can be found in the fact that the QFD was used as the final set of requirements upon which further design and development was based. The second is that at times the requirements dropped after the use of a method were picked up again in the method after the next. For example, 2% of VPA are requirements in IBIS, which were dropped in STA but picked up again. Similarly, 9% of the requirements in QFD are dropped in VPA but picked up again in QFD. This occurrence indicates that sometimes requirements were not continuously considered. This is again a case where capture of design rationale could help clarify why they were dropped.

4 DISCUSSION

This paper has shown that requirement analysis may involve the application of multiple methods. In the project studied, requirements were elicited through interviews whose outcome was presented in the Issue Based Information System (IBIS) tree structure, and Function Analysis (FA) of related products. Following elicitation, the requirements were investigated through Structured Textual Analysis (STA), Viewpoint Analysis (VPA) and Quality Function Deployment (QFD). The methods were used with the support of the Decision Rationale editor and MS Excel. However, they are not dependent on this software and, even in the collaborating company, they are frequently used with other software, e.g. MS PowerPoint and Qualica. The rationale for applying the methods in this sequence has to be searched in the practical experience of experts in the collaborating company and the benefits identified by the project team. The methods were used to analyse and inspect requirements from different perspectives and validate them.

Although it was not evaluated, it is expected that the methods helped the project team perform checks on requirements both as individual elements and as a whole. However, it is clear that the methods did not provide support in the capture of the reasoning developed as part of checking. The research also indicated that the methods supported the structure of requirements and the capture of emergent structures. Requirement checking and structuring produce requirement evolution. A model of requirement evolution was empirically developed based on the data analysis. This describes operations such as introduction of new requirements, transfer as-is, transfer with modification, and drop. The operations in the model of requirement evolution are similar to those outlined by Heumesser et al. (2004) including definition of additional requirements, direct translation, and refinement. The difference is that our model describes specifically the transfer of requirements from one set to another on the same abstraction level, whereas Heumesser's model focuses on the derivation of requirements from a higher level. The methods used by the project team did not provide support, every time that a new method was applied, in the visualisation of the operations performed by the project team on requirements. This is expected to help requirement analysis especially when dealing with large requirement sets. In addition, they did not support the capture of the rationale for requirement evolution, which can help engineers and other stakeholders make sense of change and create sharing understanding. As an example capturing the rationale for a dropped requirement can have an important role in explaining that it was not ignored or replaced by another requirement.

It is now worth asking how engineers involved in requirement analysis with multiple methods could be supported in visualising and tracing requirement evolution as well as capturing its rationales. A

promising and practical direction to achieve the objective to trace evolution would be to implement hyperlinking functionality across the file formats used to support the various methods. The DRed tool already supports mono- and bi-directional hyperlinking between information objects in its own files, and its files and MS Office files (Word, Excel and PowerPoint) (Bracewell et al., 2007) as well as a form of hyperlinking, known as transclusion, which allows to create linked and navigable copies of information objects (Bracewell et al., 2009). A solution to capture rationale to justify requirement analysis and evolution would consist of effective implementation of the IBIS concept. Although this did not happen in this project example, the concept remains to be explored and its technical feasibility was demonstrated in (Dai, Aurisicchio and Armstrong, 2012).

4.1 Limitations and contribution

The work presented in this paper has a number of limitations. First, the research is based on a single project. More project cases would help validate the understanding emerged. Second, the design project studied was undertaken by graduate engineers. This means that the project team was neither expert in design nor in requirement analysis. However, the graduates were formally trained in the application of design and system engineering methods, and worked under the supervision and guidance of company experts to solve a real technical problem. Third, the research is based on a problem, which was characterised through approximately thirty requirements. We must, therefore, question if the results would scale to a large and more complex problem.

This work contributes to engineering design research at two levels. First, it brings to the attention of the community a multi-method approach to requirement analysis and validation. Second, it advances our understanding of engineering requirements by characterising how they evolve through empirical analysis of a design project in industry and discussing the implication for tool support.

5 CONCLUSION

This paper has reported an empirical investigation of requirement analysis and has characterised aspects of requirement evolution resulting from the application of various design and system engineering methods. The motivation behind this work is to establish an effective workflow and tool set to support requirement analysis. The results from the project studied have shown that four operations typify requirement evolution: requirements newly introduced; requirements transferred as they appeared previously; requirement transferred with modification; and requirements which are no longer used in subsequent analysis. The research has also revealed that computational support to implement existing design and systems engineering methods lacks means of visualising and capturing requirement evolution. The paper has also highlighted the opportunity to provide justification and clarification of the requirement analysis process.

ACKNOWLEDGMENTS

The authors acknowledge the support for this research from Rolls-Royce plc and the Engineering and Physical Sciences Research Council (EPSRC) through the Collaborative Awards in Science and Engineering (CASE) studentship under grant number 09000154. This research was undertaken using the Decision Rationale editor (DRed) a software tool owned and controlled by Rolls-Royce plc.

REFERENCES

Alexander, I., and Stevens, R. (2002). *Writing Better Requirements*. Pearson Education Limited.

Andersson, F., Sutinen, K., and Malmqvist, J. (2003). Product Model for Requirements and Design Concept Management : Representing Design Alternatives and Rationale. *International Conference on Systems Engineering*.

Aurisicchio, M., and Bracewell, R. (2012). Capturing an integrated design information space with a diagram based approach. *Journal of Engineering Design*.

Aurisicchio, M., Bracewell, R., and Armstrong, G. (2012). The Function Analysis Diagram. *ASME 2012 International Design Engineering Technical Conferences*.

Blessing, L. T. M., and Chakrabarti, A. (2002). DRM: A Design Research Methodology. *Proceedings of International Conference on The Science of Design*.

Bracewell, R., Ahmed, S., and Wallace, K. M. (2004). DRed and design folders: a way of capturing, storing and passing on, knowledge generated during design projects. *ASME International Design Engineering Technical Conferences*. Salt Lake City, Utah, USA.

Bracewell, R. H., Gourtovaia, M., Wallace, K. M., and Clarkson, P. J. (2007). Extending Design Rationale to Capture an Integrated Design Information Space. *International Conference on Engineering Design, ICED'07*. Paris, France.
Bracewell, R. H., and Wallace, K. . (2003). A tool for capturing design rationale. *14th International Conference on Engineering Design (ICED'03)* , pp. 185–186. Stockholm, Sweden.
Bracewell, R., Wallace, K., Moss, M., and Knott, D. (2009). Capturing design rationale. *Computer-Aided Design*, Vol. 41, issue.3, pp. 173–186.
Burge, S. (2004). Systemic Textual Analysis (STA). Burge Hughes Walsh.
Burge, S. (2011). Viewpoint Analysis. Burge Hughes Walsh.
CIRI. (2011, March). Quality Function Deployment. *Creative Industries Research Institute*.
Crow, K. A. (2011). Quality Function Deployment. *DRM Associates*.
Dai, W., Aurisicchio, M., and Armstrong, G. (2012). An IBIS based Approach for the Analysis of Non Functional Requirements. *ASME 2012 International Design Engineering Technical Conferences*.
Easterbrook, S., and Nuseibeh, B. (1995). Managing Inconsistencies in an Evolving Specification. *2nd IEEE SYMPOSIUM ON REQUIREMENTS ENGINEERING*, pp. 48–55.
Eng, N. L., Bracewell, R., and Clarkson, P. J. (2009). Concept Diagramming Software For Engineering Design Support: A Review And Synthesis Of Studies. *ASME 2009 International Design Engineering Technical Conferences and Computers and Information in Engineering Conference, IDETC/CIE 2009*.
Grady, J. O. (1993). *System Requirements Analysis*. McGraw-Hill.
Heumesser, N., De Mets, A., Demeestere, L., Omasreiter, H., Tavakoli, R., Houdek, F., Weisbrod, J., et al. (2004). *Framework for Requirements* . Springer
Hull, E., Jackson, K., and Dick, J. (2010). *Requirements Engineering* (3rd ed.).
ISO15288 (2000). AP233 ISO 15288. *www.ap233.org*.
Kotonya, G., and Sommerville, I. (1998). *Requirements Engineering: Processes and Techniques*
Nuseibeh, B. (1996). To Be and Not to Be : On Managing Inconsistency in Software Development. *8th International Workshop on Software Specification and Design*, pp. 164–169.
Nuseibeh, B., and Easterbrook, S. (2000). Requirements engineering: a roadmap. *Proceedings of the Conference on the Future of Software Engineering*, ACM.Vol. 1, pp. 35–46.
Ott D. (2012). Defects in Natural Language Requirement Specifications at Mercedes-Benz: An Investigation using a Combination of Legacy Data and Expert Opinion, *International Requirements Engineering Conference 2012, Chicago, 2012.*
Robertson, S., and Robertson, J. (1999). *Mastering the Requirements Process*.
Rzepka, W. E. (1989). A requirements engineering test bed: concept, status and first results. *System Sciences, 1989. Vol.II: Software Track, Proceedings of the Twenty-Second Annual Hawaii International Conference on Systems Science*, Vol.2 pp. 339 –347.
Stoller, R. (1988). Tracer: a tool for tracing and control. *Engineering Management Conference, 1988. Engineering Leadership in the 90's*, pp. 27–36.
Systems engineering fundamentals. (1991). Defense Acquisition University.